

Learning Analytics in Intelligent Computer-Assisted Language Learning

D i s s e r t a t i o n

zur

Erlangung des akademischen Grades

Doktor der Philosophie

in der Philosophischen Fakultät

der Eberhard Karls Universität Tübingen

vorgelegt von

BJÖRN RUDZEWITZ

aus

Albstadt

2021

Gedruckt mit Genehmigung der Philosophischen Fakultät
der Eberhard Karls Universität Tübingen

Dekan: Prof. Dr. Jürgen Leonhardt

Hauptberichterstatter: Prof. Dr. Detmar Meurers

Mitberichterstatter: Prof. Dr. Andreas Lachner

Tag der mündlichen Prüfung: 23.06.2021

Universtätsbibliothek Tübingen

Abstract

With an increase in digitalization in all areas of society, including education, data and technology offer new and exciting potentials. Systems can diagnose potential problems of learners, systems and humans can learn from data to continuously improve concrete teaching practices, and researchers can contribute to an understanding of learning in specific domains and in general. It is not entirely clear what contributes to the incredible ability of human learning. Learning Analytics can shed light on what factors contribute to learning and can help improve education, thus having a positive impact on society. Learning Analytics allows us to learn from the past to improve the future.

While data points are generated already with every interaction in digital applications – such as tutoring systems – there need to be principled ways for making sense of and using the potential of these data. Concretely, the interdisciplinary context needs to provide guidance regarding which features to consider in the process of going from raw interaction logs to interpretable feature representations. Empirical Educational Science and pedagogy need to guide the design of systems, Learning Analytics tools, and data analyses to be relevant not only from a scientific, but also a pedagogical perspective. So far, there is a glaring lack of interdisciplinary collaboration between Empirical Educational Scientists, system designers, pedagogues, Computational Linguists, and Second Language Acquisition researchers. Furthermore, little research has been conducted in this area outside of strictly constrained lab studies or studies with small or homogenous samples.

In order to address this gap, in this thesis we explore Learning Analytics in the context of Intelligent Computer-Assisted Language Learning in large-scale, ecologically valid contexts.

We start by laying the foundation by describing the three main research fields this dissertation builds upon: Second Language Acquisition, Tutoring Systems, and Learning Analytics and Educational Data Mining. In the second part of the thesis, we establish an empirical basis for subsequent analyses and applications. We describe the FeedBook system and its feedback mechanisms in detail, before we turn to the FeedBook study and the data collected in it. The third part combines the two previous parts by showcasing both a range of Learning Analytics applications, targeting the needs of different user groups, and data analyses that shed light on the relation between learning process features and learning outcomes.

With respect to the Learning Analytics functions, we show how an open learner model can fulfill the needs of individual students, how information can be aggregated on the school class level and presented to fulfill the requirements of teachers, and how data about the entire learner population can be made accessible in a useful way for material designers. The findings demonstrate that raw feedback counts are not useful in predicting learning gains. Rather, answers submitted correctly to the teacher are indicative. We investigate this learning product and split it up into learning process variables. Previous knowledge manifested in answers submitted correct at first try, and uptake based on specific

feedback both lead to correctly submitted answers, and are significant predictors of learning gains. In contrast, uptake based on blocked (i.e. only binary feedback) is not indicative of learning gains. Furthermore, we show an effect of time-on-task for the control group. Taken together, the results indicate that not only the provision of specific feedback, but additionally attention to feedback and efficient strategies for processing predict learning gains.

The main contributions of this thesis arise from its unique place at the interdisciplinary crossroads of Empirical Educational Science, Computational Linguistics, Second Language Acquisition, and Learning Analytics. Our primary contributions are the development and description of a tutoring system with interactive real-time feedback, the implementation of diverse Learning Analytics tools using and visualizing the learning process data collected with this system for different educational stakeholders, and the advancement of statistical models empirically linking learning process features postulated by learning theories with learning outcomes from authentic, large-scale contexts.

Related Publications

Some aspects of the contents covered in this thesis also appear in the following publications and are cited accordingly in the running text. Furthermore, we provide an overview about the referenced articles from this list at the beginning of each chapter of the work.

1. **Björn Rudzewitz**, Ramon Ziai, Florian Nuxoll, Kordula De Kuthy, Detmar Meurers. Enhancing a Web-based Language Tutoring System with Learning Analytics. Joint Proceedings of the Workshops of the 12th International Conference on Educational Data Mining co-located with the 12th International Conference on Educational Data Mining (EDM 2019), Montréal, Canada, July 2019, CEUR Workshop Proceedings, Volume 2592, pages 1–7
2. **Björn Rudzewitz**, Ramon Ziai, Kordula De Kuthy and Detmar Meurers. Developing a web-based workbook for English supporting the interaction of students and teachers. Proceedings of the Joint 6th Workshop on NLP for Computer Assisted Language Learning and 2nd Workshop on NLP for Research on Language Acquisition at NoDaLiDa 2017. Linköping Electronic Conference Proceedings 134: 36–46.
3. **Björn Rudzewitz**, Ramon Ziai, Kordula De Kuthy, Verena Möller, Florian Nuxoll, and Detmar Meurers. Generating feedback for English foreign language exercises. In Proceedings of the 13th Workshop on Innovative Use of NLP for Building Educational Applications (BEA), pages 127–136. ACL, 2018
4. Ramon Ziai, **Björn Rudzewitz**, Kordula De Kuthy, Florian Nuxoll, and Detmar Meurers. Feedback Strategies for Form and Meaning in a Real-life Language Tutoring System. In Proceedings of the 7th Workshop on NLP for Computer Assisted Language Learning (NLP4CALL 2018) at SLTC, Stockholm, 7th November 2018 (No. 152, pp. 91-98). Linköping University Electronic Press
5. Detmar Meurers, Kordula De Kuthy, Florian Nuxoll, **Björn Rudzewitz**, and Ramon Ziai. Scaling up intervention studies to investigate real-life foreign language learning in school. Annual Review of Applied Linguistics, 39, 2019
6. Ramon Ziai, Florian Nuxoll, Kordula De Kuthy, **Björn Rudzewitz**, and Detmar Meurers. The impact of spelling correction and task context on short answer assessment for intelligent tutoring systems. In Proceedings of the 8th Workshop on NLP for Computer Assisted Language Learning, pages 93–99, Turku, Finland, September 2019. ACL.
7. Detmar Meurers, Kordula De Kuthy, Florian Nuxoll, **Björn Rudzewitz**, and Ramon Ziai. KI zur Lösung realer Schulherausforderungen: Interaktive und adaptive Materialien im Fach Englisch. Schulmanagement-Handbuch, 169:65–89, 2019.

8. Detmar Meurers, Kordula De Kuthy, Verena Möller, Florian Nuxoll, **Björn Rudzewitz**, and Ramon Ziai. Digitale Differenzierung benötigt Informationen zu Sprache, Aufgabe und Lerner. Zur Generierung von individuellem Feedback in einem interaktiven Arbeitsheft. *FLuL – Fremdsprachen Lehren und Lernen*, 47(2), 2018.

Acknowledgments

First of all, I would like to thank my two supervisors, Detmar Meurers and Andreas Lachner, for their support and valuable feedback on the thesis and PhD project.

This thesis would not have been possible without the continuous support of Detmar Meurers of me and my work over the last eight years. He recognized and fostered my abilities when neither I nor anyone else saw the potential. A big thank you for the incredible support and shaping who I am today, both in computational linguistics and beyond!

The FeedBook system emerged as a result of team work and would not have been possible without the contributions of everyone involved. I would like to thank Detmar Meurers, Kordula De Kuthy, Ramon Ziai, Florian Nuxoll, Verena Möller, and every single one of the research assistants for their hard work and contributions on the project!

Furthermore, I would like to thank Simón Ruiz, Benjamin Nagengast, and Cora Parrisius for their explanations and helpful support on the methodology for the statistical analyses in this thesis. Thank you to Elizabeth Bear for help on proof-reading the thesis.

A very special thank you to my good friend and colleague Xiaobin Chen. Since we met in 2016, he supported me in the department and subject matters. More importantly, he showed me that there is a world outside of computational linguistics and goals worth pursuing, like for example relationships. I am extremely grateful for his kindness and support.

I also want to highlight the gratitude I feel for Ramon Ziai, since he supported me from the very beginning and helped me during dark times in the PhD. Thank you!

A huge thank you also to Johannes Dellert. With him, teaching over the last three years was very rewarding and fun. I learned a lot of him in terms of how to explain and approach complex matters. Moreover, he gave me valuable advice for matters outside of the PhD when I was in need.

A big thank you also to Tobias Elßner and Marko Knab. Your input on a personal level were essential to keep me going in this thesis. Our legendary regular tables always gave me the backup I needed to go through hard times and grounded me when it was necessary. Thank you to both of you!

On a personal level, I would like to thank Aohan Yang. Without her love and kindness, I would not have been able to complete this thesis. Thank you for going through good and bad times with me, and giving my life the meaning I was looking for for so many years! And thank you to all the bears in bear land.

Contents

1	Introduction	19
1.1	Motivation	19
1.2	Thesis Overview	21
I	Background	23
2	Second Language Acquisition	25
2.1	The Role of Practice	26
2.2	Feedback	28
2.3	Reaction to Feedback: Uptake	33
3	Learning Analytics and Educational Data Mining	39
3.1	Definition of Fields	41
3.1.1	Learning Analytics	41
3.1.2	Educational Data Mining	42
3.1.3	Comparison	45
3.2	Historical Perspective	45
3.2.1	Romero et al. (2010) and Romero and Ventura (2010)	45
3.2.2	Siemens and Baker (2012)	47
3.2.3	Liñán and Pérez (2015)	48
3.2.4	Papamitsiou and Economides (2014)	49
3.2.5	Mangaroska and Giannakos (2018)	50
3.3	Domains of Application	51
3.4	Competence Modeling	53
3.4.1	Modeling of Task Difficulty	53
3.4.2	Modeling Competence in Learner Models	54
3.5	Challenges	57
4	Tutoring Systems	59
4.1	Architecture of Tutoring Systems	61
4.1.1	Domain Model	61
4.1.2	Student Model	63
4.1.3	Tutoring Model	66

4.1.4	User Interface	67
4.2	Feedback in Tutoring Systems	69
4.2.1	Analysis of Learner Production	69
4.2.2	Representation of Target Hypotheses	74
4.2.3	Presentation of Interpretation of Target Hypotheses	76
4.2.4	Recording of Interaction with Feedback in Learner and Task Models	81
4.3	Non-Language Tutoring Systems	83
4.3.1	Overview and Definition	83
4.3.2	ASSISTments	83
4.3.3	MathSpring/Wayang Outpost	85
4.3.4	Andes	87
4.4	Language Tutoring Systems	89
4.4.1	Overview and Definition	89
4.4.2	E-tutor	90
4.4.3	TAGARELA	90
4.4.4	iTutor	91
4.4.5	ROBO-SENSEI	91
4.4.6	CASTLE	93
4.4.7	ICICLE	94
4.4.8	View/Werti	95
4.5	Tutoring Systems in Authentic Contexts	96
 II Towards an Empirical Basis		99
5	FeedBook as an Instrument for Data Collection	101
5.1	Overview	101
5.2	Exercises in FeedBook	103
5.2.1	From Paper to Digital Version	103
5.2.2	Task Domain Model	104
5.2.3	Authoring Tool	107
5.2.4	Displaying Exercises to Students	107
5.2.5	Displaying Feedback	113
5.2.6	Teacher Grading Interface	114
5.2.7	Student Result View	117
5.3	System Environment around Exercises	120
5.3.1	Student Lobby	120
5.3.2	Teacher Lobby	123
5.3.3	Administrative System Functions	126
6	Interactive Real-Time Feedback in FeedBook	131
6.1	Orthography	132
6.1.1	Motivation	132
6.1.2	Error Categories	132
6.1.3	Using the Task Context for Feedback on Orthography	133

6.1.4	Validation Experiment	134
6.1.5	Comparison with Existing Approaches	135
6.2	Grammar	136
6.2.1	Development of Error Types	136
6.2.2	Basis for Feedback: Using the Task Context	137
6.2.3	Implementation of Transformation Rules	138
6.2.4	Combining Transformation Rules in a Configuration-based Transformation Algorithm	139
6.2.5	Computation of Template Specificity and Ranking of Candidates	142
6.2.6	Flexible Matching	143
6.2.7	Combining the Functionality	144
6.2.8	Comparison with Existing Approaches	145
6.3	Meaning	146
6.3.1	Computational Linguistic Analysis of Learner Answer	147
6.3.2	Alignment-based Matching of Target And Learner Answer	148
6.3.3	From Alignment Configuration to Feedback Label	149
6.3.4	Displaying Meaning Feedback	150
6.3.5	Incidental Focus on Form Feedback	152
6.3.6	Comparison with Existing Approaches	152
7	The FeedBook Study	157
7.1	Method	158
7.1.1	Participants	158
7.1.2	Materials	159
7.1.3	Procedure	168
7.2	Collected Data	170
7.2.1	System Log	170
7.2.2	Submissions	174
III	Learning Analytics in Practice	177
8	Modeling Individual Learners	179
8.1	Needs Analysis: Students	180
8.2	Diagnosing Competence	180
8.3	Open Learner Model	184
8.3.1	Language Area Competence Visualization	186
8.3.2	Target Construct Specific Error Frequencies	187
8.3.3	Longitudinal Modeling of Learning Paths	189
8.3.4	Adaptive Sequencing of Material	190
9	Modeling School Classes	195
9.1	Needs Analysis: Teachers	196
9.2	Item-Level Performance Analytics	197
9.3	Learner Progress Visualization	200

9.4	Learner Log Perspective	200
9.5	Task-Level Performance Analytics	203
10	Modeling the Learner Population	205
10.1	Needs Analysis: Material Designers	206
10.2	Needs Analysis: Educational Administrators	206
10.3	Interaction Performance Metrics Perspective	207
10.3.1	Effort in Relation to Success Perspective	207
10.3.2	Uptake Metrics Visualization	210
10.4	Learning Path Perspective	215
10.5	Answer Variability View	215
10.6	Data-driven Target Answer Extension View	219
10.7	Progress View per School Class	220
11	Learning Analytics for Second Language Acquisition	223
11.1	Method	224
11.1.1	Focus on Theme 2	225
11.1.2	Answer Comparisons beyond String Matches	226
11.1.3	Participants	226
11.1.4	Procedure	227
11.1.5	Feature Operationalization	228
11.2	Results	228
11.2.1	Feedback	228
11.2.2	From Learning Product to Learning Process	233
11.2.3	Uptake	233
11.2.4	Exploring the Relationship between Predictors Further	245
11.2.5	Correct at First Try	249
11.2.6	Time-on-Task	256
11.3	Discussion	258
11.4	Limitations and Avenues for Future Research	266
IV	Conclusion	269
12	Summary and Outlook	271
12.1	Discussion and Conclusions	271
12.2	Summary and Contributions	274
12.3	Outlook	276

List of Figures

4.1	Conceptual architecture of tutoring systems	61
4.2	Conceptual overview about feedback process in tutoring systems	70
5.1	Theme 5, task 5b5 in the FeedBook (top) and in the printed workbook (bottom) © Westermann Gruppe, integrated like in Rudzewitz et al. (2017)	105
5.2	Annotator view: authoring tool for task meta data	108
5.3	Annotator view: authoring tool for task fields	108
5.4	Student view: exercise display task type-independent information	109
5.5	Exercise view: short answer task	110
5.6	Exercise view: fill-in-the-blank task	111
5.7	Exercise view: true/false task	112
5.8	Exercise view: mapping task	112
5.9	Exercise view: survey/questionnaire	113
5.10	Exercise view: displaying feedback (stage 1)	114
5.11	Exercise view: displaying feedback (stage 2)	115
5.12	Teacher view: assigning an error annotation to a learner answer enhanced with the highlighting of the difference to the target answer. Taken from (Rudzewitz et al., 2017, page 6)	116
5.13	Exercise view: displaying the result of feedback by the teacher	119
5.14	Student view: large viewport layout	121
5.15	Student view: medium viewport layout	122
5.16	Student view: small viewport layout	122
5.17	Student view: displaying sections	123
5.18	Student view: displaying tasks	124
5.19	Teacher view: Start page/lobby	126
5.20	Teacher view: managing student accounts	127
5.21	Profile page	128
5.22	News feed with two messages: reminder (orange) and submission confirmation (yellow)	128
5.23	Login screen	129
6.1	Multi-layered hypotheses generation process, taken from Rudzewitz et al. (2018)	140

6.2	Example error template	142
6.3	Example error template, taken from Rudzewitz et al. (2018)	142
6.4	Student answer including multiple errors with feedback based on a partial hypothesis match, taken from Rudzewitz et al. (2018)	143
6.5	Feedback algorithm (simplified pseudo-code), taken from Rudzewitz et al. (2018)	144
6.6	Display of meaning feedback, with different stages of assistance: coarse information source, precise information source, positive feedback	151
6.7	Display of meaning feedback for listening comprehension task, with different stages of assistance: shorter audio clip with information source, display of transcription of audio text, negation detection	153
7.1	Pretest task part a: selection task implemented as true/false exercise	164
7.2	Pretest task part b: fill-in-the-blanks exercise with provided options	165
7.3	Pretest task part c: short-answer exercise with free input	166
7.4	C-Test exercise	167
7.5	Example screen of the student questionnaire	167
7.6	Experimental design with rotating groups (taken from (Meurers et al., 2019a, p. 13) and slightly adapted)	168
7.7	General system server log (user names blinded)	171
7.8	Exercise-specific interaction log (user names blinded)	172
8.1	Learner model top level	185
8.2	Learner model, taken from Rudzewitz et al. (2020)	188
8.3	Typical errors visualized for the language topic <i>simple present</i> , taken from Rudzewitz et al. (2020)	189
8.4	Development of competence over time (green: correct usage, red: incorrect usage, yellow: successful uptake), taken from Rudzewitz et al. (2020)	191
8.5	Proficiency-appropriate sequencing of material in learner model, taken from Rudzewitz et al. (2020)	192
9.1	Prompt-specific learner answer analysis, taken from (Rudzewitz et al., 2020)	199
9.2	Visualization of learner progress over core tasks	201
9.3	Log data view	202
9.4	Task Performance View	204
10.1	Interaction performance metrics, taken from Rudzewitz et al. (2020)	208
10.2	Interaction performance metrics for an exercise that shows problems	211
10.3	Visualization of uptake over time	216
10.4	Answer variability view: type/token visualization (part 1)	217
10.5	Answer variability view: aggregated log data visualization (part 2)	218

10.6	Interface for data-driven extension of target answer	219
10.7	Progress view per school class (rotated 90 degrees to accommodate for long format)	221
10.8	Progress view of tasks across school classes	222
11.1	Independent variable uptake after displayed feedback, normal z-standardized values (top) and log-transformed, z-standardized values (bottom)	235
11.2	Mediation analysis of uptake based on seen feedback and pretest score on the change score	240
11.3	Visualization of uptake after displayed feedback (x-axis), pretest score (y-axis), and change score (z-axis)	242
11.4	Mediation analysis of uptake based on seen feedback and number of attempted tasks on the change score	246
11.5	Visualization of uptake after displayed feedback (x-axis), number of attempted tasks (y-axis), and change score (z-axis)	247
11.6	Polynomial fit of pretest score in relation to change score	250
11.7	Mediation analysis of pretest score and number of items correctly answered at first try with respect to the change score	255
12.1	Student lobby in Didi	277
12.2	Teacher lobby page in Didi	278
12.3	Categorization task in Didi	279
12.4	Jumbled sentence task in Didi	279
12.5	Memory task in Didi	280
12.6	Underline task in Didi	280

List of Tables

4.1	Considerations in displaying feedback	79
5.1	Domain model: properties of tasks, sub tasks, and task fields . .	106
5.2	Error types in FeedBook (taken from (Rudzewitz et al., 2017, page 6))	116
5.3	NLP components used in auto-correction (taken from (Rudzewitz et al., 2017, page 6))	118
6.1	Spelling error feedback templates and messages.	133
6.2	NLP tasks and tools, taken from Rudzewitz et al. (2018)	138
6.3	Feedback generation rules ordered in layers	141
6.4	Meaning-based feedback pipeline, originally adapted from Rudzewitz et al. (2018), but extended with meaning feedback analysis components	147
7.1	Number of tasks distributed across themes and sections	160
7.2	Distribution of tasks per grammar topic per theme	161
7.3	Language topics covered in the pre-/post tests	162
7.4	Dates of the pre- and post tests per school class. Classes with incomplete data are marked in gray.	163
7.5	Overview about student questionnaire	166
7.6	Number of lines in logs filtered per theme	173
7.7	Number of LoggingTokens (recorded interactions) per theme . . .	173
7.8	Number of tasks submitted to teachers for each chapter and section	175
8.1	Uptake types recorded in the learner model in FeedBook	182
10.1	Uptake metrics presented to material designers, taken from Rudzewitz et al. (2020)	214
11.1	Answer types by category	226
11.2	Feedback features considered in regression analyses	229
11.3	Regression model: significant main effect of correct feedback not blocked and displayed for one item	230

11.4	Regression model: estimating simple effects of group and correct feedback computed for all items in a task	231
11.5	Regression models: interaction of pretest score with feedback requesting features	232
11.6	Uptake feature operationalization	234
11.7	Regression model: uptake based on unseen feedback not significant, uptake based on feedback significant	236
11.8	Regression model: no interaction between group and uptake features	237
11.9	Regression model: no direct interaction between uptake features and number of tasks attempted	238
11.10	Regression model: significant effect of group, marginally significant effect of pretest score on uptake based on blocked feedback	239
11.11	Regression model: significant effect of pretest score on uptake based on displayed feedback	240
11.12	Regression model: significant effect of uptake based on seen feedback	241
11.13	Causal mediation analysis results of uptake based on seen feedback (mediator) and pretest score (independent variable)	241
11.14	Regression model: effect of independent variable on the mediator	243
11.15	Regression model: indirect effect of mediator on dependent variable	244
11.16	Causal mediation analysis results of uptake based on 1.000 iterations	245
11.17	Polynomial regression model with uptake based on seen feedback	248
11.18	Generalized additive model with pretest, number of attempted tasks, and uptake based on seen feedback	249
11.19	Regression model: highly significant main effect of answers correct at first try	251
11.20	Regression model: no significant interaction effects of answers correct at first try with group assignment and number of attempted tasks, but with pretest score	252
11.21	Causal mediation analysis: significant total effect of pretest score	253
11.22	Causal mediation analysis: significant effect of pretest score on the mediator correct at first try	253
11.23	Causal mediation analysis: significant effect of pretest score and mediator correct at first try on gain score	254
11.24	Causal mediation analysis results of correct at first try based on 1.000 iterations	255
11.25	Time-on-task feature operationalization	256
11.26	Regression model: significant interaction between experimental group and time-on-task	260
11.27	Regression model: significant interaction between time-on-task and number of attempted tasks	261
11.28	Regression model: significant interaction between time-on-task and pretest score	262

11.29 Regression model: significant effect of pretest on time-on-task,
but no significant effect when predicting the change score 263

List of Abbreviations

e.g.	exempli gratia (for example)
et al.	et alii/aliae (and others)
cf.	confer (see)
i.e.	id est (that is/this means)
vs.	versus (against)
N/A	not available (missing data)
URL	Uniform Resource Locator (web address)
L1	first language (first language learned as a child)
L2	a language that is not an L1 for a learner
SLA	Second Language Acquisition
CALL	Computer-Assisted Language Learning
ICALL	Intelligent Computer-Assisted Language Learning
EDM	Educational Data Mining
LA	Learning Analytics
SFB	Sonderforschungsbereich (Collaborative research center)
DFG	Deutsche Forschungsgemeinschaft (German Research Foundation)
NLP	Natural Language Processing
n.a.	nicht angegeben (not available)

Chapter 1

Introduction

1.1 Motivation

“[I]t is essential that future learning analytics developments and innovations draw on, and advance educational research and practice.”

— Gašević et al. (2015, p. 3)

In their seminal work, Gašević et al. (2015) reflect on lessons learned from the first years of the field of Learning Analytics based on influential studies. On the basis of these insights, the authors outline directions in which the field needs to move in the future.

Let’s take a step back to start with and understand the above quotation. Learning Analytics is “the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environments in which it occurs” (Siemens and Baker, 2012, p. 1) Learning Analytics can thus be conceptualized as data-driven methods applied to the educational context that serve the purpose of understanding learning and improving the conditions under which learning happens.

This approach thus needs data. *Where does the data come from?* In general, Learning Analytics focuses on data from educational contexts. This can be data available also in traditional learning and teaching settings, such as classroom participation, course completion, or even homework accuracy data (Romero and Ventura, 2007). However, it can also be data not available in paper-based or traditional settings, such as interaction data or learning process data: when a learner submits an answer on paper, the teacher only has access to the last version. If learning process data is not made available, the learning process remains a black box, with potentially many similar submission, but no insights or diagnostic value from the interactions (Mangaroska and Giannakos, 2018). In the context of this thesis, we are primarily interested in making learning process data distilled from interactions meaningfully available to different end users.

What are learners interacting with? In our case learners are using an intelli-

gent tutoring system for learners of English in 7th grade, the FeedBook (Rudzewitz et al., 2017). What sets it apart from submissions on paper and from non-intelligent tutoring systems is the provision of specific feedback (Rudzewitz et al., 2018). Tutoring systems serve the purpose of modeling and teaching parts of a specific learning domain, such as algebraic operations, or, in our case, language structures and their usage (Lynch et al., 2006), as specified in the Bildungsplan (curriculum points defined by the state) for 7th grade English.

Since the system provides feedback and adapts its behavior towards the productions of learners (Chatti et al., 2013), we can not only observe learning products but also the learning process of individual learners (Hattie and Timperley, 2007). Concretely, tutoring systems generate interaction logs that record how end users, and especially learners, are working with system (Sclater et al., 2016).

In order to get insights from interaction data, it is necessary to define features that quantify certain aspects of the interaction (Patchan et al., 2016). *How to get from interactions to learning process or product features?* Learning Analytics researchers engineer features by defining abstractions over the raw data that quantify certain events logged as entries in the interaction log. *How to decide which features to engineer?* First of all, what can be analyzed depends on the richness of the original log. Information that has not been logged is very difficult to reconstruct. Ideally, the selection of features and their operationalization is theory-driven. Previous research on (a) learning in general and (b) the domain and (c) learning in this specific domain should guide which features are extracted and used in various ways, such as visualizations or statistical models. Based on the insights from these processes, ideally the results are fed back to the theory and are integrated. This allows for a circle in which theory guides practical work, and results from practical work guide the theory. This is what the quote mentioned in the beginning really implies and what constitutes the imperative of this thesis.

In the context of this work, we explore how to approach educational data from different perspectives. We investigate how to make the process of learning – and different factors contributing to it – tangible and thereby contribute to an understanding of it. Concretely, we explore how to provide Learning Analytics end user applications that address the real needs of different user groups (students, teachers, material designers, and administrators). Furthermore, we conduct statistical tests with the goal of empirically and rigorously testing which factors can predict learning gains. In essence, we are interested in contributing insights to the question of what contributes to learning a language. We are interested in finding patterns of behavior that are indicative for successful learning. Knowing about factors that contribute to it allows for (a) humans to make decisions based on visualizations of relevant factors or metrics (b) machines to make decisions by estimating models of users, adapting their behavior, and predicting the next steps of these users.

Analyses of this type are especially relevant in the context of tutoring systems. While we live in a world of an ever increasing amount of digitalization across all elements of our society – including education –, surprisingly little re-

search has been conducted on the role and effectiveness of tutoring systems, let alone on language tutoring systems. On top of that, most research on tutoring systems has been conducted in controlled lab settings or with adult learners. The idea pursued in our work is to conduct research where it matters: in ecologically valid contexts with a diverse group of learners. Or, as Gašević et al. (2015, p. 5) put it, “the true test for learning analytics is demonstrating a longer term impact on student learning and teaching practice.”

1.2 Thesis Overview

This thesis is structured in three main parts, with each part building on the previous part(s).

In Part I, we provide the theoretical background and literature reviews on Second Language Acquisition (Chapter 2), Learning Analytics and Educational Data Mining (Chapter 3), and tutoring systems (Chapter 4).

The content of Part II is to establish an empirical basis for Part Part III. In Chapter 5, we describe the FeedBook tutoring system as an instrument for collecting learning process data, enabled via the feedback mechanisms described in Chapter 6. In Chapter 7, we specifically describe the year long study in which we used the FeedBook system in a range of different schools.

The third part is concerned with making sense of and allowing different user groups to make sense of the data collected. We start by exploring Learning Analytics tools for individual learners in Chapter 8, with a focus on an open learner model. Then we zoom out to a group of learners by describing Learning Analytics tools for teachers that allow them to get insights into the learning of their school class, described in Chapter 9. In Chapter 10, we take another step back and present Learning Analytics tools that use the data of the entire learner population and meet the demands of material designers and educational administrators. Chapter 11 concludes the third part by reporting the results of exploratory data analyses that identify learning process features indicative of learning gains.

As the ultimate step, we summarize the results, draw conclusions, and provide an outlook in Chapter 12 .

Part I

Background

Chapter 2

Second Language Acquisition

“Second language acquisition (SLA, for short), is the scholarly field of inquiry that investigates the human capacity to learn languages other than the first [...] once the first language or languages have been acquired.” (Ortega, 2009, p.1–2).

Second Language Acquisition is distinguished from First Language Acquisition (FLA). Following this distinction, it is possible that a learner has several second languages: every language learned in addition to the first language(s) is referenced as a second language. While the lines can be fuzzy between SLA and research on bilingualism, SLA research typically focuses on later stages of acquisition towards the end of childhood, adolescence or adulthood.

The field of study of SLA is highly interdisciplinary, including general linguistics (the domain to be acquired), psychology (conducting research on the processes underlying learning), and pedagogy (the method with which the contents are taught). Depending on the perspective, the field also incorporates aspects of statistics and other fields. The goal of SLA is to gain an understanding of and insights into how humans learn languages other than the first language(s); it combines both quantitative (e.g. feedback interaction models) and qualitative methods (e.g. error analysis).

Adopting this definition, we provide an overview about core concepts from the field of SLA relevant to this thesis. In section 2.1, we discuss the role of practice. Since practice seems to be fundamental for language acquisition, and for learning in a more general sense, we discuss factors that contribute to the nature and effect of practice. During practice, learners have the chance to try out different forms, potentially revealing a misconception, error or mistake. In order to draw a learner’s attention to a specific form and to guide them towards a correct or targeted form, an interaction partner (such as a teacher) can provide feedback to the learner. We discuss the role of and parameters involved in the the concept of feedback in section 2.2. In an ideal scenario, learners benefit from

feedback and are capable of integrating the feedback to remedy the problems addressed by it. This reaction is operationalized in the concept of uptake is the reaction of a learner to corrective feedback, revealing if and how learners react to the support of an interlocutor. In section 2.3, we discuss both the concept of uptake and factors influencing the uptake process.

Disclaimer: This chapter and its parts have not been published before.

2.1 The Role of Practice

Practice is key to learning. Under a cognitive-interactionist perspective to Second Language Acquisition, practice is an essential factor for learning. The success of practice depends on both internal factors (such as motivation, learning goals, and attention to forms) and external factors (such as the learning material received or the learning environment) (cf. Ortega (2009)).

Receiving only input is not sufficient. In order to learn, it is necessary to practice and produce output, since only then feedback can be provided on the learner production, and misconceptions can be addressed. Swain (1985, 2000) stated this in the context of the *Pushed Output Hypothesis*. This theory predicts that learning is the product of interaction partner pushing a learner to produce a better answer.

There are a range of different theoretical approaches towards explaining the effect of practice, such as the acculturation model (putting high emphasis on social factors for SLA, e.g. Schumann (1990)), or universal grammar models (assuming an innate, underlying universal grammar, e.g. White (1987); duPlessis et al. (1987)). For this thesis we adopt a *cognitivist perspective* in the form of the *Skill Acquisition Theory* (DeKeyser, 2003, 2007) combined with a *socio-cultural focus* assuming a *Zone of Proximal Development* (e.g. Vygotsky (1978); Lantoff and Appel (1994)). The Skill Acquisition Theory assumes that declarative knowledge is automatized via practice and becomes procedural knowledge. During practice, feedback is necessary to draw the attention of learners to specific aspects of their production (cf. section 2.2). The input provided to learners needs to be interpretable, 'comprehensible' input (Krashen, 1985), since otherwise learners are not able to relate parts of the input to their previous knowledge and interpret it. Via interaction with a competent partner, such as a teacher or a tutoring system, tasks that are not too easy and not too hard can be completed in a practice phase. The tasks need to be at an appropriate difficulty level, challenging the learner without frustrating them and allowing them to succeed with passable effort.

In a recent special issue on the role of practice in SLA, Suzuki et al. (2019) identified factors influencing the effect of practice. In the following paragraph, we will summarize the main points of this work illustrated with examples and references from related work. The nature of practice depends on the tasks learners are working on, from closed, form-focused exercises to open, communication-oriented tasks. Closely related to this dimension is the modality of input (e.g. selection vs. free production tasks) and stimuli (e.g. reading vs. listening texts).

Timing of practice and the temporal distribution of practice sessions is a relevant factor to consider (see also Trautwein (2007)). Closely related to timing is the order in which practice tasks are presented, categorized as blocking versus interleaving (tasks for different categories of constructs presented at the same time). An important factor related to the role of practice are aptitude-treatment interactions. Individual differences such as working memory or previous knowledge relative to the requirements of the current task need to be considered (cf. also DeKeyser (2012); Ruiz Hernández (2018); Suzuki et al. (2020)). In order to evaluate the effects of practice, different approaches can be pursued. One dimension to consider whether the focus is on learning products (e.g. test or exam results) or learning processes (e.g. interaction logs and reaction to corrective feedback). A related concept is the distinction between explicit, declarative and implicit, procedural knowledge. With respect to measuring competence, different approaches are possible, such as a delayed post test design or tasks requiring transfer to new contexts. Furthermore, external factors such as learning conditions (e.g. practice in school vs. homework), the cultural context, and personal/emotional factors play a role.

According to Shute (2007), there are different goal orientations with respect to practice in the context of learning and SLA. The key is that "for a learner to remain motivated and engaged depends on a close match between a learner's goals and his or her expectations that these goals can be met" (Shute, 2007, p. 12). This work distinguishes between different learner goal types: avoidance types (with the goal of avoiding negative consequences for not performing a task), learning orientation types (with the goal to increase competence), and performance orientation types (with the goal to demonstrate performance). A learning orientation is often considered favorable, since it is compatible with the idea that errors are important for learning.

Research in psychology categorizes behavior for reaching a goal into extrinsic and intrinsic motivation (Sansone and Harackiewicz, 2000). Extrinsic motivation is driven by external rewards or punishments. For example in school, an extrinsic motivation can be grades. Intrinsic motivation, in contrast, relates to rewards that are not related to external rewards or punishments, for example reading a book for pleasure. The pedagogical value of factors of extrinsic motivation has been discussed controversially and with contradictory findings (Deci et al., 2001). A negative effect of extrinsic motivation described in the literature is that it can decrease or supersede intrinsic motivation with learners optimizing their behavior towards reaching external goals only. On the other hand, extrinsic motivation has been shown to increase self-efficacy and intrinsic motivation. According to the *Cognitive Evaluation Theory* (Ryan, 1982), extrinsic motivation increases intrinsic motivation as long as the external rewards increase the perceived competence and self-determination.

Due to the complexity of factors involved in practice, it is highly challenging to capture all relevant factors in studies looking at the role of practice. However, attempts have been made to shed light on the influence of a combination of factors. For example, Jurik et al. (2014) investigated individual differences, motivation, and student profiles with respect to practice. The study investigated

the interaction between teacher behavior (questions and feedback), gender, and self-reported learning activity in terms of depth of processing. The authors showed that more autonomous learners were more successful than students who needed a lot of guidance. In their study, they adopted the learner types/profiles described by Seidel (2006) to classify learners into strong students, uninterested students, underestimating students, overestimating students, and struggling students. For example, underestimating students have high previous knowledge, but a low self-concept. Learner types are thus used as a way of combining several factors of learners into one label or category. Furthermore, the authors showed that there is a difference in the effect of deep reasoning questions versus tasks requiring shallow reasoning skills. In a study with 1,335 learners from 79 ninth grade physics classes from Germany and Switzerland, they collected pretests and self-reports after recorded lesson. Regression analyses showed that deep reasoning tasks led to higher cognitive activity and more intrinsic motivation, especially for strong learner types (in contrast to struggling learners). Different types of feedback were not taken into account, but hypothesized to have an effect. Strong and overestimating learners were more active in the classroom than other learner types. Underestimating learners benefited from praise and deep reasoning tasks since it encouraged them and had a positive impact on their self-concept. They call for teaching approaches that adapt to different learner types and scaffold them according to their profile's needs.

To sum up, the range of studies discussed different aspects of the role of practice for learning. We saw that there are aspects on different levels that play a role: on the level of learners (such as motivation, goal orientation, previous knowledge), on the level of tasks learners are working on (such as openness or reasoning types), but also on the level of the environment (such as the cultural context).

2.2 Feedback

Feedback is “information provided by an agent [...] regarding aspects of one’s performance or understanding” (Hattie and Timperley, 2007, p. 81). The core idea of feedback is that a competent entity is assisting a learner in doing exercises by pointing out differences between the learner production and an expected target answer. While it is beyond the scope of this thesis to discuss the entire research on feedback, we will discuss a range of studies and models in the following that are relevant for this work because they shed light on factors that interact with feedback.

Ortega (2009), in her overview about feedback, highlights the importance of social components in Second Language Acquisition research with respect to feedback. Since feedback involves an interaction between the learner and the learning partner (e.g. teacher, parent, peer, tutoring system), there clearly is a social component involved between the interaction partners.

The concept of social interaction is central to the Zone of Proximal Development (ZPD). As stated by Vygotsky (1978, p. 131), “The zone of proximal

development [...] is the distance between the [child's] actual developmental level as determined by independent problem solving and the level of potential development as determined by independent problem solving under adult guidance or in collaboration with more capable peers." This means that for exercises that a learner attempts, (s)he is capable of reaching more when working together with some form of partner compared to working alone. Or in other words: in this case, one plus one equals more than two, since the (social) interaction between the learning partners allows them to reach a goal jointly that they could not have reached independently and helps to reduce the distance between the exercise level and the learners' proficiency level. While this concept was formulated before the age of computers, the computer in the form of a tutoring system can serve as a competent partner, with the explicit knowledge encoded in its domain model and the interaction with the learner guided by its pedagogy module. The Vygostkyan strand of research in Second Language Acquisition research has seen continued popularity and been extended in various ways (cf. e.g. Lantoff and Appel (1994)).

An influential feedback model in Second Language Acquisition research was proposed by Hattie and Timperley (2007). They propose a theoretical model combined with findings distilled from literature reviews. In the following, we will summarize their feedback model.

The authors describe that feedback leads to interaction with two components: interaction with the feedback (message), and interaction with original task. Originally, learners are working on a task and interacting with it (e.g. by reading the instructions), but once feedback is displayed, learners partly shift their cognitive resources to the feedback and interact with the feedback.

Feedback leads to cognitive processes such as confirmation, restructuring of knowledge, or pursuit of different task strategies. Since an important function of feedback is to guide learners towards a correct solution, feedback is most effective in cases in which it targets very specific erroneous aspects the learner can relate to, but not very effective if the student does not understand the task at all. The difference is explained by the fact that in the former case, the feedback is related to the specific task, whereas in the case of a complete lack of knowledge, feedback can not correct, but would have to add information to the learner's knowledge representation. Hattie and Timperley (2007) explain that a meta-analysis of feedback studies showed that feedback is effective if the feedback is on a task or/and task strategies. For this reason, positive feedback was often shown to not be effective since praise mostly does not contain task-specific information, i.e. explanations of why a specific answer was correct. Furthermore, positive feedback can also decrease intrinsic motivation, or replace it with extrinsic motivation. Feedback, in order to be effective, needs to incorporate the goal towards it leads, information about the progress towards the goal, and next steps for learners to take. According to the model, there are four levels/aspects on which feedback has an effect, which we will describe in the following.

Feedback can change the approach by a learner to a certain *task* (as is the case for corrective feedback). As stated above, feedback on erroneous parts is more effective than feedback on a lack of information. If feedback is on a task,

more explicit feedback works better. But in any case, feedback is only effective if the learners actively pay attention to the feedback messages.

Another level on which feedback can work is the *working or learning process* in general. Hattie and Timperley (2007) distinguish shallow learning from deep learning strategies. The form of feedback can have different effects on these task strategies, interacting with the proficiency of learners.

In addition, feedback has effects on *self-regulation*, that is the regulation of behavior for accomplishing a task. The idea here is to get from external feedback to internal feedback loops in which learners evaluate how to proceed next and on other tasks. Feedback has been shown to influence the self-monitoring, with the highest effects in cases where the learner expects a correct answer, but received feedback that shows it is not correct.

On an *affective or personal level*, feedback on the person of the learner can lead to higher effort on tasks, but also be detrimental, e.g. in classroom situations in which praise is distributed only very selectively by a teacher. Praise was shown to only be effective if it contains task-related information, but it is not effective if it is praise in general not about the task strategies a learner implemented. Negative feedback can have the effect that the goals of learners are adapted and raised, whereas for positive feedback, learners are not forced to adjust their goals.

Regarding the timing of feedback, Hattie and Timperley (2007) state that feedback at the task level is effective when it is delayed, since then and especially for more difficult tasks, learners have enough time to think about the task and how their solution corresponds to it, so that later they can mentally put the feedback in relation to the task's requirements. In contrast, feedback at the process level is most effective when it is given immediately, i.e. formative feedback that supports the process of accomplishing certain components of tasks.

The article concludes with the observation that very little research on feedback in actual classrooms has been conducted, even though it has been shown to be highly effective. This shows that there is a gap for the provision of timely formative feedback that tutoring systems can partly fill, justifying the implementation and testing of tutoring systems with immediate feedback.

Fyfe and Rittle-Johnson (2016) showed that the effectiveness of feedback interacts with prior knowledge. Feedback can have negative effects for learners with prior knowledge because some learners with more previous knowledge did not accept feedback due to their confidence in the subject matter and performed later worse than learners with less previous knowledge who accepted the feedback. The authors explain this finding with the possibility of cognitive overload because learners with previous knowledge need to deal with feedback that contains redundant information and thus would be better off without feedback about aspects they already know about – but given that feedback is displayed, they still have to process it. Feedback was, in their study, found to be more helpful for learners without or with only little previous knowledge. Another finding was that immediate feedback can lead to learners giving up their previously established feedback strategies for problem solving and adopting new strategies more tailored towards the feedback mechanism, i.e. they

optimize their behavior towards the feedback mechanism.

Golke et al. (2015) investigated how motivation and attention interact with feedback processing and transfer. According to Golke et al. (2015, page 2), feedback “does not function automatically. It instead demands an active learner who is motivated to process the feedback information.” In essence, when a feedback message just pops up, but learners are not motivated to engage with it, the feedback will not have a positive effect. They further explain that effectiveness of feedback also depends on the content and formulation of the feedback message (with differences in the degree of explicitness of the feedback), the timing with respect to its presentation, and the presentation type of the feedback. The authors found that immediate corrective feedback was more effective than delayed feedback for incorrect responses, and specific feedback was more effective than unspecific, generic feedback. In their study, time-on-task did not differ for feedback vs. non-feedback condition. However, Golke et al. (2015) found a difference in feedback processing with respect to whether a person or a computer delivered it. The presence of a human increased the chances that the feedback was processed successfully, partly because of the social component involved when an actual person is present in the same room. In their study, computer-mediated feedback was not as effective as feedback delivered by a person. One reason for this was that learners had no motivation because the task to which feedback was provided was de-contextualized and did not lead to an ultimate goal such as higher final grades – in this case, only a social component could motivate them to engage with feedback.

Patchan et al. (2016) investigated different factors influencing the effectiveness of feedback. They build on a model by Nelson and Schunn (2009), in which feedback is seen as a concept relating to a combination of cognitive and affective features. Cognitive features such as the specificity of the feedback, highlighting of errors, or the explanation of errors contribute to supporting the understanding of students. In contrast, affective factors such as praise or mitigation of the error serve to increase the agreement of the learner to the feedback. Both types of factors, relating to understanding and to agreement, need to be present for feedback to be effective for learners. In their experiment, they looked at 432 university students in an introductory psychology class. They reported unclear findings with respect to the effectiveness of feedback of different specificity and detrimental effects on providing only a target answer or mitigating errors. Instead, they found positive effects for feedback localizing an error. Another interesting finding was that showing more errors reduced the likelihood of feedback shown later to be implemented, probably because learners were already cognitively exhausted after reading the first feedback messages. They found a positive effect for praise, but only for high amounts of praise.

Cerezo (2016) investigated the granularity of the explicitness of feedback. They went through the relevant literature and identified as a trend that more specific feedback leads to higher learning outcomes. But: binary feedback has also been shown to be effective. With their participants, consisting of intermediate learners of Spanish, the authors triangulated different data sources for learner modeling that goes beyond competence measures and includes quali-

tative aspects such as think-aloud protocols, but also click-streams on items. They found out that additional practice can lead to fatigue and boredom and diminish the effects of more practice and feedback. An important finding from their study is that deeper processing and more attention leads to higher learning gains. Feedback can not necessarily be processed and does not lead to higher learning gains if learners are exhausted from practice. Only when learners are actively engaging with a task and feedback, this can lead to deeper processing (cf. also Leow (2019)), and subsequently a positive effect on learning outcomes.

With respect to time-on-task, the relation between time spent and learning gains is still unclear. While classical approaches like Cooper and Pantle (1967) or Carroll (1963) propose a total-time law in psychology, claiming a direct relation between time spent studying and amount of learning, more recent work views this more critically. As mentioned above, Cerezo (2016) showed that additional practice – and thus more time-on-task spent – can lead to fatigue and diminish the effects of more practice and feedback. Going from lab-based scenarios to authentic, real-life contexts, the relation between time spent and learning gains is poorly understood (cf. e.g. the research on homework by Trautwein and Köller (2003)).

Lyster and Ranta (1997) distinguish a range of feedback types discussed in the following. An *explicit correction* consists of explicitly providing a correct form. A *recast* is an implicitly provided correct form via embedded in a reformulated version of the learner answer. In *clarification requests*, teachers/learning interaction partners indicate that a part of the student answer is ill-formed and needs to be repaired. *Meta-linguistic feedback* is a type of feedback in which a teacher provides information about a deviating form or meaning of a student answer without providing a correct or target answer. Finally, *elicitation* can be achieved via strategic pauses with gaps for student to fill in the gap, i.e. provide a context in which the student knows she needs to react to the pause with another attempt at the correct answer.

As Heift (2010a) explains, corrective feedback is on a scale between two extremes. In the least specific case, a system provides unspecific default feedback that does not address a specific error. In contrast, the most extreme case is to display the target answer. On both ends of the extremes, no specific error feedback is displayed.

Panova and Lyster (2002) explain that there are different kinds of feedback with respect to the target and timing. Formative feedback is provided on the process of working on an exercise, whereas summative feedback sums up the performance on a task after having submitted it. Formative feedback is defined as "information communicated to the learner that is intended to modify his or her thinking or behavior for the purpose of improving learning" (Shute, 2007, p.1). The noticing hypothesis (Schmidt, 1990) states that learners can only notice a gap/difference between their answer and a target answer if it is salient enough. Corrective feedback can help point out gaps in the process of working on a task by comparing the learner response to a target representation and drawing attention to a specific part of the answer.

With respect to the timing of feedback, there are conflicting findings (Shute

(2007), which serves at the basis for the following arguments). There is support for delayed feedback. There is some evidence that learners forget errors, i.e. errors will not be persisted in memory. Furthermore, studies have shown that transfer of knowledge to new tasks was higher with delayed feedback. On the other hand, there is support for the positive effects of immediate feedback. It has been demonstrated that immediate feedback is better for training procedural skills, for short-term outcomes, and for more complex tasks.

According to Shute (2007), immediate feedback is often shown to be effective in field studies in contrast to delayed feedback, which is often shown to be effective in lab studies. Furthermore, low-achieving students or students with low learning orientation benefit more from immediate feedback, whereas high-achieving students benefit more from delayed feedback.

Shute (2007) attempted to identify explanations for conflicting findings from the literature on the effect of formative feedback. Factors identified in previous research as having negative effects were critical feedback, controlling feedback, grades or scores comparing students to peers together with lack of transparency or vagueness, and feedback from external source that disrupts learning. The functions of feedback, according to Shute (2007), can be categorized into directive feedback (specific feedback referring to an error in the learner answer and providing information about what needs to be fixed) and in facilitative feedback (providing hints or examples, students need to come up with the error types themselves). As cognitive mechanisms for feedback the author lists the reduction of uncertainty for learners on their production, the reduction of cognitive load since part of it is outsourced to the system providing feedback, the correction of strategies for solving tasks, and the correction of misconceptions. Related to that, the author argues that unspecific feedback is challenging for students since it requires learners to invest cognitive load in order to detect the problem. Another aspect discussed is that for low achieving students, comparisons with others (the norm) is frustrating and decreases their motivation. In contrast, "self-referenced feedback" (Shute, 2007, p. 20) is beneficial since it shows them that via practice they can get better results than before.

2.3 Reaction to Feedback: Uptake

"Comprehensible input alone is not sufficient for successful L2 learning; comprehensible output is also required" (Lyster and Ranta, 1997, p. 41). In their seminal work, Lyster and Ranta (1997) explain that practice generates student output, in which errors occur, and of which teachers correct some. In order for learners to benefit from the error correction process, there is a necessity for teachers to highlight parts of the learner input or output to draw attention to specific (in the case of language learning linguistic) features. In the following, we will summarize their work and classical model.

Uptake can be observed as part of an interaction sequence, each of which is initiated by a learner error. As a reaction to corrective feedback, teachers can continue without feedback (referred to as topic continuation) or give cor-

rective feedback. The decision as to when feedback is provided is of a pedagogical nature. Students then can react to corrective feedback (uptake) or not (topic continuation). This student reaction can still contain errors (referred to as needs-repair) or not; in the case of needs-repair, the entire process can be repeated and start over again.

Lyster and Ranta (1997, p. 49) define uptake as a "student's utterance that immediately follows the teacher's feedback and that constitutes a reaction in some way to the teacher's intention to draw attention to some aspect of the student's initial utterance." Uptake is here defined in a general sense as a reaction to corrective feedback. The authors distinguish uptake with repair, in which a specific error targeted by corrective feedback repaired from uptake that still needs repair. In the case of uptake with repair, there often is a reinforcement by the teacher after the successful correction. In FeedBook, this is approximated via the green check marks following a correct answer.

Types of reactions that still require repair are acknowledgment (saying that the teacher said what the student wanted to say), making the same error, making a different error, making another error while the original error is not corrected or repeated, providing an off-target response (circumventing feedback and original error, no further errors), a hesitation, or a partial repair (a correction of sub part of original error). These types of uptake are reflected and serve as the theoretical basis for the FeedBook learner model (chapter 8). They acknowledge, but do not consider, cases with self-initiated repair. In FeedBook thus only the intervention group is compatible with this model since there the uptake is not self-initiated (in contrast to the control group where the feedback was partly blocked).

In a quantitative analysis, Lyster and Ranta (1997) showed that 62% of learner errors were followed by corrective feedback. 55% of the corrective feedback moves were followed by uptake (27% with repair, with 28% needs-repair). Meta-linguistic feedback formed only 8% of the teacher feedback, even though meta-linguistic feedback was more effective in generating student uptake with repair (45%) than other types of corrective feedback.

Ellis et al. (2001) extended the work by Lyster and Ranta (1997) and defined the following types of uptake: acknowledgment (learners acknowledge feedback), uptake with repair (error targeted in feedback corrected), uptake that still needs repair (form targeted in feedback still not used correctly), recognition (learner recognizes the error), application (learner attempts to apply the feedback and use the form), and needs application (learners do not apply the targeted form).

Smith (2005) takes a different perspective with respect to the effects of uptake. From a terminological standpoint, the author points out that the definition of uptake changed over time. First, it represented a recall of learned concepts. Then it was used for an immediate reaction to corrective feedback. More recently, it was also used to describe an optional learner move as a reaction to implicit or explicit highlighting of a linguistic feature in the learner production. In a study over the duration of 6 weeks, the author analyzed the data of 24 university level participants of English as a Second Language. Specifically, the author looked at synchronous computer-mediated communication, i.e. chats that learners collaboratively used to solve tasks. This setting was chosen in

contrast to other studies since it was hypothesized to involve factors beneficial to SLA learners in that they would have the chance to anonymously try out new forms produce output in a written manner allowing for a direct focus on specific forms. The author found out that (I)CALL and SLA theories are not always compatible in that they found no clear link between uptake and acquisition. They broke down sequences of chat interactions into episodes and identified 66 uptake observation of different types. Among no uptake, this included a passive recognition (learners indicated that they understood the feedback) and an active application (learners used forms under discussion actively). The results showed that post test scores can not be predicted by uptake observations, that the acquisition is not predicted by uptake, and that the complexity not is correlated with uptake.

Panova and Lyster (2002) define uptake as an immediate learner response to corrective feedback referring back to the original answer. They distinguish uptake with repair from uptake that still needs repair. Of uptake that needs repair, they distinguish an acknowledgment of the error, learners committing the same error, learners committing a different error, learners going off-target, a hesitation, or partial repair. In cases where teachers continue with different topics or the student does not refer back to the original answer (e.g. when a student did not notice corrective feedback), no uptake can be observed. Based on the literature, they expected that specific types of corrective feedback (clarification requests, elicitations, meta-linguistic feedback, and repetition) correlate more with uptake than explicit correction or recasts. In their study with 25 adult learners of English, they observed corrective feedback to 48% of the student answers and uptake to 47% of the corrective feedback instances. 34% of the uptake instances were with repair. The main finding of Panova and Lyster (2002) was that corrective feedback instances that give away target answers do not lead to uptake.

Heift (2004) investigated learner uptake in the context of computer-assisted language learning. Based on the theoretical foundation of the interaction hypothesis, which states that learners negotiate linguistic forms with a partner (such as a teacher or a system), and based on the noticing hypothesis, which states that input needs to be comprehensible for learners to notice features in the input, they investigated the distribution of corrective feedback types in relation to learner uptake and learner characteristics (gender, native language, previous exposure, and computer experience). In the context of e-tutor (cf. section 4.4.2), different types of corrective feedback with different specificity are considered: meta-linguistic, meta-linguistic with highlighting, and repetition with highlighting. The study involved 177 students over the course of one semester who used e-tutor for 15 weeks covering four chapters with 50 exercises per chapter, plus pretests, post tests, and a questionnaire. Uptake here is defined as student responses to corrective feedback. For this study, students were able to resubmit an answer and get feedback again, ask for the correct answer, or skip to the next exercise. Heift (2004) conducted a quantitative analysis of the collected data by looking at the number of interactions per interaction type. The types considered were answers correct at the first attempt, correct but not

on the first attempt, submission of answers with errors, skipping an exercise, and peeking at a correct answer. The main findings of the study were that that there was no effect of gender or proficiency on the revision patterns, that more explicit feedback is desired by learners, and that more explicit feedback correlates with more error corrections and thus leads to more successful uptake. Open questions that could not be answered were the effect of error types on uptake, the effect of motivation and learner type/strategy on uptake, and the effect of additional linked material on uptake.

As a follow-up study, Heift (2010a) investigated the effect of uptake in e-tutor more longitudinally. Over the course of twelve months spanning three consecutive German courses over three semesters, ten learners of German as a second language (seven female, three male, average 20 years) were observed. The study altered the types of feedback provided to learners, with each participant initially randomly assigned to a test group and swapped after each of the 13 chapters. The chapters included a total of 633 distinct exercises. The first test group received meta-linguistic explanations including highlighting, a reference to the error category, and specific feedback. The second group received only meta-linguistic clues in the form of highlighting of error and a mention of the error category. The feedback categories included grammar and spelling. The grading consisted of looking at whether exercises have been completed (not whether they have been completed successfully or correctly). Based on previous research, the author hypothesized that more specific or explicit feedback is more helpful to students. In order to allow for comparisons across learners, uptake was normalized by the number of exercises completed (student uptake divided by exercises completed). The study investigates the effect of learner uptake with respect to grammar or spelling errors, the effect of learner uptake in beginner, advanced beginner, intermediate course, and the effect of learner uptake across all courses (longitudinal modeling). The uptake categories considered in this study included successful uptake (no errors in learner answer), unsuccessful uptake (learner answer still contains errors after learner reaction to feedback), and no uptake (skipping exercise or looking up target answer). The data consisted of 6,111 exercises/prompts completed by students, with around 200 exercises/prompts completed per course and student. 2,974 exercises/prompts were not completed correctly at first try. A total of 12,137 submissions (attempts) for these exercises were collected, with an average of 1.99 submissions per exercise. Of these, 4,943 submissions were correct at first try, 7,194 submissions with errors. In 16.2% of the cases for both experimental groups, no uptake was found. A t-test revealed that the group with feedback with explanations showed significantly more uptake and more re-submissions/attempts. Furthermore, Heift (2010a) found out that more specific feedback was more helpful for grammar than spelling errors, since for grammar errors there is more ambiguity, and specific feedback narrows the room of options more down. Consequently, there was a higher proportion of uptake for grammar errors compared to spelling errors, with more uptake for more specific feedback. The study revealed a larger difference for higher or more advanced courses, again especially for specific feedback.

In a discussion section, the author points out that uptake is not equal to

mastery. A limitation discussed is that the study did not take into account specific types of grammar errors, but only the frequency of uptake independent of the specific error. The FeedBook data can provide this type of information.

Chapter 3

Learning Analytics and Educational Data Mining

Learning Analytics is “the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environments in which it occurs” (Siemens and Baker, 2012, p. 1). Or, as Mangaroska and Giannakos (2018, p. 3) sustain, the task of Learning Analytics is the “development of measures that can increase the understanding into the learning processes and interpret those measures to inform existing theories for the purpose of developing actionable initiatives in teaching practices and design of learning environments.”

While Learning Analytics can be used to analyze data from traditional face-to-face teaching settings, it is primarily concerned and relevant in the context of web-based learning, since in contrast to traditional classrooms, learners and instructors are separated spatially and often also temporally. This disconnect deprives instructors from a direct monitoring of their students during the learning process, generating a need for tools that provide instructors with learning process related information (Romero and Ventura, 2007).

The data that are used as input to these Learning Analytics applications typically consist of client-server log data. Every entry in the log consists of a time stamp, a user id, and an action, potentially with context such as the exercise a learner is currently working on (Romero et al., 2014). Log entries can be generated every time the learner interacts with a system (Baker and Yacef, 2009). This typically results in large quantities of data points that need to be pre-processed, evaluated, and made interpretable for different purposes of different stakeholders (learners, instructors, material designers, researchers). Pre-processing data in the educational area is especially challenging when compared to other domains since on the one hand, large quantities of data of different granularity are available, but on the other hand, the data is often incomplete, since not all learners always do all activities they are asked to do (Romero et al., 2014), especially in ecologically valid settings (Siemens, 2013), which, however,

provide the most valuable data.

Typical applications of Learning Analytics as discussed by Papamitsiou and Economides (2014), are student behavior modeling (e.g. learning strategies, response times to questions), the prediction of future performance of learners (e.g. using grades, participation metrics), fostering an increase in progress reflection and awareness of learning progress (e.g. student-at-risk detection), the prediction of dropout and retention in courses, the improvement of feedback and student assistance services, and the recommendation of resources (sequencing of materials). Romero and Ventura (2007) mention further applications in the form of the identification of misleading feedback, the data-driven evaluation of the difficulty of learning material, and an automatic sequencing of material based on this. In the context of intelligent tutoring systems, a further common application of Learning Analytics is the implementation of open learner models to increase the meta-cognition of learners regarding their progress (Bull and Kay, 2006). In order to arrive at such analyses and applications, it is necessary to define features that provide quantified abstractions over the raw log data, i.e. feature engineering.

Based on a literature review, Gašević et al. (2015, p.3) argue that “learning analytics needs to ground data collection, measurement, analysis, reporting and interpretation processes within the existing research on learning”. This means that the analyses and features should make use of findings related to learning processes as a guidance and theoretical basis. With a theoretical framework making use of pedagogy and cognitive science, it is possible to put theoretical findings relating to learning processes to the test. With human learning being a complex process, an important question to ask is whether there are simpler features that can be extracted from the log files commonly used in Learning Analytics that (partly) correlate with or allow to predict the learning process. The selection and engineering of specific features depends on the relevant task at hand and is therefore domain-specific and differs depending on which application should be implemented. For example, an application designed to provide sequencing of mathematical exercises uses different features than an open learner model in a system offering a dashboard to students visualizing the complexity of essays in a foreign language.

In this chapter, we approach the research field of Learning Analytics and Educational Data Mining from different perspectives. We start by looking at how these research field emerged historically (section 3.2), and which definition is adopted today in the research community and this dissertation (section 3.1). In the next section (3.3), we inspect in which domains Learning Analytics and Educational Data Mining are used, and for what purposes. Following this discussion, we dive deeper into one application especially relevant for this dissertation: the modeling of competence (3.4), subdividing this topic into the modeling of task difficulty (3.4.1) and the competence of learners in learner models (3.4.2). We conclude this chapter by discussing open challenges of Learning Analytics and Educational Data Mining in section 3.5.

Disclaimer: This chapter and its parts have not been published before.

3.1 Definition of Fields

In the following sections, we will establish definitions of the areas of Learning Analytics (3.1.1) and Educational Data Mining (3.1.2). We present and compare different high-level definitions established in the literature, zoom into different components of them, and then elaborate certain aspects of these definitions in order to give an understandable and self-contained definition for each field. In section 3.1.3, we provide a comparison based on these definitions.

3.1.1 Learning Analytics

Learning Analytics is “the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environments in which it occurs”. This definition by Siemens and Baker (2012, p. 1) was adopted by the *Society for Learning Analytics Research*¹ and forms the official definition for this research field.

Various authors have rephrased or slightly extended this definition, with each of the definitions discussed in the following putting emphasis on an extension of a specific aspect of the original definition.

According to Mangaroska and Giannakos (2018, p. 3), Learning Analytics constitutes the “development of measures that can increase the understanding into the learning processes and interpret those measures to inform existing theories for the purpose of developing actionable initiatives in teaching practices and design of learning environments.” This definition stresses the importance of a strong theoretical basis for Learning Analytics. Measures that are implemented and presented to humans for decision making should not be arbitrary, but instead have a justification from a theoretical point of view.

An alternative definition is provided by Sclater et al. (2016, p. 4). They define Learning Analytics as “the measurement, collection, analysis and reporting of data about the progress of learners and the contexts in which learning takes place”. In this definition, the focus lies on the *progress* of learners. Since every interaction with a digital learning system, e.g. client-server interaction or client-side events triggered by the learner, generates log traces that can be stored and used as input to Learning Analytics, measures that characterize the process of learning, as opposed to summative “snapshot” measures, can be developed. When a learner submits an exam paper, the teacher can only see the final product of the task completion process. In contrast, when a learner works on a digital exercise, this provides the opportunity to save intermediate versions leading up to the final submitted version. This begs the question of ethical practices regarding data store, usage, processing, and presentation. System designers have to find solutions in agreement with ethical committee guidelines, local legislation, and best practices in system design.

Ferguson (2012), in their definition, makes additional assumptions about Learning Analytics to delineate it from other educational research. The author

¹<https://www.solaresearch.org/about/what-is-learning-analytics/>, last accessed 2020-01-28

presupposes that Learning Analytics work is based on data in a quantity that makes manual analysis not very feasible. Closely related to research in Educational Data Mining, this leads to approaches where bigger amounts of data are combined by methods such as clustering, aggregation, or visualization in dashboards in order to allow humans to find patterns that are otherwise not obvious because they are distributed over many individual log entries.

Gašević et al. (2015, p. 5) state that the “true test for Learning Analytics is demonstrating a longer term impact on student learning and teaching practices.” While the authors emphasize that Learning Analytics needs to be informed by and using insights from previous theoretical research on psychology/learning processes, they formulate the ultimate goal as a positive change in authentic environments. This implies that decisions beneficial for the learning process have been taken, and these decisions were enabled by Learning Analytics.

Or, as Chatti et al. (2013, p. 6) put it, the goal of Learning Analytics is to enable taking actions. Siemens and Baker (2012) share this perspective by stating that the goal of Learning Analytics is that machines inform humans to support their decision making or action taking process in the sector of education.

The concrete actions in this case consist of the following cases: analysis and monitoring of students, prediction of success or risk of failure in a course, assessment of students, adaptivity in learning software, fostering of meta-cognitive learning processes and self-reflection, provision of feedback to learners, fostering empirical testing of pedagogical interventions and teaching strategies. In section 3.3, we go into more detail for each of these actions.

Different authors disagree where to draw the line between Learning Analytics and related disciplines. Chatti et al. (2013) view the process of data collection and pre-processing as a part of Learning Analytics. In contrast, Ferguson (2012) make the assumption that Learning Analytics always works with existing data in digital form, thus excluding the stage of data collection from this research field.

Combined in essence, Learning Analytics is the activity and research of building tools with the purpose of enabling to take action in the domain of education. Or, in a nutshell, Learning Analytics is concerned with going from educational data to “actionable intelligence” (Arnold and Pistilli, 2012, p.1).

3.1.2 Educational Data Mining

Educational Data Mining (EDM) is a research field that constitutes of the application of data mining methods to educational data.

Data Mining refers to the task of “the discovery of interesting, unexpected or valuable structures in large data sets” (Hand, 2007, p. 1). The assumption underlying this research is that ‘interesting’ information is contained implicitly in large amounts of data, and that data mining techniques allow to extract this information (Bissantz and Hagedorn, 2009; Raghavan, 2005). In order to extract initially hidden information, two categories of approaches are pursued: model building (supervised machine learning) and pattern discovery (unsupervised machine learning) (cf. e.g. Petersohn (2005)).

The definition of what counts as interesting depends on both the domain and the purpose of the analysis. For example, data mining in the domain of bio-informatics has an interest in discriminating genome sequences (Frank et al., 2004), whereas a typical goal for data mining in the business sector is the identification of customer groups and purchase patterns (Bose and Mahapatra, 2001). In the educational domain, data mining is concerned with the understanding and modeling of learning processes and the contexts of learning, thus pursuing the same ultimate goal as Learning Analytics (cf. section 3.1.1).

A criterion applicable to all subdomains of data mining is the availability and usage of bigger amounts of data. While this formulation is somewhat imprecise in that the perception of what constitutes a 'big' amount of data varies according to the development and availability of improved technology (Fan and Bifet, 2013), the notion of data mining presupposes electronically available data in a quantity that would render the time needed to have it analyzed manually by humans unreasonably disproportionate in comparison to an analysis by computers.

Having approached Data Mining from a general high-level perspective, let us turn to *Educational Data Mining* specifically. According to Siemens and Baker (2012, p. 1), Educational Data Mining is a research field "concerned with developing methods for exploring the unique types of data that come from educational settings, and using those methods to better understand students, and the settings which they learn in". They sustain that the goal of Educational Data Mining is to use automated, data-driven methods to isolate certain characteristics of learners, tasks, or the environment in order to gain insights into the learning process or adapt the behavior of learning/tutoring systems.

According to Romero et al. (2010), Educational Data Mining emerged as a follow-up to data mining methods applied to business and biology. In order to delineate Educational Data Mining from its predecessor in Business Analytics, they compare the two fields in certain aspects. In business data mining, the ultimate goal is to maximize profit, whereas in Educational Data Mining, the goal is to maximize the learning. They point out that the data employed in Educational Data Mining is much richer, since specific analyses of input and feedback are employed in contrast to server logs or purchase lists in the commercial sector. This allows for a more fine-grained modeling of the involved humans in this domain. While this claim was most likely true at the time of writing, in the meantime the commercial data mining market has changed drastically: with a very broad adoption of smartphones and sales via apps on smartphones, users typically provide apps with extensive rights to access all the data stored on the device such as telephone or message logs, data from other apps, GPS location, etc. (Grace et al., 2012). So in today's situation, commercial apps often have more fine-grained information about their users since they can not only log all the data inside the app, but join it together with data from other apps or the device. With respect to the data mining methodology and techniques employed, the two sub fields have a significant overlap, but Educational Data Mining requires special emphasis on modeling the dependence between individuals since learners often learn in groups (typically school classes) and therefore

influence each other. A straightforward approach to model these dependencies is multilevel modeling (e.g. Gelman (2006)).

In order to perform any analysis in data mining, an essential and one of the biggest steps is to conduct pre-processing of the raw source data. In their *Survey on Pre-Processing Educational Data*, Romero et al. (2014) summarize approaches of pre-processing of Educational Data Mining.

Compositionally, Educational Data Mining contains the term “mining”. In mining, the goal is to extract valuable materials from a planet’s upper layers of the crust. In order to find the valuable veins of rare materials, tools guided by humans are necessary to remove big amounts of stone first. Once potential veins are discovered, they have to be extracted and further processed by specialized tools, whereas the surrounding stones are discarded. In Educational Data Mining, the removal of stones corresponds to pre-processing of raw data, and the processing of veins of potentially valuable materials corresponds to educational data mining for valuable learning information. Two types of pre-processing have to be distinguished: (1) filtering of information relevant for a specific task and (2) transforming the data into a shape that a specific analysis tool requires. For case (1), imagine a scenario in which access patterns of exercises by learners should be analyzed. In this case, and when a raw server log serves as a basis, log statements about teachers correcting exercises of learners can most likely be filtered out of the log. In case (2), sticking to our example, the task identifiers in the log statements may be in a string format, whereas the analysis tool may require the identifiers in a numeric format.

In detail, the following steps are necessary for processing educational data in the context of Educational Data Mining (Romero et al., 2014). Educational Data Mining researchers need to gather data, potentially from different sources. The different data sets then need to be integrated into a common data set, or prepared into a representation that can easily be merged or joined. Especially for different data sets, the data will contain aspects not relevant to the current analysis, requiring a round of data cleaning. After this, the next step is the identification of data by user and/or session. This is a necessary step for the attribute selection phase. The next step they describe consists of filtering the data in the sense that data only on a specific level is selected, for example data for a specific task or for sessions in a certain time frame. The last pre-processing step is data transformation, where the data is normalized and features summarizing other features are defined, e.g. normalization or outlier removal.

Once these steps are completed, Educational Data Mining research then can use the features as input to analyses to (a) understand learning and the learning environments better and (b) improve learning and learning environments better. The process is not one-directional, but instead circular: the results of analyses can be used to adjust systems that generate data that can be analyzed and so on.

Educational Data Mining as an interdisciplinary research field builds mainly upon educational research, pedagogy, social sciences, computer science, human-computer interaction, data visualization, machine learning, artificial intelligence, information retrieval, recommender systems, Academic Analytics (a form

of Business Analytics applied to education) (e.g. Chatti et al. (2013)). For different stakeholders, a different emphasis is put on the importance of different related neighboring areas, depending on the needs for the specific target groups.

As Siemens and Baker (2012) summarize, Educational Data Mining is about humans informing machines to make decisions in a learning context and to automatically build models about learning.

3.1.3 Comparison

In conclusion, the two research areas, while historically treated as separate fields, have converged to a point where the similarities strongly outweigh the differences. Rather than different research fields, we argue that they are different strands of the same research field, with slight differences in the emphasis of their application.

In both areas, the shared goal is to help understand and improve learning (Liñán and Pérez, 2015). What differs between the fields is mainly the methodology: while Learning Analytics puts higher emphasis on building tools that inform humans about the learning process, in Educational Data Mining research often more importance is put on using human intelligence to inform automated models (Siemens and Baker, 2012). In a nutshell, Learning Analytics fosters human inspection of learning data, and Educational Data Mining fosters automatic modeling of learning data.

3.2 Historical Perspective

In section 3.1, we discuss current and established definitions commonly adopted for and in the context of Learning Analytics and Educational Data Mining. While these definitions serve to provide a concise and precise definition of the fields, it can be insightful to look at concrete work performed in this domain to gain insights into what type of work has been conducted in practice. This is relevant in that the general definitions abstract away from concrete work. With the goal of approaching the fields from a more practical perspective, we in the following provide a discussion of past work based on systematic literature reviews. With a comprehensive literature review of all the many hundreds of papers published in Learning Analytics and Educational Data Mining in the last three decades infeasible for the scope of this thesis, we instead take existing literature reviews as a basis. In the following, we summarize several literature reviews with different perspectives on the historical development of the field. Each of the reviews offers a different perspective and approach to the historical development of the field.

3.2.1 Romero et al. (2010) and Romero and Ventura (2010)

Romero et al. (2010) and Romero and Ventura (2010) provide a very comprehensive literature review covering the time period 1993 to 2010. Educational

Data Mining emerged as a follow-up to data mining methods applied to business and biology.

Data mining methodology was, in the public or business sector, first applied successfully for Business Analytics. Only after having been applied there successfully and showing the positive impact of this methodology, these methods were transferred to the educational sector, thereby forming the first wave of work in Educational Data Mining.

While a tiny amount (less than five papers per year) were published before the turn of the millennium, the first substantial wave of papers in data mining (in general) was published in the year 2000 (cf. also (Raghavan, 2005)). In the context of Educational Data Mining, more than five papers per year were published only since 2001, more than ten papers per year since 2002, with from then on an approximately exponential growth (with the study examining 306 publications until 2009). In reviewing these articles, the authors identified as trends eleven categories of applications in Educational Data Mining, which we summarize, order and further combine into six groups of categories in the following.

The first and most important trend identified is to provide statistics and visualizations for showing information and supporting different stakeholders (learners, teachers, researchers, course developers, organizations, administrators) in their decision making processes. The most common methodology for these analyses consists of the use of statistical tools and dashboards. Features that served as input to this type of analysis consisted of navigation patterns, material access patterns, types of end user devices, demographic information, temporal development of the usage of system features, search behavior or information, forum activities, online postings, time employment, user behavior in different time frames, exercise completion, and attendance patterns. For cases where learners actively produced contents themselves, as in forums, threads, and exercises, certain analyses provided both a content analysis of the text and an indication of learner answers and errors.

As a second trend, Romero and Ventura (2010) identified functions that provide feedback to teachers. In contrast to the descriptive analyses presented as part of the first trend, these analyses go more into the direction of diagnostic and even prescriptive tools. The most common methodology for this area consists of association rule mining, clustering, and correlation analysis. Note that these are methods that put data in relation to other data and features of interest, and not merely describe it. Work in this domain concretely consisted of providing tools for the analysis of the relation between resource usage and learning outcomes, detecting patterns or associations in learner behavior or profile features, providing feedback to material designers or course authors regarding the effectiveness of tasks or curricula, mining of temporal patterns including enrollment, grouping of items in terms of diagnostic value or redundancy, alerts for problematic learning patterns, and analysis of chats or threads with respect to course outcomes.

The third trend is a summary of several areas considered separate by the authors of the mentioned literature reviews and put here under the label provid-

ing recommendations to learners. An important strand of research deals with providing feedback or recommendations for students. Similarly to the second trend, a common methodology is association rule mining, but also other techniques such as unsupervised machine learning, sequential pattern mining, or decision trees. A type of information relevant to learners is the identification of sequences of behavior associated with successful or unsuccessful outcomes. This can provide learners with diagnostic value about their behavior. On the prescriptive side, a common approach is to automatically recommend activities or resources to learners, recommend shortcuts on materials, recommending reading materials based on vocabulary profiles, automatic recommendations for learning partners, and course recommendations.

The prediction of student performance, and learner modeling in general, can be considered as a fourth major area in Educational Data Mining. The goal is to model different, primarily performance or knowledge oriented, characteristics of learners. As a methodology, supervised machine learning is used for the prediction of learner performance, Bayesian networks primarily for knowledge modeling, association rule mining, Markov decision models, sequential pattern mining, or social network analysis for grouping of learners. Areas of interest constitute the use of information about motivation, affective features, learning styles, system behavior, indicators of off-topic or cheating behavior, prediction of dropout or students-at-risk, recommendation of remedial classes to learners, analysis of learning styles, and the automatic formation of learner groups.

Closely related to student modeling is the task of knowledge mapping. The task here is to create a model of the learning domain contents. By using mostly association rules and text mining tools, work in this area consisted of the automatic organization of domain knowledge in concept maps or concept networks, inferring relationships between domain concepts by mining the content of discussion boards, or the creation of ontologies spanning an entire sub domain.

Constructing course ware and planning courses is the sixth and final area. By using mostly unsupervised machine learning and association rules, features under consideration are the promotion of re-use of materials, the planning of courses based on past experiences, the admission and counseling of students, and the data-driven identification of gaps in courses and development of new programs based on it.

3.2.2 Siemens and Baker (2012)

While relatively concise, Siemens and Baker (2012) is considered a seminal article since it provides a very widely cited historical development of the fields of Learning Analytics and Educational Data Mining and how they compare.

While sustaining that the historical roots of Learning Analytics and Educational Data Mining date back to 1995, they define the starting point of the professional development of the field as the conduction of the first dedicated international workshops. For Educational Data Mining, the first dedicated workshop was held in 2005, and the first international conference on in 2008. The field of Learning Analytics started a bit later in 2010 with the first dedicated

conference. In 2011, a professional association for Educational Data Mining, the 'International Educational Data Mining Society' was founded, the same year as the foundation of the 'Society for Learning Analytics Research'.

Siemens and Baker (2012) describe that Learning Analytics, from its beginnings, put higher value on incorporating psychological or psychometric measures, while the focus of Educational Data Mining is more on the technical adaptation of methods and their usage on data from an educational context. While they outline some trends (summarized in the following), they emphasize that there is a considerable overlap between the Learning Analytics and Educational Data Mining communities with no clear boundaries between the fields, rather trends about directions of conducted past research.

As referenced in section 3.1, Learning Analytics puts more importance on human discovery of interesting patterns in data than Educational Data Mining, which promotes more automated discovery and adaptation methods. Learning Analytics typically emphasizes a more holistic understanding, whereas in Educational Data Mining researchers often try to break complex systems down to individual components. According to Siemens and Baker (2012, p. 2, table 1), the methodology of Learning Analytics historically consisted primarily of social network or sentiment analysis, sentiment analysis, learner modeling, and domain modeling techniques. In Educational Data Mining, the methodology consisted more of classification, clustering, Bayesian modeling, relationship mining, and visualization.

3.2.3 Liñán and Pérez (2015)

Liñán and Pérez (2015) describe the historical development of the two fields by reviewing articles from 1995 to 2015. They identify how they developed and how they compare.

Before computers were available, education mostly consisted of traditional face-to-face instruction. This allowed teachers to directly monitor students by observing them in the classroom. The data that could be used for analytics consisted of data from examinations (mostly on paper, potentially with multiple data points over time), attendance data, or qualitative insights from teachers (cf. also Romero and Ventura (2007)). Then, in the next period where computers or at least television devices were available, but no world wide web, instruction could in principle be made available partly in a digital manner. An example given by the authors is television-based instruction (either recorded or broadcasted in real time). A problem with this method though, in comparison to traditional or more modern ways of instruction, was that this only allowed for passive consumption, leading to either no or very delayed interaction (e.g. by submitting answers via actual physical mail).

The introduction of the world wide web led to a new era of distance learning. The key difference is that the world wide web allowed a more timely provision of interaction, for example via e-mail (reducing the time for feedback in comparison to television-based methods from weeks to hours or minutes), or even real-time interactivity in online courses that provide exercises for learners with immediate

feedback, real-time chats, course forums, etc.

In essence, with the rise and adoption of the web, the distance in distance education became much smaller and much quicker traversable. This interaction, enabled via the world wide, web generated data traces that could be stored and analyzed. This and the fact that hardware and memory became and are still becoming cheaper led to the introduction, development and fostering of Learning Analytics and Educational Data Mining, with the professional societies *International Educational Data Mining Society* and *Society for Learning Analytics Research* founded in 2011.

The first considerable amount of papers from Educational Data Mining stems from 2005/2006, the first substantial amount of papers from Learning Analytics from 2010/2011. They analyze that in the first phase of the fields from 1995 to 2005 (with fewer papers), mostly relationship mining methods were popular, but from 2008 on prediction methods, exploratory data analysis, and clustering gained momentum.

3.2.4 Papamitsiou and Economides (2014)

In their literature review of Learning Analytics and Educational Data Mining spanning the time period 2008 to 2013, Papamitsiou and Economides (2014) identify research objectives, methods, and significant results in both Learning Analytics and Educational Data Mining from this time frame, summarized in the following.

For learning settings, most research is based on interaction with learning management systems. Other forms of computer-based education, including tutoring systems and massive open online courses, have been used, but play a less important role in comparison. The data was engineered into and combined with features such as login frequencies, number of chat messages, number of questions to instructor, response times, time on task, which resources accessed, previous grades, final grades, profiles, preferences in systems and observations of affect. The actual methodology of data analyses consisted of mostly classification (supervised machine learning), followed by clustering (unsupervised machine learning), and discovery with models. Based on a review of 209 articles, Papamitsiou and Economides (2014) grouped the work according to six research objectives summarized in the following.

The first research objective is to model the behavior of students. This includes, in the most basic approach, an observation of sequences of learner actions. On a higher level, and by adding a layer of interpretation, researchers have attempted to model meta-cognitive states, the response time for exercise items, and the identification of different learning strategies.

Moving from modeling the behavior to modeling the performance of learners and predicting it, the second research trend becomes clear. Researchers have posed the question of which factors can contribute predicting the performance of individual learners. Features under consideration discussed in this literature review are demographic factors, grades, portfolios, multi modal skills, participation, enrollment, engagement, and affective states.

As third research objective consists of the promotion of an increase in (self-) reflection and awareness. One way of pursuing this research objective has been to build tools that allow instructors to identify students that are disconnected from the rest, i.e. show a progress that is different from the normal progress. This can be seen as a student-at-risk identification task. An alternative approach under this research objective is to provide tools for students such as open learner models, dashboards, or textual output. The goal is to inform students about their progress and performance indicators. This can include the possibility for them to compare their performance to peers. An example for textual output is the visualization of output from association rule mining algorithms regarding the relation of usage of learning materials with respect to learning outcomes.

Very similar to the second and third research objective is the fourth objective, namely the prediction of student dropout and retention on a course or institute level. Research under this objective has made use of engagement patterns that are clustered and thereby allow to identify groups of learners with different characteristics. Researchers have also included factors regarding student satisfaction such as the perceived usefulness or the perceived efficiency of the training offered in a course.

A more distinct area of research and objective is the improvement of feedback and assessment services. Research has explored different types of feedback and the quantitative effects on performance and qualitative perceptions both for students and teachers. Furthermore, it has been explored how tests can be made more flexible moving away from static tests with a one-size-fits-all approach towards adaptive testing. Other studies looked at the effect of automatically suggesting tasks enabled by an adaptive sequencing algorithm.

This is highly related to the final research objective concerned with automatic recommendations. Though similar to the fifth area, the focus here is more on how to determine exercise-specific parameters related to the complexity of tasks. This includes modeling the performance of learners and putting it into relation with exercise parameters, but also modeling exercise parameters independent of the learner performance and using it for clustering exercises. System designers have combined different approaches in recommender system models that use affective states, competence models, and exercise and item models.

3.2.5 Mangaroska and Giannakos (2018)

Mangaroska and Giannakos (2018) review literature from the time frame 2007 to 2017, with a special emphasis on Learning Analytics.

They identified that the sample population in most studies primarily consisted of undergraduate students and teachers.

The most popular learning system providing data for analyses was Moodle, but they also state that in the studies under consideration there is an approximately equal distribution of studies with purely digital, hybrid, or face-to-face classrooms. Most studies based their analyses on logs of specific systems or learning management systems, some on interviews or questionnaires.

From an analysis point of view, the analysis techniques used mostly are regression, clustering, and correlation analysis. Very similar to what Papamitsiou and Economides (2014) identified (cf. section 3.2.4), Mangaroska and Giannakos (2018) state the following research goals: evaluation of learning activities and environments, learner modeling, effectiveness of usage and impact of Learning Analytics tools for different stakeholders, predictive modeling, assessment, and adaptivity.

In their review, they not only identified trends, but also several areas where problems occurred. One example is that although many tools offer visualizations in dashboards to inform learners or students, (at the time of the publication of the respective studies) they were mostly available outside of the actual learning environments. They furthermore stress the importance of using time as an explicit factor in modeling learners, tasks, learner performance, and usage trends. The authors also lament a lack of theoretical grounding of the analyses.

3.3 Domains of Application

In Learning Analytics and Educational Data Mining research, there are different approaches with a different degree of complexity, but also a different degree of merit, as exemplified e.g. by the *Society for Learning Analytics Research*². The work, as classified by them, can be placed in in four categories. In the following, we will describe these categories and substantiate them with examples and actual systems.

The simplest category of work are **descriptive statistics**. This line of application is concerned with describing previously collected data. Its purpose is thus to describe and abstract over instances of past events. Typical applications in this domain are visualizations of learning activities (Duval, 2011), either for individual learners or tasks, or in dashboards that integrate multiple levels of data and present them in an intuitive way (Verbert et al., 2013). These tools often are implemented as dashboard visualizations that enable self-reflection and self-quantification (a detailed quantification on different levels and with different aspects, comparable to e.g. fitness apps). The information can be visualized with respect to different needs depending on the target user (Romero and Ventura, 2010).

Descriptive analyses can help discover problems related to learners or learning materials. This forms the basis for the next category of work in Learning Analytics/Educational Data Mining: **diagnostic analysis**. In this strand, the goal is to diagnose why a certain problem arose, or why a certain success condition was met. As for the first category, research in this category is conducted on past data. It however goes further by asking why something happened, and not only what happened. This typically involves putting measures of (learning) success or failure in relation to measures of process data. Since in educational environments the ultimate goal is to foster learning and acquire competence

²<https://www.solaresearch.org/about/what-is-learning-analytics/>, last accessed 2020-01-31

in an area, the diagnostic measures typically involve some sort of competence modeling associated with tasks and mapped for individual learners (Bull et al., 2015). Another application in this area put forward by Romero et al. (2010) is the inference or testing of domain structures based on learner responses (e.g. Junker et al. (2010)). Diagnostic analysis is typically integrated as part of learner models Kump et al. (2012) that can be visualized or used as input to cluster analyses (Romero and Ventura, 2013). Another application on this level is domain modeling, i.e. data-driven modeling of learning materials, the constructs, and the performance of learners on them (Sottolare et al., 2016).

Going one step further on the spectrum, **predictive statistics** allow to compute predictions about the future based on past data. Based on diagnostic analyses, these models predict how certain learners will develop or how certain tasks will be received, i.e. a future diagnosis based on the current state. This requires that the models that are trained on past data generalize and allow to make claims about future states, i.e. that there is enough information in the past data to make reliable predictions. This can be especially challenging in cases where there are non-linear developments. A typical application in this domain is the task of student-at-risk identification (Stevens and Pihl, 1982) or the prediction of learning outcomes and grades (Romero et al., 2010).

The most advanced application domain of Learning Analytics is the task of **prescriptive analysis**. Based on a predictive analyses, prescriptive analyses suggest interventions/actions that, given a predictive model, would affect the current prediction the model is making, i.e. it attempts to not only generalize based on past events to the future, but also to make changes so that the currently predicted state in the future is not reached in this form (Soltanpoor and Sellis, 2016).

This can have different consequences, depending on what optimality criteria are defined for the specific task at hand. For example, while a prescriptive analysis prescribing exercises for learners for an improvement of their domain competence as a consequence of diagnosed insufficient knowledge with the goal of better preparing a student for an exam (e.g. Linillos et al. (2006)) can in general be viewed as a beneficial intervention, other prescriptive approaches can invoke different perspectives. In an extreme case, an over-sensitive system might suggest to exclude a student from a course based on short-term performance metrics such as two consequent assignments not submitted. The actual reason for the student's (temporary) lack of display of performance might however be rooted in external factors not known to the system, such as the loss of a close family member and a permission by the teacher to not submit the assignments. If this information is not known by an administrator on a higher level, the administrator could decide to exclude the student because the prescriptive model makes this suggestion. This example shows that for prescriptive analyses, systems/models make recommendations, but it is important that humans stay in control of decision making and critically evaluate all predictions by the system.

Prescriptive analysis is commonly performed in the context of tutoring systems that sequence exercises based on the performance of learners (e.g. Linillos

et al. (2006)). Another application in the same vein is the recommendation of educational courses enabled by recommender systems (Zhang et al., 2014).

When talking about domains of application, a practical consideration that influences the possible analyses – or complicates them – is the location and representation of the involved data. As for example Chatti et al. (2013) discuss, there is a fundamental difference between instances of centralized learning systems such as for example Moodle, versus distributed learning environments on the other hand. In the former case, all data and actions are performed, logged and stored in one system (e.g. click streams, material accesses, tests results). For distributed systems or systems that incorporate different data sources, data from different contexts, devices, with different formats, potentially a different granularity of etc. pose challenges both for data exchange and merging of data.

3.4 Competence Modeling

The concept of competence is not well-defined and consequently viewed as a multi-faceted and fuzzy concept (e.g. Ashworth and Saxton (1990)). For the purpose of this thesis, we define competence as the mastery of a set of pre-defined skills based on the empirical evidence of interaction data.

In this part of the thesis, we primarily discuss the modeling of competence in learner models in section 3.4.2. In this section, we explain conceptually different approaches towards representing different skills in, and estimating the values for, learner models in tutoring systems.

However, we are aware that it is not possible to separate the modeling of competence of learners from the modeling of task difficulty. The reason is that learners can only demonstrate competence or lack of it when they are working on tasks. Provided that we assume that different tasks differ in their complexity, both between tasks and when related to the skills of (different groups of) learners, we take a quick glance at data-driven task difficulty modeling in section 3.4.1. We acknowledge that it is not entirely clear what contributes to the difficulty of tasks and that giving a comprehensive overview of cognitive models for task difficulty is not the focus and would clearly go beyond the scope of this thesis

3.4.1 Modeling of Task Difficulty

Task difficulty models are concerned with modeling how challenging exercises are for different learners. It allows to predict the difficulty of different tasks when compared to each other, as well as the difficulty of tasks for different groups of learners differing in their knowledge and skills.

While we are aware that there are more approaches for modeling task difficulty, e.g. the widely known Cognition Hypothesis (Robinson, 2007), we describe a widely used and well understood formalism for task difficulty analysis: Item Response Theory (IRT, e.g. von Davier and Lee (2019)). The core idea is that each item in a task has a specific difficulty level, with the probability for a

successful completion of an items related to (a) the item difficulty and (b) the skill level of test takers.

IRT models are latent class models. The models assume that unobserved, latent classes underlie exercise completion and determine or explain observed variables. The test takers are assumed to belong to different classes that perform differently for tasks, with each test taker belonging to one specific class. The number of classes is not determined beforehand, but emerges from the data.

Item-characteristic curves allow to plot the ability of learners against the probability of obtaining a correct response. The shape of the curve allows to determine how well the item discriminates between different types/classes of learners. Similarly, Wright maps allow to plot the skill of participants versus the item difficulty on the other side. IRT models assume a categorical outcome variable (item solved correctly or not), but continuous values of traits/skills/abilities to predict it.

The models are typically implemented as logarithmic models and are similar to linear or mixed regression models, with a so-called link function mapping the input (learner skills and task difficulty) to the output (probability of answering an item correctly).

There are different types of IRT models, most notably the distinction between parametric and non-parametric variants. In parametric variants, models assume a set of skills associated with persons and items, whereas non-parametric variants assume only one dimension. Explanatory IRT (Wilson et al., 2008; Hartig and Höhler, 2010), also called MIRT (multidimensional IRT) models tasks with different constructs/skills involved. It attempts to explain how specific item properties influence the probability of correct answers. The goal is to explain what causes certain items to elicit patterns of responses of certain groups of learners. Double explanatory models contain factors about both persons and items.

A widely used special case of IRT models is the Rasch model (e.g. Andersen (1973); Masters (1982); Bond et al. (2020)). It only assumes a single skill/component for all items and computes the difference between the intercept (person parameter) and the item difficulty scaled by a logarithmic function.

3.4.2 Modeling Competence in Learner Models

The idea of modeling competence in learner models is to store properties inferred by the system about a learner that serve as the basis for inspection by users or adaptation by the system (Slavuj et al., 2017).

A learner model is by definition a model of the interpretation of user actions that abstract over and approximate certain aspects of learner interactions. Learner models typically consist of a learner *profile* comprising stable traits (name, age, gender, etc.) on the one hand, and a learner *model* with dynamic (frequently updated) data on the other hand. The approaches described in this section are based on the work by Slavuj et al. (2017) and refer to the second source of information. The dynamic data comprises both *domain-unrelated* data such as the device used, learning style, age, but also *domain-specific* data such

as the learning path, learning goals, tasks, interests, or test results. Again, we are interested in approaches that are concerned with the second source of information.

Since the data is partly dynamic, we are interested in different adaptation processes of domain-specific data (Slavuj et al., 2017). There are long-term, stable traits that change and need to be adapted at most over many sessions, such as aptitude, learning style, or personal data. Mid-term data is stable over the duration of one learning session such as the device used, the location, or the time. Finally, there are short-term properties that are dynamic within one session such as activities, goals, or affective states. Competence models can include short-term properties (e.g. interaction with feedback mechanism), mid-term properties (e.g. timing of learning), and long-term properties (e.g. previous knowledge from certain courses).

Important to keep in mind in the discussion about different approaches for modeling the competence of learners is that there is not 'one' way of designing learner models (Bull and Kay, 2006). Instead, there are many different approaches that have different strengths and weaknesses and can often be combined with each other depending on the system needs. There are conceptually different learner models regarding explicitness. In approaches with observable nodes, values can be updated directly (e.g. overlay models), whereas in models with unobservable nodes, the values can not be updated directly (e.g. Bayesian models). In the following, we will discuss different approaches for modeling competence in learner models.

In their discussion and literature review of different types of competence models, Chrysaftadi and Virvou (2013) distinguish at least five approaches for learner model representations. In the following, we will use their categorization for a presentation of learner models augmented with examples from the literature.

In an **overlay approach**, the learner model is a simple overlay over the domain model. The domain model represents the concepts to be learned in a structured way. For each of those elements, there is a corresponding value in the learner model that indicates how well the learner has mastered this element. Typically, the values are numerical, but categories like for example "mastered" or "struggling" are possible as well. The goal of the system is to minimize the divergence between the learner model and the domain model. An important aspect for overlay models is that the complexity of the approach depends on the complexity of the domain and the estimation technique for values in the learner model. This means that even though the approach may seem simplistic at first glance, having a complex domain model and an elaborate technique to update the values, can make overlay models complex and powerful. A weakness with respect to overlay models is that often a simplistic way of representing concepts in the domain model is chosen that assumes independence of the concepts in the domain. While this provides simplicity, it often causes a weakness in terms of power of the derived learner models (cf. also von Davier and Lee (2019)).

An example for an overlay approach is described by Bull and Nghiem (2002). In their approach, they present an open learner model that allows to compare

the performance of one learner to the performance of peers in the same group (school class). For each of the concepts in an associated domain model, the system stores the number of attempts that were correct and incorrect related to this concept in the learner model. The data for these values comes from multiple choice tasks where each item is assigned to a domain concept. The frontend allows to display the data in two formats: in a tabular format and as a bar chart. In this specific system, the learner model is open to both students and their teachers.

The **perturbation approach** is an extension of the overlay model that in addition includes misconceptions. The misconceptions can be modeled/represented in different form, for example as a set of mal-rules/misconceptions specific to a student, target topic, or task. The FeedBook learner model (cf. chapter 8) is an instance of a perturbation model.

In a **stereotype approach**, the population of learners is divided into groups or clusters. Each group (or stereotype) represents common characteristics and serves as a category summarizing shared characteristics. Stereotype-based approaches implement both trigger and retraction conditions that determine the assignment to specific stereotypes. Stereotype-based systems solve one important problem that other approaches face: the cold start problem. Since in the beginning only very limited data is available, the question is how to estimate values for unobserved properties. Even with incomplete knowledge, stereotype-based approaches allow to assign learners to a group/stereotype and make use of information about the group for estimating unknown values. Typically, system designers hand-construct stereotypes before a system goes live. While the assignment of learners to groups is dynamic, the stereotypes themselves are typically static. Stereotypes can combine different information sources from the learner model or profile, both permanent traits and dynamically computed features emerging from interaction with a system.

An example of a stereotype-based approach is described by Jurik et al. (2014). In the context of physics education, they define five stereotypes: strong, uninterested, underestimating, overestimating, and struggling students. The stereotypes combine properties like cognitive activation and previous knowledge. They show that learners of different stereotypes react differently to questions of teachers that require deeper versus more shallow processing.

Another way of modeling competence in learner models is the **constraint-based approach**. The idea here is that the domain model is represented as constraints, and the learner model is represented as those constraints that learners have violated while working on an exercise. Suraweera and Mitrovic (2002) present a hybrid approach that combines a constraint-based model with an overlay approach. In the domain of teaching learners about relational data bases, the system implements short-term and a long-term component of the learner model. The short-term model is used to generate feedback for specific tasks, whereas the long-term model is an overlay approach that stores the frequency of applied constraints in the learner model.

In **Bayesian approaches**, knowledge is traced in a probabilistic manner and updated with every interaction. The domain is modeled as a network with

nodes that represent knowledge states and misconceptions, and edges that represent the transitions between sets of states. Edges can have different functions or conditions of applicability, including causal links between knowledge states and/or misconceptions. Bayesian networks are typically based on structures defined by experts, but there are also data-driven approaches. It is important to keep in mind that Bayesian approaches, in contrast to competing models, are probabilistic and thus able to model uncertainty. The Andes tutoring system discussed in section 4.3.4 implements a Bayesian knowledge tracing learner model.

3.5 Challenges

Educational Data Mining and Learning Analytics are concerned with modeling learning. Since different learning domains differ in their domain structure and pedagogical approaches and comprise very different subjects, the two fields are by definition interdisciplinary. Interdisciplinary work requires a team with experts from each domain. It is not sufficient that some team members involved have a little bit of knowledge or even no knowledge in one of the fields involved. The Dunning-Kruger effect states that people with less knowledge tend to overstate their perceived knowledge of fields when compared to real experts in fields (Kruger and Dunning, 1999). This can lead to making wrong decisions and being confident about them. The solution to this is to involve real experts from all the domains relevant for the Learning Analytics or Educational Data Mining project to help make sound decisions.

Another important challenge is that developing, testing, and maintaining systems is difficult and expensive. This holds for the duration of research projects (in the context of which tutoring systems are often developed), but is aggravated after the end of projects (typically three years). Retaining developers and the competence in developing a specific system over time can be difficult. Furthermore, often research prototypes are developed under time pressure and lack good code documentation, making it more difficult for other programmers later to work on the code quickly.

In an ideal world, Learning Analytics and Educational Data Mining are cyclic processes where systems are always updated based on the gained knowledge, re-evaluated, and so on. Due to the limitations discussed, this can be difficult in reality. Related to that is the fact that standards change over time and should also be updated. However, once something is updated, it should be evaluated, and the feedback should be incorporated in the refinement. For example, user interfaces outdate after a while – a user interface of a web application from 20 years ago is significantly different from a web application now since the standards have evolved over time.

Another aspect that can cause problems for Learning Analytics and Educational Data Mining is the question of ethics. For example if a system provides misleading feedback and this leads to a bad grade, who is responsible for this? Data privacy in general, but also the question of who makes directives about

how to deal with problematic cases needs to be considered carefully. For example, how should teachers be treated who seem to under perform in comparison with their peers? In general, a problem for systems that are developed or used in more than one country is that different data regulation frameworks apply in different parts of the world. This can lead to an information asymmetry between the involved stakeholders that goes up to the point where there is a shift of power and responsibility from state-regulated institutions to digital publishers.

Learning Analytics and Educational Data Mining tools need to address real needs and solve problems for all involved stakeholders and not just provide some fancy or primitive visualizations or only address the needs of one party. For example in the case of tutoring systems, teachers should have less (tedious, repetitive, time-consuming) correction work and more time for preparing classes. Students (and parents) need to benefit in that learners learn more and more deeply, get better grades, and spend less time stuck on exercises. Administrators should benefit from better grades for students and graduates, and from teachers that are satisfied with their job. If a tool does not address real-life needs, then its adoption will be very difficult (cf. also Meurers et al. (2018)).

Finally, a huge challenge is that a lot of Learning Analytics and Educational Data Mining approaches attempt to explain or approximate the human learning process with features that are easy, reliably measurable, and do not overwhelm the involved stakeholders. In contrast, the human learning process is extremely complex and not understood entirely up to this day. It is questionable if such a complex process can be approximated sufficiently with simple models involving only a few factors.

Chapter 4

Tutoring Systems

From a high-level perspective, tutoring systems are software applications designed for teaching aspects of a learning domain to a specific target group of learners.

In order to achieve this, tutoring systems model a part of a domain to be learned, for example a foreign language or an area of mathematics. This can be done explicitly with a domain model where the structure of a domain is made explicit, such as with a hierarchical structure of concepts or with knowledge rules allowing to solve problems from this domain.

Or it can be done implicitly with another essential component: tutoring systems need to provide materials from the domain under consideration. This can be exercises, but also instructional materials such as fact sheets or educational videos.

Materials are worked on by users, thus requiring that tutoring systems have some kind of information about who is interacting with it, and a user interface for interaction. In the simplest case, the user model is simply a session identifier or user name, but many tutoring system go beyond that and also model properties like exercise completion states or some form of competence estimation.

On top of a domain model, a user model, and a material pool, tutoring systems implement one or more pedagogical strategies in a pedagogical module. When users interact with a tutoring system, the system reacts to user actions with some form of feedback. This can be for example explanations about incorrect answers, statistics, or a suggestion of new exercises or materials. Arroyo et al. (2014) summarize this in a concise manner: according to them, instructional systems model or predict some aspects about learners, and use this information to chose a pedagogical strategy to help learners master a part of the learning domain.

Optionally, and depending on the intended usage, tutoring systems can inform other stakeholders about interaction data in the system. In the case of a tutoring system employed in schools, teachers typically are informed about what learners from their school class did or how they interacted in the system. Such extensions with reporting tools about learning actions are implemented as

Learning Analytics modules.

Tutoring systems are used in different and diverse educational contexts. In a relatively traditional setup, they can assist learners and teachers in the form of homework helpers. In flipped classroom scenarios, they can be used in schools or at home. Distance education, as in the case of massive open online courses, offers the possibility to deploy tutoring systems for a high number of users without direct involvement of teachers. Finally, many tutoring systems get used in purely experimental contexts or research labs.

In order to delineate tutoring systems from other software for the purpose of this thesis, we require that tutoring systems fulfill two criteria: interactivity and adaptivity. Interactivity in this context means that learners get the opportunity to actively interact with the system and not only passively consume its content, as would be the case in a system only offering educational videos. Adaptivity in this context means that these systems are aware of certain learner actions and react on them, for example by adapting the materials, providing feedback, or visualizing collected past data for users.

The rest of this chapter is structured as follows. Section 4.1 explains the architecture of tutoring systems in depth. Given the importance of feedback in tutoring systems and for this thesis, we dedicate section 4.2 to a discussion of strategies for and steps in providing feedback. In section 4.3, we take a look at non-language tutoring systems before moving to a discussion of language tutoring systems in section 4.4. We conclude this chapter with a review of the usage of tutoring systems in authentic or even ecologically valid contexts in section 4.5.

Disclaimer: This chapter and its parts have not been published before.

4.1 Architecture of Tutoring Systems

The architecture of tutoring systems in general consist of three pillars (e.g. Padayachee (2002)): a student model of the learners and their current state of knowledge, a domain model containing expert knowledge about the contents of the learning domain, and a tutoring module that contains information about how to instruct the learner. In addition, there are practical requirements like a user interface allowing learners to interact with the system, a data base for storing records, and source code that connects the listed modules with each other. Figure 4.1 depicts the conceptual architecture of tutoring systems. In the following sections, the conceptual components, abstracting from specific implementation issues, will be described.

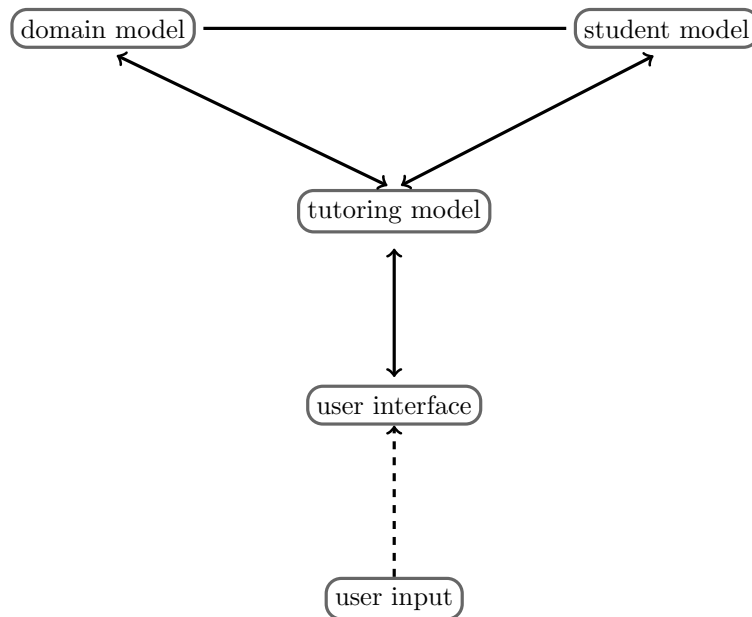


Figure 4.1: Conceptual architecture of tutoring systems

4.1.1 Domain Model

The domain model contains a structured representation of the learning domain. Essentially it models what should be learned by learners using the tutoring system, how this knowledge is structured, and how this knowledge can be used to solve problems in the domain.

In order to achieve a structuring of the domain, it is necessary to break the domain knowledge down into smaller parts. This division of the domain into learning points or knowledge items has to be performed by experts in the specific domain. For this reason, the domain model is often also called expert model

(Anderson, 1988; Ramesh and Rao, 2012). The complexity of the breakdown of a domain model into smaller units correlates with whether the learning domain is a well-defined domain (in which case by its nature it is easier) or an ill-defined domain. In section 4.2.1, we provide a discussion of ill-defined versus well-defined domains.

The domain model on the one hand contains facts, and on the other hand rules. Nkambou (2010) describes these two components of a domain model: a knowledge representation containing declarative knowledge, and an inference mechanism representing procedural knowledge.

For example in the case of language learning, these rules can be in the form of NLP components that process text and provide analyses thereof. It can also contain patterns for typical errors or mechanisms to detect them (Padayachee, 2002) (cf. section 4.2 for a detailed discussion).

Slavuj et al. (2017) describe a range of different approaches for structuring and modeling domains, which will be discussed in the following. In the simplest case, all items are defined independently without any relations between them. In this so-called “vector model” (Slavuj et al., 2017, p. 72), each knowledge item is represented as one dimension in a vector space with each smallest knowledge unit completely independent of all others.

In contrast, network models contain links between knowledge items. These links can have different categories. An instance of such models consists of encoding that certain knowledge items are prerequisites for other items. This allows to draw inferences like if one skill or knowledge item is known by a learner, then it is likely that those forming the prerequisites are also known. Another instance of modeling domains is to order skills or knowledge items by parameters related to cognitive complexity, such as the requirement of different skills to complete an exercise (Nkambou, 2010).

Another approach is to model that certain knowledge items are sub items of more general items/skills, forming a hierarchical representation of domain items. Going one step further in this direction, ontologies have been proposed as an instance of semantic networks to map knowledge items and logical relations between them (e.g. Wang et al. (2014)).

Amaral and Meurers (2008) discuss structuring domains according to acquisition sequences found in Second Language Acquisition research and structuring the domain according to course curricula.

Another approach discussed by Nkambou (2010) is to use frame-based approaches like HPSG (Götz and Meurers, 1995) in order to define semantically rich ontologies with links between the frames. Götz and Meurers (1995) point out that there has to be a trade-off between expressivity of the domain representation language and the computability of the approach.

There are different definitions as to what constitutes and is modeled in the domain. On the one hand, there is the possibility to index knowledge items, like for example mathematical axioms or rules, or rules for tense formation in language teaching. On the other hand, there is the possibility to add more complex material. Slavuj et al. (2017) and Brusilovsky (2012) discuss an approach called indexing that goes beyond purely structuring knowledge items

and also includes an indexing of educational materials such as exercises in the domain model. More specifically, educational materials are mapped to smaller knowledge items or skills.

4.1.2 Student Model

A student model, or from a more general perspective, a user model, is a set of variables associated with a user in a tutoring system. Generally speaking, this models someone who is learning the contents of the domain model. There are different dimensions for classifying the contents of student models (cf. Slavuj et al. (2017)) on which also the following discussion is based). One dimension is the distinction between stable aspects and dynamic (i.e. frequently updated) content. In this case, the static part is called learner profile and the dynamic part learner model. Another dimension is to distinguish between (learning) domain-related and domain-unrelated data. In any case, a learner model can be conceptualized as a map with keys mapping to values, with the keys being from the domain model and the values different among users.

“Stable” here does not necessarily mean that the values are never changed, but they are typically stable over longer periods of time, such as the duration of an entire seminar. Instances of variables in this area are the name of learners, their user id, their date of birth, their gender, their background (e.g. first language), but also individual difference measures classified as stable traits like personality traits (Matthews et al., 2003). To a certain extent this can also include factors such as their previous knowledge (e.g. preceding courses taken) or their instructor for a specific course.

Moving on the spectrum towards more dynamic variables, student models can represent information stable over one session. This typically includes the device and software (e.g. web browser) used, the location of the learner, the time of the last login, and the time of the day.

Finally, the student model contains dynamic fields that can have multiple changes in their values during one session. This on the one hand includes domain-independent factors such as a user’s navigation history, affective states, and measures of specific task strategies (Amaral and Meurers, 2008). On the other hand, there are domain-specific variables like scores for the proficiency related to certain domain contents, lists of activities related to specific domain contents, or task-specific measures like number of attempts or frequency of errors.

The dynamic part of a learner model consists of concrete and frequently updated measures for sub parts of the domain model. In summary, the domain model represents what can be learned, and the student model represents how well this has been learned by an individual student.

The goal of this section is to provide a general description of student models in the context of tutoring systems. In section 3.4 we dive into details about how the dynamic measures of competence mentioned here are represented, measured, and updated.

One distinction of student models that does not pertain to the modeling of competence, but to its presentation is to distinguish closed from open learner

models.

Open Learner Models

Closed learner models are only visible to the developers of systems, whereas open learner models are visualized for and visible by end users. End users can be different user groups, and the selection and visualization of the underlying data can and often should vary depending on the user group it is visualized for (Lee and Bull, 2008).

Reasons for Opening Learner Models Bull and Kay (2010) discuss reasons for opening learner models, which will be used as a basis for discussion in the following. The first reason for opening learner models is to not only display the data, but additionally provide options to edit the data. When users first interact with a tutoring system, no or only very little data is available. The problem is that the tutoring system has to base its pedagogical decisions on some data. This data can come from the domain model, and/or the student model in order to personalize the system behavior. However in the beginning with only few interactions the data in the learner model is very sparse, and due to this, the few data points are over-interpreted since they can not be compared or put into relation to other values. This issue has been discussed in the literature as “cold start problem” (e.g. Zhang et al. (2014)). In order to overcome this, a possibility is to open learner models for end users so that they can fill in missing data. Opening learner models for edits opens up the potential for misuse, making it important to implement checks for ensuring the validity of edits, like for example test questions that pertain to the specific parameter and have to be answered in order to accept the edit (e.g. Mabbott and Bull (2006)). Another reason for opening learner models is to increase and foster meta-cognition. The goal is to encourage learners to reflect on their learning, both by rewarding them to see the system acknowledges they have learned a specific knowledge item and to allow them to identify knowledge gaps and take action to fill these gaps. Similarly, if a learner model is open to teachers or parents, it can serve as the basis for planning, interventions, and evaluation. If the model is in addition open to, or available in some form, to other learners, it can stimulate competition among learners (Brusilovsky et al., 2015). In case several learner models are open to a teacher, this can help instructors to form groups for collaboration among the learners since students with complimentary strengths and weaknesses can be paired up upon comparison of learner models.

Furthermore, the learner model can serve as a means of navigation in the tutoring system in that the learner model can contain suggestions for exercises or pointers to learning materials learners directly accessible from there, with the content of the pointers depending on the learner model of an individual student. Another purpose of open learner models is to realize the (legal) right of users to know what data a system has collected about them. Especially in European

legislation where the DSGVO¹ grants each user the right to ask for all the data a system has collected. This need can be satisfied with an open learner model displaying all the information.

Finally, learner models can serve as a means of assessment. In contrast to exams that only test knowledge in a “snapshot approach”, i.e. only testing learners in a short period of time, learner models collect data over much longer periods of time and can serve as an alternative means of assessing the knowledge of learners and its temporal development. This longitudinal assessment shifts the focus away from performance more towards competence since over longer periods of time differences in performance can be ironed out with a larger data point sample size.

Visualizing Open Learner Models Open learner models have been implemented in a range of different ways and presentation styles. Even with the same underlying data, many different visualizations are possible, plus the data can be aggregated in new ways if needed for specific visualizations. Since the goal of Learning Analytics (see chapter 3) is “the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environments in which it occurs” (Siemens and Baker, 2012), the task of visualizing data in an open learner model is a Learning Analytics task.

In their literature review, Bull and Kay (2006) present a range of visualizations of open learner models. In text-based formats, different text snippets can be shown depending on the student level. Similarly, numerical values like probabilities for knowing a concept can be displayed in textual format. Graphical representations of skills can be skill meters, where progress bars act as an indicator of progression or knowledge (e.g. Mitrovic and Martin (2002)). Alternatively, or as a supplement, colors have been used with different color values and intensities signifying different levels of knowledge (e.g. Zapata-Rivera and Greer (2004)). This can be extended to include the values of peers by adding more bars or lines. Similarly, line charts have been proposed with average lines to allow learners to see whether they are above or below the given average (Bull, 2004). Going beyond individual indicators, visualizations of networks have been proposed where the visualizations of individual components are connected and form a larger picture of how knowledge is distributed among areas of the domain (e.g. Mabbott and Bull (2006)). Special considerations have to be made if the open learner model’s primary users are younger children. In this case, special emphasis has to be put on a visualization that is understandable and accessible to them (Bull et al., 2005). For example, text-based formats are less recommendable than more graphical types of visualization. Another consideration is whether users only have access to one visualization or to multiple visualizations, potentially linked to each other with more information available if requested or users zoom in for details.

¹<https://eur-lex.europa.eu/eli/reg/2016/679/oj>, last accessed 2019-08-28

4.1.3 Tutoring Model

The tutoring model, also called adaption model (Slavuj et al., 2017) or instruction model (Amaral and Meurers, 2008), links the domain model with the student model (cf. Figure 4.1), and steers the interaction with learners. As (Ramesh and Rao, 2012, p. 1) puts it, the tutoring model “designs and regulates instructional interactions with the students.” The key task of this model is to provide an adaptive and interactive link between the learner on one side, and the domain contents and educational material on the other side. Chatti et al. (2013) distinguish between adaptivity and adaptability. Adaptivity means that a system automatically adapts contents to learners, whereas adaptability denotes the concept of learners having the possibility to adapt a system according to their preferences. In this section, we refer to the concept of adaptivity. This adaptivity can have different forms discussed in the following.

Sequencing of Material

One instance of adaptivity in the tutoring model is to adapt the sequence of educational resources (exercises, content pages, tests) according to the learner model and the domain model. The alternative to not having a sequencing model would be to provide no guidance at all to the learner what to do next. Conceptually there are two broad strands: prevent the learner from accessing higher-level material until certain criteria are met, or suggest material to the learner to interact with next.

This implies that the instruction model has information about the level or difficulty of educational material. In the simplest case, this is approximated via an index assigned to each educational material, such as a theme, chapter, or page number in a work book. In a more complex scenario, explicit models of task difficulty are used (Quixal (2012), cf. also section 3.4.1 in this thesis). A task model making explicit factors such as the complexity and types of the constructs involved, the format of the task, the strategies required to successfully complete the task, and the previous performance of learners on it allows the tutoring model to compare these task models with the model of the current learner and suggest tasks that are most likely to maximize the learning gain for this learner. The sequencing does not necessarily have to be restricted to exercises, but instead a system can also suggest other educational resources like explanation pages to the learner. Slavuj et al. (2017) further suggest to incorporate factors such as the time of day in order to provide learners with material when they have shown to be most productive. Another approach, involving simpler or no task models, consists of automatically learning sequences of effective tasks and using these sequences to suggest new exercises. An approach allowing this is to employ association rule mining (Kotsiantis and Kanellopoulos, 2006).

Adaptation of the Presentation of Material

Another possibility to adapt material is to modify the presentation of (already selected) material to the learner (Slavuj et al., 2017). This can be achieved

by having pre-compiled, different versions of the same material, or it can be done by having a modular approach where parts of an exercise can be swapped. There are different parameters that can be altered such as the complexity of the material (Robinson, 1995), the modality (visual/textual/dynamic/etc., e.g. Holzinger et al. (2008)), the salience of specific parts of the material (White et al., 1991), but also the explicitness of feedback provided to learners (Heift and Rimrott, 2008).

Note that this involves, but goes beyond decisions regarding the presentation of feedback (section 4.2.3), since it involves altering the presentation of entire exercises.

Provision of Feedback to Learner Productions

The provision of feedback to learner productions (in Figure 4.1 “user input”) constitutes a form of adaptivity in the sense that the system adaptively has to process user input and select a tutoring strategy to react to it. The tutoring model is responsible for passing the learner production to the domain model. In the domain model, the necessary processing tools are present that have the ability to analyze the learner production (section 4.2.1). The analysis is returned to the tutoring model, which then consults the student model. Based on the learner’s previous interaction data, the analyses returned from the domain model are re-ranked in the tutoring model (section 4.2.2) and a strategy for presenting feedback is selected based on the analysis and the student model (section 4.2.3). In section 4.2, we explain the entire process of taking a learner production as input, processing it, selecting a pedagogical strategy, and displaying feedback in detail.

4.1.4 User Interface

The task of the user interface is to provide a means of interaction for users with the system. Fundamentally there are two types of interfaces that can be used: graphical user interfaces and text-based interfaces. With early operating systems for home computers only providing console-based user interfaces, early tutoring systems like Typing Tutor III from 1981² needed to make use of the command line as a text-based interface. Since console-based tutors however only constitute a minority in comparison to tutoring systems that offer a graphical user interface, we base our discussion of user interfaces for the remainder of this section on graphical user interfaces in tutoring systems.

In modern devices where user interfaces can be displayed not only as text like in console programs, it is common to have input modalities that go beyond keyboard input only. On desktop and laptop computers, commonly a mouse or touch pad is available in addition to a keyboard. This opens up opportunities for inputs like drag and drop or point and click. When moving to mobile phones, typically even more sensors are available that user interfaces can make use of

²https://archive.org/details/a2_Typing_Tutor_III_1981_Image_Producers, last accessed 2019-08-30

and, like in the case of touch screens, also have to accommodate for. These include for example brightness sensors, the location of the device in terms of coordinates, acceleration and movement sensors, cameras, microphones, heart rate sensors, eye tracking sensors, and thermometers. With the broad range of devices a system can be deployed on, especially if it is a web application, it is important that the user interface adapts to multiple resolutions, operating systems, and screen sizes. This is typically achieved with a responsive design that uses breakpoints to switch components of the layout depending on viewport size parameters (Kim, 2013).

User interfaces should have a good usability and support learners in learning instead of posing obstacles. Molich and Nielsen (1990) in their seminal work proposed nine design principles for user interfaces that involve human-computer interactions, which will be listed in the following. User interfaces should be restricted in terms of the information presented to the necessary facts and avoid unnecessary and superfluous information. This information should be formulated in consistently used language understandable to the end user instead of technical system terms. This also means that the same action should always be associated with the same term. Furthermore, the user interface should be self-explanatory and intuitive, and if necessary, help and guidance should be easily available. The user interface should be responsive to user actions and provide feedback in a time frame comprehensible to the user. They describe that it is important to always provide clear exits from every part of the interface to navigate quickly and without hurdles. Related to this mention that a system should provide shortcuts for experienced users to work efficiently. Finally, a user interface should be stable, i.e. avoid crashes, but if a crash occurs, error messages should be clearly formulated, provide useful suggestions for next actions, and avoid blaming the user for the crash.

Moving from fundamental design principles, Schrepp et al. (2017) emphasize the importance of testing user interfaces. They present the popular User Experience Questionnaire (UEQ) that contains questions on six scales related to the usability of user interfaces. The first scale is overall attractiveness of the system, i.e. whether users in general subjectively like the system. The second scale is perspicuity, meaning designating how easy it is for users to get used to the interface. Similarly, the third scale, efficiency, measures how efficient users can handle the interface. The fourth scale is dependability. This indicates whether the user feels in control and secure when in the human-computer interaction. The last two scales, stimulation and novelty, capture if the user interface is motivating and creatively designed.

In the case of tutoring systems, specific considerations need to be made, which are partly covered by the previous points, but for the sake of clarity shall be made explicit. The user interface of a tutoring system needs to be tailored towards the groups of users that use it. As motivated in section 4.1.2, different user groups such as students, teachers, parents, and administrators can be given access to the system and then need a user interface that suits their needs (Rudzewitz et al., 2017). This also implies that the interface is age-appropriate in its presentation and (if present) language. In case of tutoring

systems, it is crucial that the user interface provides a means of displaying feedback. Depending on where the feedback should be displayed (overlay of text, inline, below the text), this factor has to be considered early in the design phase of the user interface of a tutoring system.

4.2 Strategies for Providing Feedback in Tutoring Systems

The process of taking a learner production as input and generating and displaying feedback to the learner can be broken down into five conceptual steps, illustrated in Figure 4.2. Given a learner production, such as a word or a sentence, the first step is to generate potential analyses of this user-generated content. The next step, denoted with a (2) in the figure, is to select a target hypothesis from the set of analyses generated in step (1), i.e. an interpretation that, given the task and learner, is most likely in this context. In the next step (3), tutoring systems need to determine how to use the selected target hypothesis, thus determine a pedagogical strategy and rank feedback candidates in order to present the interpretation of the analyses to the learner. This involves decisions about factors like modality or location of presentation of the feedback. The last step in the figure is optional and for this reason depicted with a dashed line. This step represents the usage of Learning Analytics. When feedback is presented to the learner (step 3), the system can record if and how the learners reacts to the feedback (step 4), for example how long they look at the feedback message, or whether they manage to address the error pointed out by the system. This information can be stored in both a learner model (to record measures of competence/performance of individual learners), as well as in a task model to get keep track of the complexity and interaction with a task. This information then can be fed back into the loop to adjust feedback strategies. In the following subsections, all these steps are explained in more detail.

4.2.1 Analysis of Learner Production

The first step in providing feedback in a tutoring system given a learner production is to perform an analysis of the learner production. This analysis varies depending on different factors. First of all, the learning domain of a tutoring system can determine the type of learner production and thus the analysis strategies. In the literature about tutoring systems, usually two domains are distinguished: well-defined domains and ill-defined domains (e.g. Roscoe and McNamara (2013)).

Since the subsequent work is based on the processing of (learner) language, we turn to a discussion of how ill-defined domains are different from well-defined domains. Lynch et al. (2006) define five criteria for distinguishing ill-defined from well-defined domains, which will be described in the following.

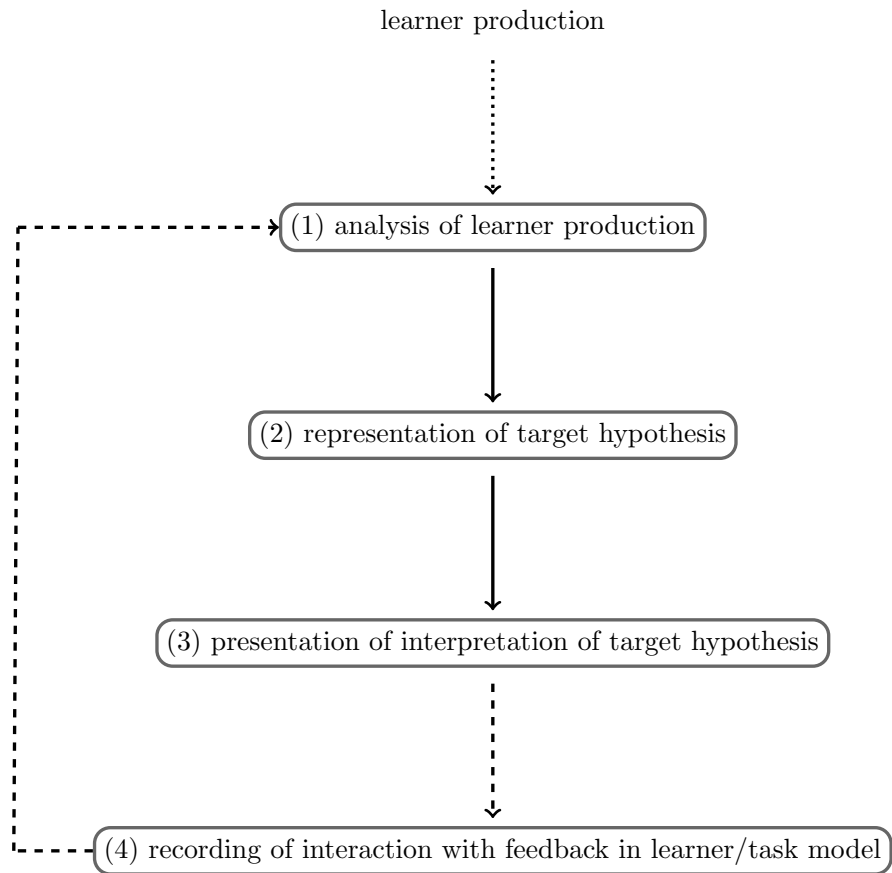


Figure 4.2: Conceptual overview about feedback process in tutoring systems

Definition of Ill-defined Domains

The first criterion is verifiability: in well-defined domains, a production by a learner to a given task is clearly verifiable. This means that there is (at least) one single clear correct solution, and the criteria for its correctness are precisely, explicitly, and formally defined. In contrast, in ill-defined domains there is typically not one correct answer, but an undefined number of answers can potentially be viewed as correct. Qualitative reasoning and world knowledge instead of or in addition to formal criteria can be employed to determine whether an answer is considered correct or not. An example provided by Lynch et al. (2006) is an exercise in mathematics, for which a clear verifiable solution exists, in contrast to a task in architecture, where many competing solutions for one task can exist.

The second criterion is the existence of formal descriptive theories. In well-defined domains, there are formal, empirically validated domain theories that comprehensively structure and describe the domain knowledge. In contrast, in ill-defined domains, theories typically cover parts of the domain only, and different theoretical attempts at describing the domain with different degrees of empirical validity compete. As Lynch et al. (2006) mention, domain theories in ill-defined domains (such as law) tend to be more prescriptive, whereas domain theories in well-defined domains (such as physics) are more descriptive.

The third principle mentioned is the structure of tasks. Similar to the first principle, the authors argue that in ill-defined domains, tasks often have a "design" (i.e. more open-ended) nature encouraging creative solutions, whereas in well-defined domains, tasks require primarily analytical strategies.

The fourth and fifth characteristic mentioned is that in ill-defined domains, concepts are abstract and consisting of multiple sub concepts that can not be solved independently of each other. Concepts in ill-defined domains, such as for example a "good movie" from the domain of film studies, are abstract and not clearly defined in all aspects and design principles. Sticking to the example of a movie, a movie scene consists of many sub concepts such as the angle of the camera, the lighting, the location etc. It is not possible to treat them completely independently since the mutually influence each other.

Language as an Ill-defined Domain

Hockett and Hockett (1960) describe 13 design principles that characterize human language. Of relevance to our discussion are the principles discreteness, productivity, semanticity, arbitrariness, and displacement. Discreteness describes the phenomenon that languages are compositional (to a certain degree) into smaller, discrete components. Productivity describes the fact that these smaller units can be composed in productive ways to form completely new utterances that are nevertheless understandable to speakers of this language. Arbitrariness describes the fact that there is no direct link between the form in the language and the object denoted, and semanticity the fact that signal units have an associated meaning. Finally displacement describes the ability to talk about entities

or affairs not present at the time of utterance, such as past or future events. These properties combined lead to human language as an open communication system allowing to talk about abstract concepts in arbitrarily many new ways.

The task of short answer assessment (SAA, Burrows et al. (2015)) will be used as an illustration of these findings and their relevance and relation to language as an ill-defined domain. In short answer assessment, the task to automatically classify or grade a short answer given to a question (Ziai, 2018) in terms of the answer's semantic correctness. The task is to allow for form variation while focusing the comparison on semantic aspects of the answer. The evaluation is performed by comparing a learner answer to one or more target answers, commonly provided by the question author. Due to the described properties of human language, it is very likely that a system grading short answers is provided with unseen answers (e.g. Heilman and Madnani (2013)). The unseen answer can be new in different aspects: it can contain unseen vocabulary, unseen grammatical structures, or unseen semantics. This requires short answer assessment systems to abstract away from the concrete form to more abstract levels of comparison. In this abstraction however, important nuances can be lost, and it is not clear on which level of abstraction a comparison should take place. An additional complication arises when learners use hypernyms or hyponyms, or provide extra not-at-issue-material (Tonhauser, 2012). While there are more challenges related to short answer assessment, these examples serve to illustrate that the verifiability of short answers, and by extension language, is not straightforward, and that language thus falls, under this perspective, into the category of an ill-defined domain.

The second criterion of ill-defined domains is a lack of comprehensive formal, descriptive theories. While for certain languages like English a lot of different theoretical frameworks for the analysis of aspects of the language have been proposed, many smaller languages are under-researched or even in danger of being lost (e.g. Hale (1992)). Provided that language loss is happening now, and that lost languages are not recoverable once lost, there is no general theory of language, which would be a criterion for well-defined domains.

Coming back to our example of short answer assessment, it can be argued that since short answer assessment is often done with questions posed for reading comprehension requiring the extraction and summary of information from a reading text and the synthesis of an answer based on this information, these tasks require creative work in summarizing and extracting answers and lead to linguistic variation in the learner responses (Roy et al., 2015). Other task types like fill-in-the-blank exercises with pre-defined lemmas are less open-ended, but the spectrum of tasks related to language include open-ended tasks with design elements.

The fourth criterion, the abstractness, or lack of clear definition of concepts, becomes evident when looking at distributional semantics models. In distributional semantics, words are represented as high-dimensional vectors with each dimension standing for a context the word occurred in in some training data (Erk, 2012). In this framework, words are treated as concepts that are not explicitly defined, but implicitly as some combination of the contexts they occur

in. Approaches have however been made to build ontologies and hierarchies of words, such as WordNet (Hirst and St. Onge, 1998), to explicitly model the semantics of words by providing explanations, examples and relations between words/concepts such as synonymy, hypernymy etc. One challenge with this approach is that the examples and explanations are again consisting of language material for which a definition is required. This is a recursive problem in that the object under consideration and its explanation are both language. This and the fact that in distributional semantics, the dimensions in the vector are only useful when in combination with the other dimensions, clearly fulfills the criterion of ill-defined domains with respect to an interdependence of sub problems and an abstractness of the concept under consideration.

The application of these criteria shows that language can be considered an ill-defined domain. What aggravates this problem is that in language tutoring systems, the type of language being processed is learner language, for which there is not necessarily a one-to-one correspondence between the form of the language and the meaning intended by the learner, introducing another significant level of ambiguity (Díaz Negrillo et al., 2010). In this case, language is not only to be considered an ill-defined domain, but the language being processed is in addition ill-formed and requires special analysis methods to derive interpretations that accommodate for this fact rather than gloss over it. In the next subsection, we discuss this ambiguity and approaches to deal with it.

Deriving Linguistic Analyses of Learner Language

Learner language, in contrast to standard language produced by native speakers, often contains non-standard elements. As Díaz Negrillo et al. (2010) demonstrate, learner language can exhibit divergences on the level of distribution, morphology, and lexicon. In standard language, when looking at a specific form, the analysis found on the analysis level of distribution, morphology, and lexicon converge and lead to one interpretation of a form (if the context is not ambiguous). In contrast, for learner language it is the case that the evidence at the various levels point to different analyses and it is not clear how to select one of the alternatives. For example, (Díaz Negrillo et al., 2010, page 8) show the example “to be choiced for a job”. In this example, the word “choiced” exhibits a mismatch between its stem and its distribution, as well as a mismatch between its stem and morphology. The stem of the word is a noun (“choice”), but it is used in a distributional slot expecting an adjective. Furthermore, it contains a morphological marker that is used to mark past participles (“-ed”), but this suffix is attached to a noun and not to a verb as it is done in standard language.

In these cases multiple competing analyses are possible. In order to arrive at these analyses, two steps are necessary: (1) analyze the learner language on different levels of linguistic evidence to derive competing analyses, and (2) represent the different analyses in order to allow for a selection of the most likely one(s). For the remainder of this section we illustrate step (1), and in section 4.2.2 we discuss step (2).

Meurers (2012) describes strategies to analyze learner language that serve to

identify linguistic divergences instead of glossing over them or ignoring one level of linguistic evidence, since the linguistic divergences of evidence are the cases tutoring systems should be able to recognize and provide feedback on. Meurers (2012) distinguishes two conceptually different strands: language licensing versus pattern matching approaches. In the former, the complete learner answer is analyzed and needs to be parseable by the analyzer. They describe two mechanisms for language licensing: in a validity-based approach, the idea is to have a set of formal grammar rules that consists of both rules for licensing standard language, as well as extensions in the form of rules that cover certain non-standard phenomena found in learner language. This approach allows to parse learner language that would not be parseable with a grammar for standard language only, and the type of non-standard rules ('mal-rules') used in the parse analysis can provide guidance towards diagnosing a specific error in the learner answer. As an alternative, constraint-based approaches are discussed. In this case, the formal grammar consists of a set of constraints that must not be violated. The constraint(s) violated can give insights into the divergence in the learner answer. Conceptually different are approaches that are based on pattern matching. In this case, not the complete learner answer is parsed, but instead patterns are applied that, if matched, indicate the presence of a specific mistake or error. The key difference between pattern matching and validity-based approaches lies in the fact that in the latter case, the complete answer is parsed, whereas in the former case, only parts of the answer are parsed (those that contain the error/mistake).

Another approach discussed in the literature to derive analyses of learner answers is to not directly analyze learner answers, but rather compare them via (non-linguistic) similarity metrics to previously analyzed learner answers (e.g. Paaßen et al. (2017)).

All these approaches have the goal to derive a set of analyses of learner answers that accommodate for the fact that these learner answers contain divergences when compared to utterances in the standard language. The next step, described in the next section, is to derive one or more target hypotheses from the set of possible analyses.

4.2.2 Representation of Target Hypotheses

In the previous section we explained how (linguistic analyses) of learner language can be computed and how often multiple competing analyses of learner productions are possible. This section is concerned with two interrelated questions: how to integrate one or more of the generated analyses into a target hypothesis, and how to represent target hypotheses.

Target hypotheses are "some kind of reconstructed learner utterance" (Reznicek et al., 2013, p. 1). Target hypotheses are modified variants of learner answers where annotators have changed one or more parts of the learner answer. The specific correction of one part of a learner error is often dependent on a specific correction of another part of the learner production, a typical characteristic of ill-defined domains (cf. section 4.2.1). This leads to the fact that for the same er-

ror(s), there is ambiguity since it is not clear what the learner wanted to express. This ambiguity can be represented by annotating different target hypotheses prioritizing different aspects in the learner answer. Lüdeling et al. (2005) show how one learner answer can be altered in terms of orthography, grammar, lexical aspect, information structure and style. This generation of multiple learner production variants on one single and on multiple levels of linguistic representation leads to the question of whether these minimal target hypotheses should be treated independently or whether they should be combined in some manner, i.e. viewed not independently but rather emerging from a joint cause manifesting in multiple places. Lüdeling et al. (2005) argue for combining minimal edits into two target hypotheses for each learner answer: target hypothesis one contains corrections only on the grammatical and morphological level, whereas target hypothesis two contains corrections on the pragmatic, stylistic, and lexical level. This bipartite approach allows to represent corrections of learner utterances in terms of (1) form and (2) meaning.

The next step is to represent these target hypotheses, both conceptually and in practice. Lüdeling et al. (2005) present different approaches discussed in the following. One approach is to collapse multiple target hypotheses/analyses into one single complex annotation. This annotation can be represented easily in a tabular format with each line standing for one word and its annotation. Advantages are the simplicity of the approach and the unification of information in one category. On the other hand, this makes it difficult to have overlapping or discontinuous annotations. Having complex labels that join information, Portmonteau tags (Meurers, 2015), can potentially lead to a difficulty in interpreting these complex categories and a large number of artificially created categories. Furthermore, contrary to what is claimed in the article, it is possible to have categories that span multiple words if the tags contain indicators of whether they are part of a sequence or not, analogous to how the UTF-8 encoding works with characteristic high bits for multi-byte sequences (Allen et al., 2012).

Another approach discussed is a tree structure annotation. In this approach, inline markup is used to annotate spans of the learner answer. The approach discussed uses an XML-based mixed content model in which tags and text are on one level. This allows to have annotations that span more than one single word, and also annotations that are nested within each other. A problem with such a mixed content model is however though that due to a core principle of XML stating that every XML document should be mappable to a well-formed tree (e.g. Harold and Means (2004)), it is not possible not have overlapping spans on the same level since this would violate the tree structure of XML. Furthermore, the introduction of annotations directly inside the source text can make it non-trivial to revert to the original text because it is modified and meta characters with a special meaning are introduced, such as '<' or '"', requiring these characters to be escaped or replaced by entities in the original text, adding another level of modification of the original. This violates best practices in corpus annotation, as discussed for example by Leech (1993).

The third approach discussed is the usage of multi-layer standoff annotation models. In this setting, annotations are not placed inside the source text, but

rather in a separate location and refer to the original text with the means of indexes that point to specific places in the original document. Since the annotations can be independent, but can also potentially refer to other annotations, this approach allows for an unlimited number of annotations and annotation layers with potentially overlapping spans. This representation, like the tree-structure approach, can use XML as a representation format, but with the annotations placed in a different part of the tree. This model does not require a collapse of annotations into combined tags, but allows to represent competing analyses in parallel. This approach thus overcomes certain problems present in other approaches due to its flexibility, both from a theoretical and a practical standpoint, and therefore can be considered the most elegant solution to represent target hypotheses.

4.2.3 Presentation of Interpretation of Target Hypotheses

Having established what types of target hypotheses can in principle be annotated and how to annotate them in practice, the next question is how to proceed with a set of target hypotheses in order to provide some kind of interpretation of one or more target hypotheses to a learner using a tutoring system. It was established by van der Linden (1993) that in the context of a tutoring system, feedback with three or more lines of text is not read by most learners. While three lines seems like an arbitrary number, the underlying insight is that learners can easily be overwhelmed by too much information. This finding is in line with the cognitive load theory (Sweller, 2005). This theory states that learning is facilitated or hindered by intrinsic load (the complexity of exercise material), extraneous load (the presentation of the learning material), and germane load (the creation of schematic representations of regarding the learning domain).

These findings imply that it is advisable to not present to the learner the interpretations of all target hypotheses at the same time, but rather restrict the feedback to a subset of the analyses only. Heift (2001), based on van der Linden (1993), suggested to only present feedback describing one error at a time. This poses the problem of how target hypotheses can be ranked in order to select the most important feedback message for a specific student and exercise. This involves two considerations: (1) which target hypothesis should be chosen for an ambiguous part of the learner answer that can be analyzed in different ways, and (2) how to, given a target hypothesis, select the best pedagogical strategy for presenting the interpretation of this target hypothesis.

Ranking of Target Hypotheses

In ranking the target hypotheses, three factors play a main role: (1) the language, (2) the task, and (3) the learner.

On the **language** side, the target hypotheses should be based on and similar to the learner answer. The modifications should be minimal edits like described by Reznicek et al. (2013) and not attempts at replacing learner tokens with e.g. completely different words or inserting lots of new material. While this is a

rather abstract perspective, it can be seen as a guiding principle to be operationalized in a specific manner for specific applications. A concrete illustration of how to operationalize this principle would be to implement a beam search algorithm (e.g. Koehn (2004)) that uses the edit distance as a cost function and searches the space of edits while only considering the edits within a specific beam width and discarding hypotheses where the edit distance is too far off a pre-defined beam.

The second factor to be considered is the specific **task** for which the learner language was produced. The task should and can be made use of in different ways to weight target hypotheses. On the one hand, the task performed by the learner always consists of some language material. In the minimal case the context consists of the instruction, but often also more facets are available like a reading text for reading comprehension tasks, the transcription of an audio clip for listening comprehension exercises, or a list of lemmas provided in fill-in-the-blank tasks that need to be inflected and inserted into a gap. Priming is an effect researched on in psycholinguistics that describes the phenomenon that recently perceived words or linguistic constructions are more salient in memory and more likely to be used (e.g. Tulving and Schacter (1990)). Since a learner is exposed to the task context when working on an exercise, and since priming as an unconscious process takes place, a target hypothesis ranking algorithm should give higher weight to a target hypothesis with a normalization that occurs in the task context than to a target hypothesis that has a normalization for the same source material but with a normalization appearing not in the context. An example for such a task-specific weighting is a spell checker that gives higher weight and likelihood to words that appear in the context material rather than corrections to other words. Flor and Futagi (2012) present a context-sensitive spell checker with this property.

Furthermore, the task context can be used in another, complimentary way when prioritizing target hypotheses. A task always has a certain pedagogical goal with the aim of eliciting learner productions of a specific form and/or content. For example, a reading comprehension task with an associated task that asks the learner to compare two entities from the text might require the learner to make use of comparatives. The practice of comparatives is thus one of the pedagogical goals of this exercise. In this example, the remaining language that has to be produced serves only the secondary purpose to enable the usage of comparatives. For this reason, target hypotheses that correct an incorrect usage or formation of a comparative should be ranked higher than target hypotheses that target other aspects of the learner production.

The third factor to consider is the **learner** that produced an utterance. The concept is to use this learner's profile (cf. also section 4.2.4) in order to rank target hypotheses differently for each individual learner. This can lead to different target hypotheses being prioritized for each learner because each learner has a different background and history of interaction with the system. A first instance of how learner profile data can be used in order to weight target hypotheses is to check at which stage a learner currently is in the curriculum. In terms of lexicon and linguistic constructions, the curriculum serves as a proxy of what a learner

has already been exposed to or, ideally, learned. If there are target hypotheses with corrections that contain lexical material or linguistic constructions that are far beyond the current state of the learner, the information will be difficult to understand to the user. Furthermore, it has been shown that there are certain orders of acquisition in second language learning (Larsen-Freeman, 1976), and by combining the learner's current state in the curriculum and the order of acquisition of constructions, certain target hypotheses can be ruled out or ranked lower if there is a big divergence, i.e. if it is the case that a learner is very unlikely to have learned this construction or word already. An example would be a form that involves a tense form that is learned much later in the curriculum: in this case, a target hypothesis that correct the learner production to a tense that is near the current state of where the learner stands in the curriculum should be prioritized over a form that is much further down the curriculum. In the same line, vocabulary information can be used: if for example a beginning learner writes a nonword, and there are two target hypotheses, with one that correct the nonword to a very rare and exotic word, and the other one to a word that is on the same level as the learner is currently in, the latter target hypothesis should receive higher priority. Note that these types of rankings are not possible by using surface measures only like for example edit distance.

Another way of using the learner profile to rank target hypotheses is to use, in addition to the specific task, the constructs in the learner's Zone of Proximal Development (Vygotsky, 1978) and prioritize constructs that the learner is currently in the process of acquiring. A concrete operationalization of this approach could be to give higher priority to target hypotheses that contain corrections for which the learner has shown some uptake already, but has not mastered it (i.e. also still shows some errors).

Regarding the learner background, information about the learner's language background can be incorporated. Cross-linguistic influences, i.e. transfer from the first language to the second language, have been shown to have significant impacts on the language produced by learners in the second language (Odlin, 2005). Similarities between the first and second language, as well as the presence of constructions in the second language not present in the first language, have specific influences on the language and types of errors language learners make in the second language. Furthermore, similarities between other second languages and the target language learned at the moment have strong influences on the production of learner language (Ringbom, 2007). Therefore, target hypotheses that contain phenomena explicable by typical patterns of transfer from the learner's first language or other second languages should be given higher priority than target hypotheses that do not show any of these patterns. The factors described to rank target hypothesis paying attention to the learner, the language, and the task are not exclusive and ideally all combined into a ranking mechanism. The highest ranked target hypothesis is selected by jointly making use of these factors.

Selection of a pedagogical strategy for presenting feedback

After having prioritized target hypotheses, the divergences between the target hypothesis/target answers and the learner production should be presented to the learner, i.e. the actual feedback should be provided to the learner. In order to display the feedback, a range of pedagogical considerations need to be made. While human teachers will make many of these decisions by intuition, for computer-based tutoring systems it is necessary to explicitly code and parametrize the decision points. For this reason, we will discuss each individual decision in the following by going over the points listed in Table 4.1.

learner answer correct	consideration for pedagogical strategy
true	1) absence of negative feedback 2) reinforcement
false	3) modality of signal for feedback 4) specificity of feedback message 5) formulation of feedback 6) marking of location of error 7) time of presentation of feedback 8) location of display of feedback 9) learner interaction history

Table 4.1: Considerations in displaying feedback

In the easiest case, the learner answer is identical to the target answer, or similar enough to be accepted by the system. This means that in the target hypothesis, there are no significant changes. What constitutes a significant change is dependent on the task: while for form-based activities a tense formation error might be a criterion to reject a learner answer, for meaning-based tasks like a listening comprehension exercise this might be an insignificant divergence as long as the semantics are correct. As discussed in section 4.2.1, form and meaning are not independent or separable and therefore algorithms that check the correctness of learner answers need to consider both simultaneously.

If an answer is recognized as correct, there are two options: in the easiest case, no error or feedback message is shown. This distinguishes this case from the instances when a learner makes an error and gets shown feedback. In a more elaborate case, a learner receives a positive reinforcement (e.g. Lyster and Ranta (1997)). This reinforcement can have different forms like for example a green check mark or a message praising the learner for having produced a correct form like "Well done". The motivation for the reinforcement is to psychologically encourage the learner to attempt to produce a correct answer again next time.

In the more complex case, a learner answer is not accepted as correct by the system. In this case, there is some divergence in the learner answer worth pointing out that the system should provide feedback for. The first consideration is the modality of the signal for feedback. In a dialogue-based system with spoken communication, a feedback in the form of audio might be more

appropriate than a written feedback. Related to this, system designers can decide whether or not to associate feedback with certain audiovisual signals like a specific sound playing or a symbol in a specific color, e.g. red, popping up. The second and third point, the specificity of the feedback and its formulation, are closely related. The most unspecific way of presenting feedback is to just use a symbol or sound to indicate that there is some deviation in the learner answer.

Feedback can be made more specific by marking the location of an error (next point in table) and by providing an explanation of an error. This explanation is the point where the formulation of the feedback can fine-tune different degrees of specificity of the feedback. Heift (2010a) conducted experiments in a tutoring system to measure the effect of highlighting only versus explanation. They showed that more explicit feedback led to more uptake (i.e. successful corrections) by learners, but also that more specific feedback was more beneficial for learners with higher proficiency. Going into more specific feedback, the formulation of feedback becomes important. Lyster and Ranta (1997) distinguish five types of formulations with respect to corrective feedback. The first type, named explicit correction, points the learners to the error and directly provides them with the correct form. This type of feedback that gives away the correct form has been shown to be ineffective (e.g. Panova and Lyster (2002)). The second type is a recast that is a "reformulation of all or parts of a student's utterance, minus the error" (Lyster and Ranta, 1997, page 46). Often tutors also change their intonation to make the location and nature of the correction more salient to the learner. Another type of feedback formulation is a clarification request. In this case, a teacher or the system points out that a part of the learner answer is not correct and asks the learner to clarify or correct this point.

The fourth type of feedback presented, elicitation, is closely related to a clarification request. In elicitations, tutors repeat a part of the learner answer, but make a pause and indicate to learners that they have to provide a corrected form to fill this gap. Finally the most specific type of feedback is meta-linguistic feedback. The aspect of linguistics included is due to the domain of language learning this paper originated from. From a more general point of view, this type of feedback is concerned with the provision of domain knowledge explaining to the learner why, in comparison to the model of the learning domain, the answer is incorrect, without providing a correct answer to the learner. This type of feedback has been shown to be very effective for learning (e.g. Heift (2010a); Meurers et al. (2019a)).

The formulation of the error is also related to the point termed "learner interaction history" from Table 4.1. If there is an interaction history of the learner (cf. section 4.2.4) and the learner has made a specific error before and been shown a certain feedback message before, then this history can be used to adapt the specificity of the formulation. If a learner makes the same error or an error of the same nature again, potentially in a specific time interval or the same exercise, then the formulation can be adapted, for example to make it more specific. Another example of how the interaction history can be used is to adapt the feedback based on whether a learner has previously shown

to have mastered a specific form. In this case the feedback formulation can differ depending on whether the system diagnoses it as a competence error or a performance error. In the latter case, the learner in principle has acquired the correct form or construction, but made a spurious error, e.g. via distraction, whereas in the former case, the learner has not learned or sufficiently practiced the concept needed to be successful for a form or construction of this type. A further consideration is the time of presentation of feedback. Conceptually two types can be distinguished. Formative feedback describes an approach in which feedback is provided while the learner is working on an exercise.

Formative here means that the process of practice is formed and scaffolded by feedback that is presented directly while a learner is working on an exercise. In contrast, summative feedback is a type of feedback provided after an exercise has been submitted by a learner. Summative means that the performance of the learner is summed up. An example of a type of summative feedback is a grade assigned to an essay. Especially relevant for formative feedback is the next parameter, the location of display of feedback. While this at first glance might seem more like a user interface consideration, it does have implications relevant for learning.

Two approaches can be distinguished with respect to this parameter. The first approach is that feedback is displayed at some location outside of the actual exercise data, for example at the bottom of the page. Alternatively, feedback can be displayed next to or inside the input field (or in the case of paper, on the same line on the sheet). Computer-based tutors also offer further options like displaying the feedback in a popup window anywhere on the screen. The positioning of the feedback message can have implications on the salience of the feedback and thus the probability of it being read and implemented by learners.

4.2.4 Recording of Interaction with Feedback in Learner and Task Models

In the previous section we established what parameters can be tweaked with respect to feedback presentation. The combination of these parameters and the effectiveness of certain parameter configurations can be stored for each student and task. Every time feedback is generated and displayed by the tutoring system, this event, together with the reaction of the learner to whom the feedback was shown, can be recorded. This interaction data serves as the basis for subsequent Learning Analytics procedures and is typically recorded in the form of time-stamped log entries (Romero and Ventura, 2007).

In chapter 3 we explain in detail different types of Learning Analytics, and in chapter 2 more about different types of learner reactions to corrective feedback. In this section, we focus on the immediate application of the records of learner-feedback interaction in the process of providing feedback in tutoring systems. From a general standpoint, the interaction data can be stored in or associated with both a model of the exercise and a model of the learner, contributing to task and learner models.

Regarding step (1) in Figure 4.2, the linguistic analysis of learner productions, the first usage of Learning Analytics can be an optimization of parsers for learner language for individual tasks, users, or combinations thereof. If the Learning Analytics data shows that certain rules are applied more frequently, and/or if the data shows that analyses with certain rules more often lead to corrections by learners, then these rules can be ranked higher so that they are applied before other rules in order to arrive at a linguistic analysis more quickly. This is especially useful in pattern-matching approaches that stop after one error pattern has been detected. In linguistic analysis frameworks using probabilities or weights for individual rules, the weights can be adjusted based on frequency and success metrics.

This weighting is at the boundary between step (1) step (2), the selection and representation of target hypotheses. If there are ties in the ranks of target hypothesis, or also to select more likely target hypothesis by adjusting their weights, learner and task models can be used in the target hypothesis ranking process. The first language of the learner can be explicitly encoded in the learner model. Rules also have to be labeled with information regarding their frequency of application for learners with a specific first language. These labels can be assigned top-down with knowledge about different language families, or, as illustrated in Figure 4.2 in a data-driven bottom-up approach. The combination of labels for rules and learners can rank target hypotheses higher for cases where there is a match in these parameters. Furthermore, target hypotheses that prioritize constructs that the learner is learning at the moment, but has not yet mastered, can be ranked higher, i.e. constructs in the Zone of Proximal Development (Vygotsky, 1978). The decision as to which construct has been learned already can come from the Learning Analytics records, more specifically from the learner model.

Regarding the third step, the presentation of the interpretation of a target hypothesis, interaction data can reveal which presentational parameters have shown to be most effective for specific learners. Research on individual differences in second language acquisition has shown that learners with different learning styles exist (Cronbach and Snow, 1977; Corbett and Smith, 1984; Plass et al., 1998). Visual learners prefer feedback and explanations in visual mode, such as in images or videos explaining concepts or feedback. Auditory learners learn best with feedback and explanations in audio form, and verbal learners favor feedback in static written form. The parameters described in section 4.2.3 allow to adapt the tutoring system behavior to accommodate for these presentation styles. If in the interaction data the system stores information about the modality of presentation, and if the system varies the presentation format, or if the learner indicates a certain specific learning style preference, then this data can be used to select which modality of presentation is most effective and apply this strategy for this learner.

4.3 Non-Language Tutoring Systems

4.3.1 Overview and Definition

Non-language tutoring systems, in the context of this thesis, describes tutoring systems whose primary learning subject is not a second language. Concretely, the goal of these systems is not to teach a language first and foremost.

This delineation is necessary since tutoring systems of this type still employ language in various ways, and language often plays an important role. For example, most tutoring systems provide task instructions in textual form. Understanding the instruction and thus language is key to solving exercises. In the MathSpring system (cf. section 4.3.3), learners are required to first select which type of activity a task constitutes before being allowed to work on the actual exercise. With this step, the developers aim at ensuring that the learners can understand the language and textual cues for specific types of exercises. The primary goal of MathSpring is still to teach mathematics, but (specific) language still constitutes a part of the learning domain.

Relatedly, exercises always require some form of response of learners. While there is the option to allow input only via pre-defined options or only in mathematical terms, as soon as understanding should be tested without an easy way of guessing, learners have to justify their reasoning via language. An example for this is ASSISTments (cf. section 4.3.2). The primary goal of this tutoring system is to teach Mathematics. However, in addition to providing mathematical solutions, in certain tasks learners have to justify their answer in textual form, and the system attempts to automatically score learner responses (see Erickson et al. (2020)).

Even for tasks where there is non-textual input, non-language tutoring systems provide sophisticated answer checking mechanisms that are in principle transferable to a language learning context. Systems such as Andes (cf. section 4.3.4) are cognitive tutors which model rules in a domain (in this example physics) explicitly, are able to combine rules, and thereby solve problems based on these rules independently. Since language tutoring systems (cf. section 4.4) are often also based on production rules for language, the approach is compatible.

In the remainder of this section, we discuss examples of non-language tutoring systems mainly for mathematics and physics, i.e. well-defined domains, since these systems offer interesting aspects apart from language processing. These aspects comprise user model designs, domain models, exercise difficulty estimations, the employment of tutoring system in studies, among other aspects, and are in general transferable to language tutoring systems.

4.3.2 ASSISTments

ASSISTments (Heffernan and Heffernan, 2014) is a tutoring system platform originally designed for mathematics. It is used mostly in the domain of mathematics, but has also been extended to other domains, including English.

The term tutoring system platform in this case means that the system has been designed with flexibility in mind in the sense that the system can be instantiated for different domains. As Heffernan and Heffernan (2014) explain, a platform can be used to create systems for a specific purpose. A wide range of studies testing different parameters and contents with the system have been conducted, also in large-scale contexts (see e.g. Kelly et al. (2013)). For studies, an established design consists of a pretest and post test to measure the effect of an intervention. However, traditionally pre-post test designs costly in preparation and administration for both researchers and teachers. In order to facilitate such study setups, the developers of ASSISTments allow researchers and teachers to design and run randomized controlled field trials with different experimental groups, pretests, and post tests in the system within the normal usage instead of as extra activities to be done outside.

A key principle in the design of the system is to put the teacher in charge: they can create their own materials, hard code the sequencing for exercises depending on their students' performance, and have reports generated with performance indicators of their students. The last functionality is achieved with a form Learning Analytics tool for teachers that shows metrics such as the percentage of correct answers, but also frequent errors by learners as a means of supporting the preparation of school class lessons. In summary, the authors argue against a too strong integration of automatic methods, but more a design where the teacher is always in control for keeping them in the loop and control.

It is important to note that teachers can give away some control though, but they need to actively take action. If they chose to do so, they can assign their students to a round of mastery learning for a specific skill. The idea is that the system only lets students move to another topic once it recognized that they have mastered the current topic. The system performs this kind of gating by giving new tasks to students until a mastery criterion is reached. The default case in ASSISTments is to achieve a direct sequence of three correct items. Additionally, the system offers the option for spaced repetition in which the system repeats a certain topic/skill in a fixed interval span.

Another way in which teachers can give away control is via a form of adaptive sequencing in a variant of the system called PLACEments. This system explicitly models a skill hierarchy and uses it for an automatic suggestion of exercises based on skills associated with exercises. When a learner doesn't answer an item correctly, the system recommends exercises for the respective skill.

Another feature worth mentioning is the parent notification functionality. Parents can subscribe to notifications about the progress in terms of exercises and homework. This feature has been shown to increase homework completion rates.

In terms of task types, the system offers multiple choice, selection, ranking, and free text input. It does not incorporate NLP, therefore the free text input needs to be graded by teachers or students via peer feedback. For cases where the system provides feedback, in the mathematics domain, it implements a model tracing approach, but without a very specific student model or advanced deduction. As the authors put it, "ASSISTments does not know how to solve

the problem” (Heffernan and Heffernan, 2014, p. 3). Earlier versions of the system were more focused on production rules and allowing users to extend the grammar the system uses. This however proved to be too time-intensive and required too much expert knowledge for users, therefore there was a shift to write examples and solutions instead of production rules. The reasoning here is that with enough examples, the output of a model tracing approach could be imitated closely enough. In order to get examples and information about teaching strategies, the developers also recorded teachers in their classroom to learn which pedagogical strategies they use.

Interestingly, the system’s capabilities were extended in a way that the system can be seen as a hybrid between a language and a non-language tutoring system. In certain tasks, learners need to provide an answer in mathematical notation, and justify their reasoning in textual form. ASSISTments then conducts short answer assessment on these semi open tasks and attempts to automatically classify the answers in terms of their correctness for this task (Erickson et al., 2020).

In 2014, the developers reported 4,000 concurrent users every day (Heffernan and Heffernan, 2014) of the platform/system, which, in contrast to many other system, is tightly integrated in the school context in the United States and 13 other countries in 2020³. The contents of various state-approved text books have been adapted for use in the system. The data from the system has been and is used in data mining competitions, like for example the *ASSISTments Data Mining Competition 2017* (Patikorn et al., 2018).

4.3.3 MathSpring/Wayang Outpost

MathSpring, formerly known as Wayang Outpost Arroyo et al. (2014), is a tutoring system for mathematics. Based on a high variation between students in terms of motivation and knowledge in mathematics, as well as different degrees of aptitude, the authors argue for a system that is adaptive on an affective, meta-cognitive, and cognitive level.

In this paper it is explained that there is a correlation between memory-related capacities and mathematical abilities. Difficulties with self-regulation and affective experiences play a big role as well (e.g. implicit motivation for the subject).

Key features of the system to act upon are a model of affective states, a model of self-regulation, and an adaptive exercise sequencing model. In combination, these components allow the system to select a suitable pedagogical strategy.

Similar to what is described in section 4.3.2, MathSpring follows an example-based approach, drawing partly on a pool of items from standardized tests. Arroyo et al. (2014) compare this approach to model tracing approaches and argue that this approach takes additional affective and meta-cognitive factors into account and models exercise difficulty not only in terms of correct or incorrect attempts like in Item Response Theory, but also includes and considers time

³<https://new.assistments.org/>, last accessed 2020-03-17

and feedback behavior. The result is a continuous update of a range of diverse difficulty metrics for exercises in the system.

When working on an exercise, learners first have to identify which type of problem it is they are working on (i.e. from which sub area of mathematics) in order to help them choose an appropriate strategy. The authors argue that students often show difficulties in determining what kind of problem they are working on, but once the type of problem has been decided, following a solution strategy is less difficult. While learners are working on an actual exercise, the tutoring system offers scaffolding feedback with different modalities of feedback: explanations of errors with different media, highlighting, enhancements via drawings, sound, examples of solutions, and steps towards solutions for similar problems.

The system provides an adaptive sequencing of materials with the goal to keep learners in the Zone of Proximal Development, i.e. make the exercises not too easy or too difficult. The number of exercises given to a learner is not fixed, but rather determined by when a certain mastery criterion has been reached. The idea is to give harder exercises when learners show a good performance, and sequence easier tasks when learners struggle. It is worth mentioning that teachers are still in the loop: they can limit the domain in which learners should practice, and additionally limit the time. The sequencing algorithm considers time, feedback, and trials.

Different scenarios are distinguished with different sequencing criteria: exercises that are too easy or too hard, students putting in low effort trying to game the system, exercises at the right difficulty, and students avoiding hints provided. In essence, the system puts effort, hints and success into relation with each other. By doing this, they link behavior to diagnoses and provide meta-cognitive strategies. The item difficulty parameters are estimations based on past trials, concretely an expectancy value computation based on a statistical region of parameters such as hint taking or time-on-task of past attempts. A concrete example is the situation of mastery with high effort: in this case, the system recognizes a longer working time than the average student, but also that the learner did not request many hints. In this scenario, the system chooses the strategy to maintain the current difficulty of tasks to be sequenced, and to praise the learners for their effort.

Teachers are able to use Learning Analytics functions for visualizing student progress. They can look at the progress of their school class as well as individual students, and visualize the metrics on both a topic and an item level. These metrics include time-on-task, feedback behavior, engagement, motivation, and success criteria and serve the purpose of helping teachers to prepare their next classes.

For students, the system also offers Learning Analytics tools with the goal of increasing self-regulation and meta-cognition, both of which have been shown to have an impact on mathematics performance. Firstly, the system incorporates an open learner model where the progress is visualized for different topics. For each topic, it allows to continue working on it, reviewing past exercises, and requesting challenging new exercises. The learner model is not only pas-

sively waiting to be opened by learners, but the system contains affect detectors that open the learner model based on certain recognized affective states, more specifically transitions between affective states.

These affective states are detected via sensor-free affect detectors that can recognize confidence vs. anxiety, interest vs. boredom, frustration, and excitement (based on interaction log patterns). The system contains animated figures that display emotions and encourage students to work on the problems, motivate and encourage them, and ignore them if the system classifies the student behavior as not exerting effort. These characters showed a high positive effect especially for initially lower performing students, with better results when the gender of the scaffolding character matched the gender of the learner.

Secondly, the system offers progress screens after every six completed items. Based on the learner's performance, these screens show different hints and visualizations, corresponding to different pedagogical strategies.

MathSpring and its predecessor Wayang Outpost are in use since 2004, and their effectiveness has been demonstrated in various studies in both pretest/post test designs and in delayed standardized tests showing significantly better results for students using the system.

4.3.4 Andes

Andes (Vanlehn et al., 2005) is a cognitive tutor for physics. This type of tutoring system contains explicit domain and problem solving knowledge enabling it to automatically solve the exercises included in it.

Designed as a homework helper, it assists learners in physics education on the topics of mechanics, magnetism, and electricity. In the learning of these topics the demonstration of the steps towards the target answer, i.e. the derivations, has higher priority for teachers, and not only the final result. For this reason, Andes was designed for formative scaffolding feedback on individual derivations steps of exercises.

Learners can write equations freely in input boxes of exercises with one input box for every step in a derivation. The authors however acknowledge that the required input format is more precise than in regular interaction in class.

Given a learner answer, the system first conducts a binary correctness check. Correctness in this case has two notions: a learner answer is correct if either logically follows from the task, or is a valid part of the system's pre-computed derivation towards the target answer. Once binary feedback has been provided, learners can receive detailed specific feedback, but only when they actively request it after an answer has already been flagged as incorrect in the binary feedback step.

Upon requesting specific feedback, the rest of the interface gets disabled while feedback is shown in the system in order to focus attention on the message content. A message consists of a sequence of three hints with increasing specificity per feedback message, with the last part of the message telling learners explicitly what they need to do in the given situation. This hint is however not shown in the beginning; instead learners can, after each hint, request more

(specific) hints, or alternatively confirm to return to the exercise to continue working on it.

The rules underlying the distinction specific feedback distinguish between errors and mistakes so that conceptual errors are prioritized over spurious mistakes. Each rule or pattern encodes whether it models an error or mistake, thus offering a rule-based approach to this problem. These error patterns and/or mal-rules explicitly offer conceptual feedback. Earlier versions of the system came with a matching of learner answers to target answers (or their equivalence in terms of derivation step equations), but the system did not provide conceptual feedback, only syntactic feedback. This approach leading to surface corrections without a deeper understanding fostered a switch to error pattern rules with scaffolding information.

In terms of handling ambiguities, i.e. in cases where the system does not understand an answer, it asks learners to provide the principles they intended to apply so that the system can narrow down potential problems. Additionally, Andes requires learners to provide formal definitions of variables via a drop down menu (thus offering the system an unambiguous denomination). Making the connection to language tutoring systems, this solves problems inherent in language tutoring systems where the system needs to decide on a target hypothesis of an incorrect learner answer. Selecting an explicit variable definition is comparable to a selection of parts of speech of words by a learner.

Additionally to the feedback on a wrong answer, learners can request a conceptually different type of feedback. This type of feedback describes the next steps learners should do for this task and does not point out properties of a specific learner answer, but instead is guided by the task and pedagogical considerations. This type of feedback takes into account what has been entered already, what typical steps are to be taken at this stage in general for this type of exercise, and what the next target steps are for this concrete task. Different from feedback that points out what is wrong in a specific step, this feedback type informs learners about what steps to take next.

Another type of assistance to learners is an equation solving tool that allows to automatically solve equations by variables in order to let learners focus on the conceptual understanding of physics rather than algebraic operations. After having used the correct formulae, this tool can, as a last step, assist in computing the concrete target quantity.

Andes computes a score based on the correct attempts and frequency of requested specific feedback. The score is increased with correct attempts and not subtracted when learners request the least specific feedback, but only when they request the most specific feedback telling them exactly the next step to pursue.

This tutoring system can be categorized as a cognitive tutor, i.e. a type of system that is able to solve problems from its domain. In the associated authoring tool, authors thus provide exercises, but no answers. Andes attempts to solve them, computes a solution, and shows authors the steps involved towards the solution. Underlyingly, the system uses 550 rules underlying the exercises/computations/principles. These rules are used for a pre-generation of

derivations of all problems before the system is deployed (offline generation approach, cf. also section 6.2). Generated from that is are output files representing directed graphs of the space and sequence of steps for reaching a solution.

The effectiveness of Andes with its approach to emphasize the importance of and apply the fundamental principles underlying a problem has been confirmed in multiple longitudinal studies in the United States Naval Academy.

4.4 Language Tutoring Systems

4.4.1 Overview and Definition

A language tutoring system is a form of tutoring system in which language is the main pedagogical content. 'Main' pedagogical content because language is used to communicate meaning, and, especially in the school context, the introduction of new language structures is often associated with cultural studies of the countries in which the foreign language is spoken. While this constitutes an aspect, the focus of system designers for language tutoring is on language as a system and its linguistic properties.

Meurers (2012) distinguishes between language tutoring systems that analyze the language of learners in the target language, and systems that analyze utterances of native speakers in the target language for language learners. The first type of systems typically provides feedback to learner productions (e.g. E-tutor, cf. section 4.4.2), whereas the second type often performs input enhancing for learners (e.g. View/Werti, cf. section 4.4.8).

Another relevant distinction for language tutoring systems is the target user group addressed by the system designers. Systems like FLAIR (Chinkina et al., 2016) support language teachers in selecting relevant material for their students, while tutoring systems like TAGARELA (cf. section 4.4.3) are designed to support students while working on exercises.

In general, the number of language tutoring systems in comparison to tutoring systems in other, well-defined domains, is smaller given that on the one hand free input interaction by learners is desirable, but on the other hand the interpretation of input that is not fully constrained is very difficult due to ambiguity inherent in natural language.

In the following, we discuss a range of language tutoring systems that illustrate different approaches towards building and using language tutoring systems. E-tutor (section 4.4.2) and TAGARELA (section 4.4.3) illustrate the concept of feedback and an architecture supporting it for language tutoring systems. iTutor (section 4.4.4) is interesting since it illustrates how tutoring systems can be integrated into authentic school contexts. ROBO-SENSEI (section 4.4.5) serves as an example of how to generate feedback for natural language tasks. CASTLE (section 4.4.6) is discussed because it shows how Learning Analytics can be combined with language tutoring systems into learner models. Finally, View/Werti (section 4.4.8) demonstrates how language tutoring systems can be used to enhance native language for learners of this language.

4.4.2 E-tutor

E-tutor is a web-based tutoring system for learning German as a second language (Heift, 2010b). Originally, it was called German Tutor (Heift, 1998) and iteratively developed and improved since 1999. The typical use case of e-tutor consists of the system as a supplement to a regular workbook combined with traditional classroom instruction. In the application, content is organized according to chapters and mirrors the content of three subsequent introductory courses of German. Each chapter contains both exercises and supplementary content. The exercises have one of nine types ranging from vocabulary exercises to essay writing tasks. In the background, there is one common analysis module for all task types, but different pedagogical module instances for specific task types, allowing to re-rank feedback according to exercise requirements. The learner progress is saved between sessions.

The system offers immediate feedback on learner productions with respect to spelling and grammar. In addition to displaying meta-linguistic feedback on errors in the learner production, the location of an error is highlighted. Only one error at a time is displayed. Every time a feedback message is displayed, this action is stored in log record and can be accessed by the learner while working on a n exercise. Learners furthermore have the option to show the solution for an exercise (Heift and Rimrott, 2008).

The log data is the basis for a learner model to individualize pedagogical interventions, provide progress reports to learners and instructors, and alert learners if exercises are probably too difficult. Additionally, the system compiles learner corpora that can be searched for examples via an interface in the system, forming the basis for an analysis of task difficulty based on performance of learners recorded in the system. E-Tutor has been used in a range of studies investigating design choices and learner behavior in ICALL systems. For example, Heift (2002) identified four types of learners based on their interaction behavior with the feedback and system functionality: browsers, peekers (frequent and sporadic), and adamant learners. Furthermore, the system has been used to investigate the effect of different types of corrective feedback and the reactions of learners towards it (cf. section 2.3).

4.4.3 TAGARELA

The web-based tutoring system TAGARELA (Teaching Aid for Grammatical Awareness, Recognition and Enhancement of Linguistic Abilities) (Amaral and Meurers, 2008, 2011) is an electronic workbook that provides interactive feedback for learners of Portuguese. Developed since 2005, it contains a range of exercises that are independent of a specific printed workbook. Its intended usage is as a supplement in traditional classroom teaching. In the system, there are six exercise types with different contexts: listening comprehension, reading comprehension, vocabulary, reformulation, and picture description. The expected input ranges from individual words in, for example, fill-in-the-blanks exercises to single sentences in short answer tasks. TAGARELA is capable of provid-

ing feedback to spelling, grammar, morphology, and semantics, thereby going beyond e-tutor in terms of feedback variability. One notable characteristic of TAGARELA is its activity model. For each activity, it stores information about the difficulty level, length of expected learner production, task type corresponding to how much manipulation of the content is necessary by the learner, and required strategies to accomplish the task. Interactions with the system are recorded and stored in a (closed) learner model where the learner competence is recorded by type (form or meaning), task type, required task strategies, and transfer from the first language.

4.4.4 iTutor

The iTutor system (Choi, 2016) is an ICALL system that is designed to support Korean learners of English. In use since 2016, the system has been tested in elementary, middle, and high schools. The electronic workbook consists of stories in Korean followed by fill-in-the-blanks translation tasks. The base forms of the (translated) words expected in the blanks are provided, but the gaps were selected in a way that requires learners to inflect the words in certain linguistic forms, thus eliciting and testing certain grammatical constructions. While learners are working on a gap, the system checks the learner production in a multi-stage process. If a learner answer is incorrect in the first attempt, the system highlights the part of the learner answer that is not expected. This is referred to as “elicitation” (Choi, 2016, p.342). If the answer is still incorrect in the following attempt, the system provides highlighting again and additionally displays meta-linguistic corrective feedback regarding the learner answer. If the correct answer is still not produced by the learner in the next step, it is displayed in the gap. The feedback provided by the system, as categorized by Choi (2016), falls either into the category of syntactical errors (misordering of lexical elements) or lexical errors (malformation, addition, omission of words).

4.4.5 ROBO-SENSEI

ROBO-SENSEI: Personal Japanese Tutor (Nagata, 2009) is a language tutoring system for learners of Japanese. It covers grammatical structures of the first three years of Japanese learning by using NLP analyses for enabling scaffolding feedback.

The motivation for the system is to deal with a combinatorial explosion for word forms that already becomes evident for well-formed variants. Since language is a creative system, for a simple sentence, already many thousands of well-formed variants are possible. Since interlanguage introduces non-standard variants of forms, even more forms are possible. Nagata (2009) argues that this abundance of form requires automatic methods to be able to deal with the variability.

As a mechanism for providing feedback, ROBO-SENSEI makes use of manually constructed answer templates that are automatically compiled out to natural language strings and stored as target answers with some linguistic analyses,

primarily token segmentation boundaries. Further analyses are then computed in the live system, analogous to the processing of the student answer.

The system generates fine-grained annotations on the token, morphology, syntactical, and semantic role level. For segmentation, the system searches the longest common sub string segmentation of tokens and chunks, augmented with a constraint-based approach including grammatical knowledge about which constellations are possible or impossible. For every token, the system adds morphological analyses by looking up the corresponding entry in the system lexicon. The next step consists of parsing: by using 14 context-free rules in a bottom-up parsing method, the system creates a constituency parse tree. This parser includes a feedback module that generates a feedback message based on divergences of the linguistic analyses of the learner and target answer(s). This is conducted by reducing answer schemata, basically by selecting the target answer (generated from the manually constructed answer templates) that is most closely related to the learner answer, i.e. most constraints are satisfied (e.g. presence of a certain word), and comparing its analysis to the analysis of the learner answer. The system provides meta-linguistic feedback on different levels of analysis.

First of all, it detects words in the student answer that are unknown, i.e. not in the system lexicon. Not all words are simply treated as out-of-vocabulary items: the algorithm includes a rule-based handling of common spelling error patterns by learners and then gives specific feedback pointing towards them.

Secondly, the system detects missing words: if a word is present in the target answer, but not in the learner answer, then the learner answer is flagged with a missing word error.

Inversely, the system also recognizes unexpected/extra words that are not present in a target answer. In addition to the simple checks, the system includes a rule-based handling of special cases, e.g. confusion of different types of appellations in Japanese.

Based primarily on the morphological annotations, the system furthermore gives feedback on predicate form errors regarding tense, negation, auxiliaries, and style. The semantic role annotations are used to give feedback on semantic errors, when learners for example confused the subject and object of a sentence. A final category of feedback consists of the detection of word order problems.

Since the system generates a parse forest based on the learner answer, and since for every learner answer potentially a different target answer gets selected, the system needs to rank the analyses for displaying feedback on the highest ranked learner-target answer pair. This ranking is conducted by selecting the analysis with the fewest number of detected errors in its analysis. As a means of improving the system over time, the author talks about a mechanism in which erroneous analyses can be flagged and then are then excluded in the live system in subsequent parsing approaches.

Once the learner answer is fully analyzed and compared, the system concatenates the feedback messages of all errors into one error message (which can potentially become very long) and displays it. The textual representation of the feedback messages is created by using feedback templates with slots. These

slots are filled with relevant sub strings from the learner answer in the message template and the all the triggered templates are concatenated into one feedback message text.

4.4.6 CASTLE

The CASTLE system (Murphy and McTear, 1997) is a language tutoring system for English.

It contains both grammar and function-oriented tasks (operationalized via role play tasks for learners in different scenarios). The system contains exercises where the base words to fill in are provided, but with the system allowing free-text input.

As a first strategy, CASTLE suggests exercises with a functional goal. If a learner encounters problems with them and makes errors, the system is able to recognize the points where the learner had problems and suggests grammar-oriented tasks focusing on these points.

The system's architecture consists of several modules that interact with each other. In the context model, the system stores session-specific information informed by the other modules. The diagnosis module is responsible for analyses and feedback generation. This is done via a chart parser for creating analyses, and error rules that model specific error patterns. Errors in CASTLE are organized in a hierarchical data structure.

The job of the tutoring module is to adapt exercises and decide on pedagogical strategies most suitable for the learner in terms of feedback type and exercises.

The domain model follows an ontology-like approach organizing the constructs to be learned in a hierarchy. Top-level categories in the domain model are simple present, simple past, present progressive, and present perfect. This is important because the learner model builds upon the domain module and thus implicitly also has information about a hierarchy of constructs.

More specifically, the learner model consists of a profile (background, name, motivation etc.), a model of the estimated proficiency of different grammar constructs, and a 'cognitive' model (storing feedback preferences, exercise preferences, and the student's interest in grammar).

In the learner model, the system keeps track of four levels of proficiency for every category. In the beginning, where there are not enough data points for a reliable learner model, the system uses a stereotype-based approach to assign learners to proficiency levels based on a pretest and on stereotypes encoded in the system. The values in the learner model are initialized according to the closest stereotype. Stereotypes indicate the proficiency in certain parts of the domain model and are labelled with four levels of expertise (novice, beginner, intermediate, advanced). The score for each grammar construct is based on the performance in items relating to this construct and the number of feedback messages needed for reaching a correct answer.

Worth mentioning is the system's implicit acquisition rule mechanism: when a learner successfully completes an exercises that tests a more complex topic that

has a set of simpler prerequisite constructs, the system also updates the learner model proficiency levels for these simpler prerequisite items. Furthermore it keeps track of an error proneness measure that models the number of errors committed divided by the number of questions answered.

By interacting with each other, the individual models allow for adaptivity. CASTLE implements a constraint-based approach towards sequencing materials by considering both explicit prerequisites in terms of constructs that are necessary to learn, and also taking pedagogical considerations of what is normally taught first into account. After three errors of one category (with errors organized into categories), the system sequences a grammar exercise focused on this specific category the learner has problems with. While learners also have the option to choose exercises, the system warns them if the selected exercises are estimated to be too easy or complex given the current proficiency.

4.4.7 ICICLE

ICICLE (Interactive Computer Identification and Correction of Language Errors) (Michaud and McCoy, 2000, 2006) is a language tutoring system for English as a second language. The target group consists of learners that are deaf and use sign language (American Sign Language) as their primary language. The authors explain that in sign languages, different features are relevant in the communication compared to standard language, for example the use of facial expressions.

The tutoring system offers diagnostic capabilities paired with a pedagogical model in order to react appropriately to diagnostic information, with a user model central to both processes. The use of this user model is twofold: interpreting learner language and sequencing material.

A component called SLALOM (Steps of Language Acquisition in a Layered Organization Model) forms the core of the user model, with each grammar point in the learner model marked with a value from a tripartite value space: acquired, not acquired, and in the process of being acquired. The learner model is implemented as an overlay over the system's domain model, in which grammar points are ordered by typical sequences of acquisition in different areas, but with connecting layers across these areas that mark features from different areas that are acquired concurrently, e.g. from the morphology and sentence structure area. These layers in the domain model, which can be conceptualized as stereotypes, are used for the initialization of the learner model, i.e. to overcome the cold start problem, but later actual evidence based on the learner's productions is used to override the initial stereotype-based values. This approach allows to initialize the learner model even with incomplete data in the beginning.

Based on this model of acquired or partly acquired grammar points, ICICLE is able to adapt the interpretation of language productions for individual learners. In order to provide meta-linguistic feedback for written production together with a highlighting of errors, the system needs to find the most plausible interpretation of what the learner wrote. This is achieved by combining a set of standard grammar rules with a set of mal-rules modeling patterns of

misconceptions. The grammar used in the parser spans over 300 context-free rules.

When parsing a production by a learner, standard grammar rules are used for features that have been marked as acquired by this user, mal-rules are used for unacquired phenomena, and both types of rules for grammar points in the Zone of Proximal Development (since learners experiment with forms and features currently in the Zone of Proximal Development). By implementing these dimensions, ICICLE is able to distinguish between competence errors and (accidental) mistakes of already acquired concepts, and prioritizes feedback on actual errors. Incorporating the aforementioned pedagogical information about what is teachable at a certain stage, i.e. in a typical acquisition sequence, this allows ICICLE to scaffold on and sequence exercises on those points with errors that are in the Zone of Proximal Development rather than on points that are normally acquired and taught later or have been marked as already acquired.

4.4.8 View/Werti

WERTi (Working with English Real Texts interactively) (Meurers et al., 2010; Ziegler et al., 2017) is a visual input enhancement system for learners of English as a second language. Implemented as a browser plugin, it allows learners to use any web page as input to the system. A benefit of this feature is that the material selection can be tailored by the learners based on their own interests.

The English version of the system models a range of constructs on a lexical as well as syntactic level, such as determiners and prepositions, conditionals, gerunds, questions, and phrasal verbs. In order to identify these constructs, WERTi builds on a UIMA-based NLP pipeline to add multiple layers of annotation while preserving the original text. Internally, it combines a part-of-speech tagger with a constraint grammar to first get basic analyses and then search for specific patterns relating to constructs based on the previous annotations.

Once the annotations have been added, users can select between three task types: firstly, the system offers a highlighting of a specific construction on the web page. A feature worth emphasizing is that in the case of multi-word expressions, including discontinuous annotations, elements with different functions and linguistic categories are highlighted with different colors. Secondly, users have the option to work on selection activities where every word on the page is clickable, and users receive correct or false feedback depending on whether they clicked on a word relating to the selected target construction. Thirdly, WERTi offers fill-in-the-blanks activities with binary feedback based on a string identity comparison with the original form.

WERTi has been extended to support multiple languages into the Visual Input Enhancement of the Web (VIEW) system. One extension of the approach towards Russian, with a special emphasis on the morphology level, is described by Reynolds et al. (2014).

4.5 Usage and Role of Tutoring Systems in Authentic Contexts

There is a fundamental difference between the usage of tutoring systems in controlled, experimental, or even lab-based conditions and the usage of tutoring systems in authentic, open, ecologically valid contexts.

Essentially, a tutoring system is a tool that is designed for a certain task (to support learners in a specific domain). The difference between the conditions is that in very controlled settings, the interaction, i.e. the usage of the tool, is constrained to the design of the experiment. Metaphorically, and sticking to our analogy of tool usage, a very constrained lab-based usage is like an experiment where students are supposed to learn how to use a hammer, but their hand and arm are always guided by an experimenter. In more open experiments, students may get explanations and controlled first experiences of how to use the tools, but may then experiment themselves with different types of usage (in the metaphor e.g. using the hammer in new creative ways) and combination of the tools not intended by the experimenters.

One illustration of such a highly creative and innovative approach happened during our FeedBook study (cf. chapter 7): talking to a participating teacher we learned that he had invented a completely novel way of using the FeedBook: instead of letting students work individually with the system - as we had envisaged it - he created a game in which he would use a projector to show an instance of the FeedBook system to his school class on the wall. The class would be split into several groups to compete against each other. While the teacher moved from prompt to prompt in an exercise, the groups had to shout the solution when they thought they knew the answer to the current prompt. This allowed them to collect points, and the group with the most points won the game.

In the end, when tutoring systems are supposed to be adopted in a standard curriculum, they need to function and provide a tangible value in authentic or ecologically valid scenarios and under different use cases, and not only in tightly controlled experimental conditions that do not mirror the everyday usage, since not all users will and can be expected to use tutoring systems in a predefined way.

Arguably, tightly controlled experimental conditions are important for scientific inquiries into the design and effectiveness of components of a tutoring system when there are design choices with open answers (e.g. Heift (2002)). However, after certain design choices have been made - ideally based on controlled experiments - it is essential that the question of generalizability is addressed. Are the results replicable for different learner samples? Does the approach transfer to different contexts, e.g. from a lab to classrooms? What is the impact of teachers or learners with different backgrounds on the findings? In which different authentic contexts does the approach work in ow well?

A crucial aspect determining whether a tutoring system is used in real-life contexts is whether it addresses real-life needs of the involved stakeholders,

primarily students and teachers (cf. chapter 8 and 9 for a discussion of the needs analysis). As (Heift, 2010b, page 457) put it, "both learners and instructors must experience immediate benefits." If the adoption or usage of a tutoring system creates more problems than it offers benefits, the likelihood of it being adopted will be small.

There are a few examples of tutoring systems that have managed the transition from development and testing in controlled experiments to a larger scale adoption and integration into regular classrooms. The ASSISTments system (Heffernan and Heffernan, 2014) is a prime example for how tutoring systems can be adapted to the needs of the involved educational stakeholders (teachers, students, researchers) to foster a more widespread adoption. The system has been continuously developed for over two decades, with an increase in the amount of contents, a continuous feedback loop to teachers and students, and a range of studies that systematically tested new features that would later be integrated into the main system. Kelly et al. (2013) is an example of such a study, comparing feedback in web-based homework with a more traditional homework setting in which feedback would not be given immediately by the system. The study comprised 63 learners in middle school that needed to do regular class work in mathematics, both using ASSISTments, but one group receiving feedback and the ability to correct answers based on binary feedback. A post test revealed higher learning gains for the group with immediate feedback. McGuire et al. (2017) followed up on this research and designed an experiment using ASSISTments with different feedback conditions: image-based feedback, textual feedback, and binary correctness feedback. They found out that image-based feedback led to consistently lower results and completion rates than the other types of feedback, with binary feedback leading to the highest completion rates. In 2020, ASSISTments reported over 20,000 teachers using the system ⁴. While this is a positive example of how a tutoring system can be adopted by a mainstream user base, it is important to keep in mind that it required decades of funding, a modular code base that can be adapted by researchers to do both experiments and adapt the system to the needs of users, and the involvement of teachers in the system design and material creation. For many other projects working on tutoring systems, one of the listed conditions is not met and the tutoring system does not see continuous widespread adoption.

Arroyo et al. (2014) shed light on another relevant point related to an adoption of tutoring systems. In the discussion so far, the main focus was on how to address needs of teachers, developers, and researchers. In their article, Arroyo et al. (2014) emphasize that the tasks in their tutoring system (MathSpring) are tailored towards a preparation of learners for standardized tests. Especially for tutoring systems that are designed to support learners outside of the school for homework (cf. also VanLehn et al. (2005) for the Andes system), it is crucial that learners experience direct benefits like higher grades in examinations. Indirectly, this point also involves meeting the needs of parents: if they pay for a license allowing their children to use a tutoring system, the question of whether

⁴<https://new.assistments.org/>, last accessed 2020-05-18

they are willing to pay continuously depends on the effectiveness of the system.

The E-Tutor/German Tutor system is an example of a tutoring system that has been in use for over a decade (Heift, 2010b), but only in a relatively controlled, academic setting. Heift (2010b) points out that in many tutoring systems coming from an academic context, there are most of the time limited human resources for the technical development of tutoring systems, especially for cases where development should be continuous over a longer period of time to adapt to both needs of existing and new users and to keep up with the fast-paced world of application and especially web development.

Despite the positive examples outlined here, there is a general trend that most tutoring systems are developed as prototypes that remain in the prototype stage. They are used for testing certain features in some experiments, but not for a longer period of time (Heift and Schulze, 2007; Rudzewitz et al., 2017). For the systems that manage to build a large enough user base over time to be adopted broadly, a diverse team of researchers, developers, and pedagogues is necessary to ensure a demand-driven development.

Most systems that reach popularity are tutoring systems for well-defined domains where the processing of user input is associated with less interpretation. There are counter examples, like Duolingo⁵, that have become popular language learning applications. For the majority of tutoring systems that have seen a rise in popularity and users, there is however a considerable gap between what the systems offer and what research has established as effective and beneficial for learning (Meurers et al., 2018).

While tutoring systems can see a wider adoption, this requires significant resources on multiple level: first of all a system can not be used for a longer time if it does not provide enough materials for a longer period of time. Secondly, a system used by many users needs to be built on a scalable architecture to support multiple requests and concurrent backend processing. Thirdly, a system needs to be built in a modular fashion that allows to update or change certain components. Monolithic systems, especially in the fast-paced context of web technology, sooner or later run into the problem that the technology is outdated with the effect that for features present in other architectures, disproportional effort needs to be done to reverse-engineer features from there. In order to do this, a team of experts on each of the different areas is necessary, adding a financial dimension to the endeavor of bringing a tutoring system to wider adoption.

In essence, tutoring systems need to address real-life needs, provide enough materials, and be continually maintained, updated, improved, and tested if they are to be adopted in everyday teaching and learning contexts by a mainstream user base. The very small number of systems that manage this shows the difficulty of this endeavor.

⁵<https://www.duolingo.com/>, last accessed 2020-05-18

Part II

Towards an Empirical Basis

Chapter 5

FeedBook as an Instrument for Data Collection

5.1 Overview

The FeedBook is a tutoring system for English 7th grade. It is a multi-layered web application that represents the adaptation of a paper-based work book enhanced with scaffolding feedback and Learning Analytics. It is implemented as a platform-independent system in the Google Web Toolkit (GWT¹) and usable with any web browser supporting HTML5 and JavaScript.

Developed in the Transferprojekt 1 (T1) in the Sonderforschungsbereich (SFB) 833 of the University of Tübingen, it is based on the SFB project A4 "Comparing Meaning in Context", of which it transfers fundamental research insights into an authentic application scenario.

Special about this project is the fact that it constitutes a cooperation project with an application partner, in this case the Westermann Gruppe. The application partner provided a starting point with a work book already established and in use in schools, of which all materials used in the system were adapted. The application partner thus provided a link to the actual school context in which the system was evaluated.

The system was developed in three year project starting in September 2016, but development began already in August 2016 since a pilot was needed for the start of the school year in 2016. From a high-level perspective, the system was developed over time in mainly three stages, with each stage corresponding to one year of the project.

The goal of the first stage was to build a web application for displaying exercises to students, as close as possible to the printed version. What emerged from this was a fully-fledged modern online platform with user management, user roles, profile management, navigation, messaging system, and many more.

¹<http://www.gwtproject.org/>, last accessed 2020-05-11

The reason is that users will not use a system if it does not provide basic features that are established for modern web applications (Rudzewitz et al., 2017).

In the second stage, the focus was on implementing orthography, grammar, and meaning feedback, and to implement Learning Analytics functions such as an open learner model of task analytics views that make use of the data generated through the feedback. This phase constitutes the step of going from a system where teachers need to do all the correction work to a system where for most tasks learners are supported by the system and led to a correct solution, requiring the implementation of Learning Analytics tools that provide different stakeholders insight into the learning process.

In the third phase and year, a year-long study was run. The main focus was to ensure system stability, implementing and conducting pretests and post test, and to collect data.

It is very important to emphasize that the FeedBook was not developed by a single person, but by a team of researchers with different backgrounds and skill sets, and research assistants helping in the realization of certain aspects of the system. The system is the result of group work by a large set of involved people. Even though we describe it here, it is important to keep in mind that it is the result of group work and would have been impossible without the contribution of everyone who worked on it. The author of this thesis is merely a member of the team that collaboratively developed the system.

Another point to make explicit is that in the dissertation, the final state of the system is described, as used in the study (see section 7). In certain works that describe the system on the way towards the final stage, previous versions of the system in different stages were described. The system developed over time and the earlier articles do not always reflect the final stage.

Since there is no data collection without an instrument, in this chapter present the FeedBook as a system. We explicitly exclude both the feedback generation mechanism, since this aspect is described in chapter 6, and the Learning Analytics functions, which are described in chapters 8, 9, and 10.

The motivation for the work described in this section is to give users a system that they can use and that is useful to them. For teachers, they can save time and effort if the system grades student submission automatically. For students, they can learn while being supported with the system offering real-time feedback. In return for the benefits offered, we get data that we can analyze for scientific purposes and use for improving the system.

This chapter is structured in two conceptual parts: in section 5.2, we describe the system from the perspective of exercises. We explain how exercises are displayed, entered, represented and interacted with. In the second part in section 5.3, we describe essential system functions apart from exercises that are nevertheless important for running a system, such as user management, a messaging system, and student and teacher start pages.

Disclaimer: Specific sub parts of the contents of this chapter have been mentioned or referenced in Rudzewitz et al. (2017) and are cited accordingly in the running text of the respective sessions.

5.2 Exercises in FeedBook

Exercises constitute the central ingredient of the FeedBook. They allow students to practice and test their English skills. Having a system with exercises enables interaction and feedback both between students and the system, and between students and teachers (Rudzewitz et al., 2017). Exercises furthermore provide the basis for the work of material designers and researchers, since interaction with exercises creates interaction data that serves as input to Learning Analytics tools and Educational Data Mining applications.

Since all exercises in FeedBook are adapted from the paper-based and state-approved workbook *Camden Town Gymnasium 3* by Westermann Gruppe², several steps and stages with associated interfaces to the system are necessary to arrive at a system with workable exercises.

First of all, it is necessary that exercise data is entered into the system's data base in a structured way. In order to do this, an authoring tool (cf. section 5.2.3) built on top of a domain model (cf. section 5.2.2) is necessary to enter and edit exercises in the system. For entered exercises, an interface to display them to students is needed (cf. section 5.2.4), with different renderings dependent on different task types (cf. section 5.2.4). For exercises with interactive feedback, this includes components to display generated feedback to learners (cf. section 5.2.5). For submissions of students to exercises, the system requires a grading interface for teachers (cf. section 5.2.6), and an interface displaying the teacher feedback to students (cf. section 5.2.7). In the following sections, we will explain each of these components in detail.

5.2.1 From Paper to Digital Version

The FeedBook is based on the paper workbook "Camden Town Gymnasium 3" by the project cooperation partner Westermann Gruppe. The reason for building on this established workbook is that on the one hand, it is already approved by the state and integrated in regular teaching in schools, and on the other hand the exercises in it are present and can be built on.

The idea behind the adoption of an established workbook is to take it as a basis and make it one step better with real-time feedback. To this end, the adaptation from the paper workbook was conducted with the goal to make the system look and feel as similar to the printed, established, already known workbook as possible. Keeping the system as close to the existing workbook as possible allows users to focus on the added features while feeling familiar with the rest. Figure 5.1 shows a comparison of the FeedBook version (top) with the printed version (bottom) of the same exercise, adapted from Rudzewitz et al. (2017).

While we aimed to mirror the printed workbook as close as possible for what is supported, not the entire breadth of tasks in the paper workbook could be supported. In order to determine which tasks to include, we first classified

²<https://www.westermanngruppe.de/>, last accessed 2020-05-11

every exercise in the workbook terms of activity format, language topics, input types/modalities, task orientation (form/meaning), and openness (restrictions of expected input by the task context) (Rudzewitz et al., 2017).

In the first stage of the project we only included a constrained set of 55 exercises that were determined suitable for interactive feedback. No tasks were included which require peer interaction. In the selection of exercises we made sure that they cover all linguistic constructs in the state curriculum for 7th grade. In the second phase of the project, we step by step extended the exercise data base to 230 tasks from the workbook, and an additional 154 exercises with annotated difficulty from an accompanying book "Camden Town 3 Differenzieren und Individualisieren 3". The reason behind the inclusion of a lot more tasks was to make the adoption of the FeedBook easier for a usage in regular teaching via a broader coverage of printed workbook content. Even if not all tasks are enhanced with scaffolding feedback, school classes do not always have to switch between the printed and online version if the online version provides a high enough coverage of tasks. Another advantage of having as many tasks on the online version as possible is that the FeedBook tasks include all the media normally delivered on a separate DVD delivered with the book. Playing the media from the companion DVD causes learners more trouble than when the media are already delivered in the exercise itself. In the meantime, the BiBox³ by the cooperation partner offers the delivery of media files as well.

Not only the exercises were adapted from paper, but also the error categories for the feedback algorithm (Rudzewitz et al., 2018) and the error categories in the teacher grading interface (Rudzewitz et al. (2017), cf. section 5.2.6). A teacher in the project analyzed scans of workbooks with student answers and teacher corrections made available via the cooperation partner serves as the basis for analyzing which errors 7th grade learners typically make, and what type of feedback teachers provided to the learner answers.

In the next section, we describe which task types were developed based on the printed workbook and how they are used to render exercises in a programmatic way.

5.2.2 Task Domain Model

In order to represent tasks with all the information required to display them to end users and allow for interaction, we developed a domain model for tasks consisting of three levels (cf. Table 5.1): tasks, sub tasks, and task fields.

A task is essentially only an indexed (task id) wrapper around sub tasks. Sub tasks, in turn, are a wrapper around task fields. Let us begin by explaining the properties of sub tasks.

Each sub task is assigned to one *task type* that determines the display and prompt structure (cf. section 5.2.4). The *instruction* and *instruction at the bottom* are text fields with HTML instruction texts display on top and below the exercise. The *page number* is the corresponding number from the work

³<https://www.bibox.schule/>

The top part of the image shows a browser window of the FeedBook platform. The page title is 'Diverse Britain' and the task is 'Exciting things happening at Brixton Village indoor market'. It includes a video player and five questions. The bottom part shows a printed worksheet for the same task, with handwritten answers in green ink. The worksheet includes a video still, a table of answers, and a word puzzle section.

Task 5b5: Exciting things happening at Brixton Village indoor market

b) Watch it again and answer the following questions.

- When does the report take place?
- What was the situation in the market up to 6 weeks ago?
- What has happened since?
- What kind of shops can you find here now? Name 2.
- What is the man's message to his viewers?

Answers:

- On a snowy December night
- many/20 empty shops
- meeting for people to look at empty shops and come up with ideas; since then, the 30 people with the best ideas have opened up shops
- two of the following: old-fashioned sweet shop (vintage) clothes shops, lantern maker, ethical fashion designer, furniture restorer, pop-up galleries, a community shop
- come to Brixton, check out the project and be part of the future of Brixton Village

Task 86: Words: Find the right nouns

You can make nouns from all these verbs. Find the right nouns and sort them into the grid.

-ation	-ion	-ing	-y	no ending
celebration	expression	ending	apology	change
combination	impression	meaning		challenge
				end
				report

Figure 5.1: Theme 5, task 5b5 in the FeedBook (top) and in the printed workbook (bottom) © Westermann Gruppe, integrated like in Rudzewitz et al. (2017)

task	sub task	task field
task id	subtask id	sortIndex
list of subtasks	list of taskfields	prompt text
	task type	input type
	instruction	target answer
	instruction at the bottom	example answer
	page number	link to information sources
	links to media files	list type indicator
	links to grammar sections	
	support text	
	given words	
	task focus	
	task orientation	
	target language constructs	
	difficulty level	
	feedback disabled	

Table 5.1: Domain model: properties of tasks, sub tasks, and task fields

book. In the feature *links to media files*, references to media such as images of texts, audio or video files are stored. *Links to grammar sections* is similar, but contains references to HTML pages explaining specific grammar phenomena. The *support text* is simply a plain text field that provides additional hints to students and is rendered at the top of the page. In *given words*, a list of words provided for this exercise is encoded. These will be rendered as clickable objects that upon click are enhanced with a strike-through decoration, comparable to crossing out words on paper. The *task focus* property indicates whether it is a meaning-oriented, form-oriented, or meaning-and-form-oriented task, relevant when generating and displaying feedback. Related to that is the *task orientation* property responsible for indicating the degree of openness of the task (open, closed, or completely open). This again determines if feedback is displayed (if it is not a completely open task), and if yes which type of feedback (general-purpose grammar feedback for open tasks, specific scaffolding feedback for closed tasks). In *target language constructs*, a list of manually selected language constructs is stored that indicate the pedagogical goals of the task in terms of language constructs to acquire or practice. The *difficulty level* feature was introduced to accommodate the difficulty rating (easy, medium, complex) assigned to exercises provided as additional materials to the workbook. Finally, *feedback disabled* is a flag that allows to turn feedback off for cases where the data shows it is not helpful or confusing.

Task fields constitute individual prompts or items and are always assigned to a specific subtask. The *sort index* annotation tells the system about the order of the prompts. In *prompt text*, the text of this item is stored, with the format depending on the task type (cf. section 5.2.4). Similarly, *input types* (character, partial word, sentence, text, true/false input), *target answers* and

example answers are dependent on the task type and discussed in more detail in the respective sub sections. For meaning-oriented tasks, the *link to information sources* contains a reference to media with annotations for the part of the input that answers this task field. The last property, *list type indicator*, determines whether the answer to this task field is a list of items or a single item. In the former case, it tells the system how many items are expected (in any order).

5.2.3 Authoring Tool

In the FeedBook system, each exercise was entered manually into the system. First, an exercise needs to be added to the system's table of contents. Once present there, it can be edited in the system's authoring tool.

The authoring tool allows to create objects and annotate all properties of the system's domain model (cf. section 5.2.2) on the level of sub tasks and task fields.

One shared domain structure for all task types (cf. section 5.2.4) is used. The advantage is that internally the data structures for all task types is shared and the interface can be built once on the same format. On the other hand, it requires flexibility in the format used for different task types since when all data structures are shared, distinctions can only be made in the values of the data structures.

The interface is split in two parts: at the top (cf. Figure 5.2), meta data of the sub task like instructions, given words, etc. are annotated. These properties are independent of the specific task type selected (cf. Figure 5.2 at the top for the selection of a task type). Below the sub task meta data part, the interface allows to add and edit task fields (cf. Figure 5.3). Annotators need to add one prompt at a time, select an input type, add a sort index, and then add properties like a target answer or prompt text (cf. Table 5.1 for a comprehensive list).

The task of 'translating' tasks from the printed work book into the system constitutes going from unstructured data in the printed work book to structured data in the system's domain model. This translation is not lossless since the system's domain model represents an abstraction over many examples that can not fully accommodate for specifics of all examples, especially in term of how elements will be rendered.

5.2.4 Displaying Exercises to Students

Just like the authoring interface (cf. section 5.2.3), the exercise perspective for students is split in a task type-independent and a task type-dependent part. In general, the target is to display exercises as close as possible to the printed version.

In the task type-independent part at the top of the exercise (cf. Figure 5.4), the interface displays the workbook chapter and chapter title. On the right, there is a print button in case students want to print the exercise out and work on paper. Below the header, the interface displays the task number and title as a heading. The next element is the instruction text for the exercise. Below

FeedBook

feedback.schule/feedbook/mini

Options | Mai (Annotator) | Feedback

FeedBook: Camden Town Workbook 3

Choose a XML file to import a subtask: Detail auswählen | Keine ausgewählt | Reset submissions | Import XML

Export current subtask to xml: Export XML

Theme 0 , SubTask 3 : A sunny day

Select the type of this task: SHORT_ANSWER | TRUE_FALSE | CLASSIC | TRUC_FALSE | MAPPING | SURVEY_TABLE | MULTIPLE_CHOICE

SubTask index (starting from 1):
1

SubTask page number:
4

Zeige C-Test Konverter

Enter the task support text here:

Enter the task instruction here:

Enter the second task instruction here:

Figure 5.2: Annotator view: authoring tool for task meta data

FeedBook

feedback.schule/feedbook/mini

Options | Mai (Annotator) | Feedback

FeedBook: Camden Town Workbook 3

Add a prompt!

→ Caroline → in the TV? What are you doing?
 Enter target answer (1)
 Enter example answer (1)
 Enter course information source (1)
 Enter image for course information source (1)
 Enter precise information source (1)
 Enter image for precise information source (1)

Keine Angabe | Keine Angabe

Enter number of expected:
1

Disable prompt

→ you → in front of the TV?
 are you doing?
 Enter example answer (2)
 Enter course information source (2)
 Enter image for course information source (2)
 Enter precise information source (2)
 Enter image for precise information source (2)

PHRASE | Keine Angabe

Enter number of expected:
2

Disable prompt

→ you → look → out of the window yet? It's warm and sunny outside. Let's have you looked.
 Enter example answer (3)
 Enter course information source (3)
 Enter image for course information source (3)
 Enter precise information source (3)

Figure 5.3: Annotator view: authoring tool for task fields

that, the interface shows a link to a grammar section, in Figure 5.4 a link to an explanation of the simple past. Clicking on it opens a popup window with the grammar section in it. Below this, there is a list of given words that can be clicked on for striking them through. To the right, there is a text box titled "Support", which contains additional hints for students. In the example, an image is displayed. In the general case, the system supports the display of any number of media (audio/video/photo/text) in this section.

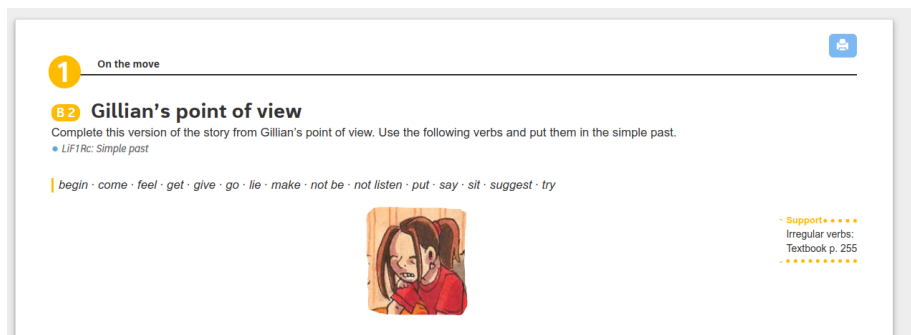


Figure 5.4: Student view: exercise display task type-independent information

FeedBook offers five exercise/task types. A task type is a generalization away from a specific exercise example to a principled way of representing and displaying an exercise. Having exercise types allows to automatically generate the display of an exercise instead having to fully spell out the entire HTML of a task like in a traditional approach. A task type not only determines the rendering of a task, it also determines the representation of task information.

Worth mentioning before that is that the domain model (cf. 5.2.2) is the same for all task types apart from the task type setting and the prompts. For example an exercise can always have an instruction or prompt image at the top, independent of the task type. The function of a task type is to provide a principled way of representing and visualizing prompts.

For example while for short answer tasks the prompt text is always a sentence or phrase above a learner answers, for fill-in-the-blanks tasks the prompt text is the text up to a gap and not the entire sentence or reading text. In the following, we will present the different task types one by one.

Short Answers Short answer tasks consist of a list of individual prompts (typically questions or statements) and an input field allowing the student to provide a short answer, typically one sentence. Each prompt is displayed on a separate line and comes with a textual target answer that provides a direct answer to the question raised in the prompt. Figure 5.5 shows an example of a short answer task in which students need to use the textual cues and images to form sentences using the past progressive.

FeedBook

feedbook.schule/FeedBook/#init

Optionen FeedBookTestSchüler (Schüler) FeedBook : Camden Town Workbook 3

1 On the move

C3 What was ... doing while Gillian was doing something else?

Write down what Gillian's friends were doing while she was running away from home. Use the past progressive in both parts of the sentence.

L1/1 Re: Past progressive

1 buy Arsenal tickets 2 feed Patch 3 watch TV 4 sit on the bus

1. buy Arsenal tickets/sit on the bus
Charlie ... while Gillian

2. feed Patch/sit on the bus

3. watch TV/sit on the bus

Figure 5.5: Exercise view: short answer task

Fill-in-the-Blanks Fill-in-the-blanks tasks consists of a list of prompts that, taken together, form a text with gaps to fill in. Each prompt models a gap and the text following the gap. The target answer of a fill-in-the-blanks task is the textual representation of the answer to the gap in front of the prompt text. Therefore, the task context of one fill-in-the-blanks gap can span multiple prompts since a sentence can have multiple gaps and is thus split up into several prompts internally. This requires a special treatment of the first prompt: it solely consists of a prompt text and no input type or target answer, since there is no gap or interaction method right at the beginning of the task. The system offers the function to present learners with a multiple choice menu instead of a gap for this activity type to avoid typing errors. In this case, the target answer consists of an index that indicates which of a list of provided example answers is the target answer. Figure 5.6 shows an example of a fill-in-the-blanks task in which learners have to use the correct tense forms to fill out the text's gaps.

True/False In true/false tasks, each item consists of a statement that learners have to evaluate as either true or false. The prompts are rendered sequentially below each other in a table. Upon clicking on a cell, an arrow is inserted in the corresponding cell, with no cell selected if the learner did not yet interact with this prompt. The target answer of a true/false prompt consists of a number with '0' meaning false and '1' meaning true. In the example in Figure 5.7, learners

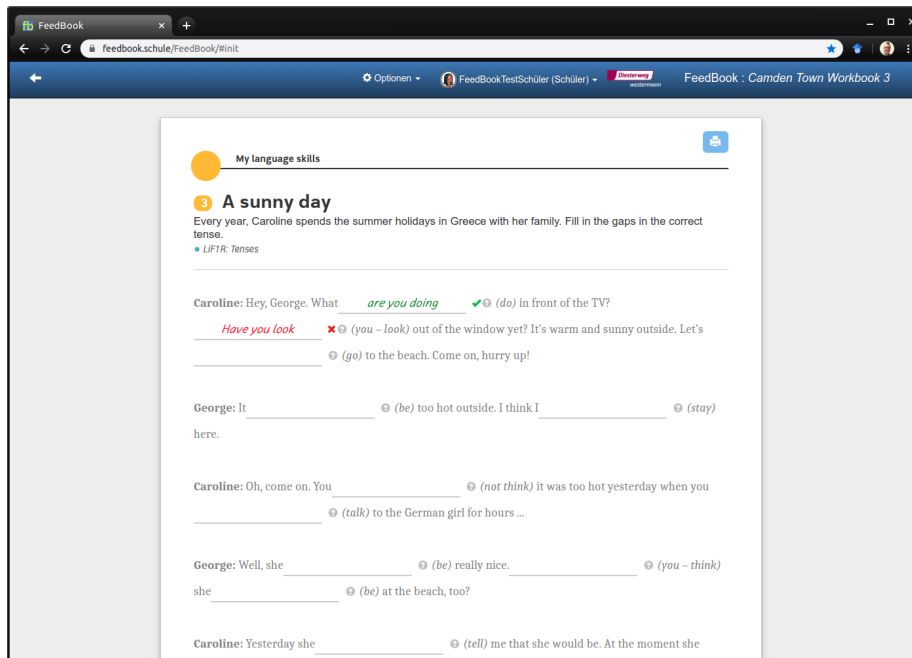


Figure 5.6: Exercise view: fill-in-the-blank task

need to listen to a spoken text and tick each statement as true or false.

Mapping Mapping tasks require learners to provide a response of one character (letter or numeral) to map a prompt text to a set of alternatives. Figure 5.8 illustrates this: learners need to read a text and are then presented with a set of statements. Each prompt requires the learner to evaluate which of the given labeled answers is correct and communicate this choice to the system by entering the index/label of the choice into the input field. In a later stage of the project, mapping tasks were subsequently modeled as fill-in-the-blanks tasks with multiple choice input.

Survey Survey tasks offer a list of choices to learners where they need to fill out which of the options pertain to them. The system allows to configure whether only one choice can be given, multiple of them, or even if there are free-text input fields. Items in survey fields do not come with target answers, since the answer depends on each individual student. This is illustrated in the example in Figure 5.9, where learners need to state which of the given languages they are learning in school.

In follow-up projects of FeedBook, more task types such as jumbled sentences, categorization tasks, drag-drop exercises, memory, and more were developed and implemented, but were not part of the system described here.

FeedBook

feedbook.schule/FeedBook/#init

Optionen FeedBookTestSchüler (Schüler) FeedBook : Camden Town Workbook 3

1 On the move

B.7 Talking to Gwynn

a) Listen to this conversation and decide if the sentences below are true or false.

0:00 / 2:11

true	false	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	1. Mrs Collins and Gwynn talk about the move to Wales.
<input type="checkbox"/>	<input checked="" type="checkbox"/>	2. Mrs Collins tells Gwynn what a difficult child Gillian is.
<input type="checkbox"/>	<input type="checkbox"/>	3. Gwynn gives Mrs Collins some advice on how to help Gillian.

Vorherige Aufgabe a b Nächste Aufgabe

Speichern Abschicken

Figure 5.7: Exercise view: true/false task

FeedBook

feedbook.schule/FeedBook/#init

Optionen FeedBookTestSchüler (Schüler) FeedBook : Camden Town Workbook 3

Part 4

Then she asked me angrily: "Will I have to share a room with this new baby?" I tried not to show my disappointment although she must have noticed my reaction. "There will be enough room for everyone in our big new house in Wales," I answered. I didn't expect such a strong reaction from her. "Why don't you just get out of my room?" she shouted. I tried to

kiss her on the cheek but she turned away angrily. So I went into the kitchen. Even if Gillian seemed upset about moving, I was relieved that I had told her everything. I was confident that she would be fine once she met Gwynn.

1. Mrs Collins decided to make Gillian a special breakfast because it was...

a) early in the morning.
b) her wedding day.
c) the day of Gwynn's visit.
d) the weekend.

2. Mrs Collins was upset because Gillian didn't...

a) like shopping.
b) smile at her.
c) speak to her.
d) want any breakfast.

3. Mrs Collins decided to go into Gillian's room because she wanted...

a) to bring Gillian her breakfast.
b) to make it a happy day for them all.
c) to tell Gillian about Gwynn's job.
d) to tell Gillian that she was angry.

4. When Mrs Collins talked about Gwynn, Gillian...

↑

Figure 5.8: Exercise view: mapping task

The screenshot shows a web browser window with the URL 'feedbook.schule/FeedBook/#init'. The page title is 'FeedBook : Camden Town Workbook 3'. The main content is a survey form titled '1 Fragebogen' and '1 Angaben zur Person'. The question is 'Welche Fremdsprachen lernst Du in der Schule?'. Below the question is a table with five columns: 'Englisch', 'Französisch', 'Spanisch', 'Latein', and 'andere'. The 'Englisch' and 'Französisch' columns have blue checkmarks. Below the table, there is a note: 'Kreuze alle Fremdsprachen an, die Du zur Zeit in der Schule lernst:'. At the bottom of the form, there are navigation buttons: '< Vorherige Aufgabe', 'a', 'b', 'c', 'd', and 'Nächste Aufgabe >'. At the very bottom, there are two buttons: 'Speichern' and 'Abschicken'.

Figure 5.9: Exercise view: survey/questionnaire

5.2.5 Displaying Feedback

Feedback is computed and displayed as soon as a learner leaves an input field (i.e. it loses focus) or when a user clicks on a question mark next to an input field. In the former case the feedback popup is triggered automatically, in the latter case the learner actively requests feedback. In an earlier version of the system, the practice interface contained additionally a "check all" button which triggered the computation of feedback for all input fields. In this case, learners had to click on the question mark next to an input field to get the textual feedback in the feedback popup. The same mechanism was used for checking learner answers before submitting them to a teacher and when re-entering a previously saved, but not submitted exercise.

As shown in Figure 5.10, the system displays a green check mark for answers that have been evaluated as correct by the system. For cases where the learner answer does not match a target answer, a red cross is displayed. Furthermore, the student text is displayed in the same color. In case feedback was blocked in the study (when a learner was in the control group for a certain language phenomenon), no check mark or cross was displayed and the learner answer remained gray.

Once feedback is computed and the system decided the learner can see it, it displays it in a popup. The popup contains the feedback text, a magnifying glass, a radio button, and an 'ok' button which closes the popup window. Initially, the system only tells the learner that there is an error, but not where the error is (cf. Figure 5.10). In case learners want to get this information, they can click on the magnifying glass. This extends the feedback message by repeating the learner answer and displaying the location of the error (cf. Figure 5.11). Especially for longer learner answers, this is a useful feature to pinpoint the

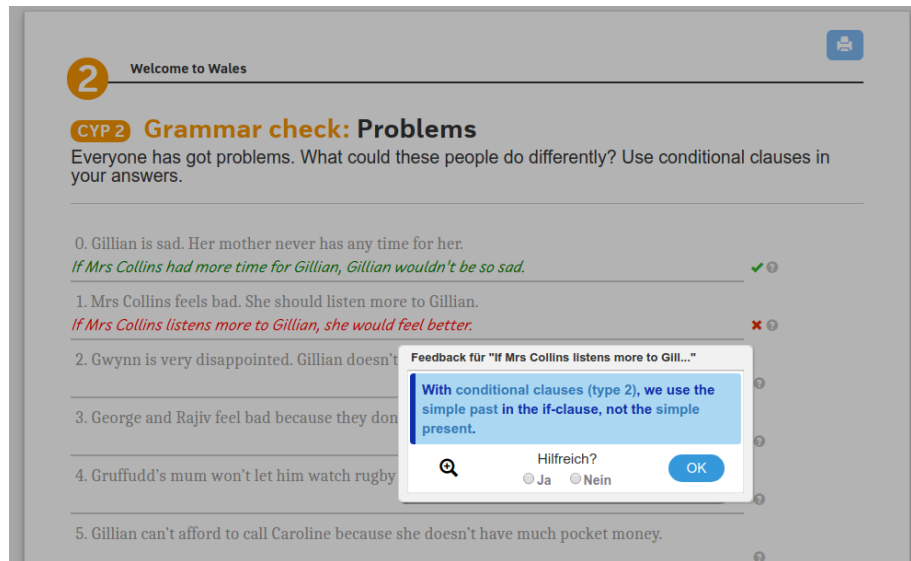


Figure 5.10: Exercise view: displaying feedback (stage 1)

location of the error. The highlighting shown there is based on an efficient surface form comparison of words in the target and learner answer and also used in the teacher grading interface (cf. section 5.2.6).

Once a feedback has been computed, it is cached on the client side. When the learner enters the same answer in the same input field again, the system can then efficiently retrieve it again. For every computed feedback, even if it is filtered and blocked by the blacklist for experimental groups, a `LoggingToken` object is created and added to the interaction log. This object contains information about when it was computed in the form of a numeric sort index for this task and this student along with a server time stamp. Additionally, it contains a reference to the feedback message and the learner input and a flag that indicates whether this is a manually triggered feedback or automatically triggered. Automatically triggered means that this feedback was computed as part of a batch process, either with the check-all button, with the check-all function when re-opening an exercise, or when submitting to the teacher. The manual case includes both cases in which a learner actively clicked on a question mark and when the feedback got displayed because the input field lost focus.

5.2.6 Teacher Grading Interface

The teacher grading interface (Rudzewitz et al., 2017) allows teachers to provide delayed feedback to students. This interface was developed in the first stage of the project where no automatic feedback was given yet by the system. In essence the task of this component is to offer teachers the opportunity to grade student submissions as similar as possible compared to the paper version, but with some

2 Welcome to Wales

CYP 2 Grammar check: Problems

Everyone has got problems. What could these people do differently?

0. Gillian is sad. Her mother never has any time for her.
If Mrs Collins had more time for Gillian, Gillian wouldn't be sad.

1. Mrs Collins feels bad. She should listen more to Gillian.
If Mrs Collins listens more to Gillian, she feels better.

2. Gwynn is very disappointed. Gillian doesn't like Wildings

3. George and Rajiv feel bad because they don't have a present for Gillian to take to Wales.

4. Gruffudd's mum won't let him watch rugby because he hasn't finished his homework.

5. Gillian can't afford to call Caroline because she doesn't have much pocket money.

Feedback für "If Mrs Collins listens more to Gill..."

With conditional clauses (type 2), we use the simple past in the if-clause, not the simple present.
 If Mrs Collins listens more to Gillian, she feels better.

Hilfreich?
 Ja Nein

Figure 5.11: Exercise view: displaying feedback (stage 2)

assistance.

The key functionality of the teacher interface consists of the function to add annotations. Teachers can select a span of a learner answer with the mouse, or they can touch a learner answer and are then offered two sliders in a popup to determine a span of the learner answer they want to mark. Once the span is determined, the system opens the annotation dialog popup (see Figure 5.12). As a first step, teachers select the type of error: the system offers a choice of thirteen form error categories and six content error types depicted in Table 5.2. The error types were determined based on errors annotated by teachers in paper data that served as the basis for creating the initial version of the FeedBook (cf. section 5.2.1). Below the error type selection, teachers have the option to adjust the error span post-hoc. The annotation popup furthermore contains two free text input fields: the first one is used for providing comments or hints to the student about this error, the second one contains by default the target answer. Teachers can modify the contents of both input fields to provide students with additional explanations. Once a teacher clicks on "Update", the annotation is added next in a column to the exercise, with a handler to delete them again if needed. Once a teacher saves the correction of a learner answer, all annotations are saved in the data base with information about which task field the annotation pertains to and which teacher created this annotation.

The first feature to assist teachers in grading is a visual marking of learner answers. If an answer is identical to a target answer, it is marked with a green check mark next to it. For answers that are not identical to a target answer,

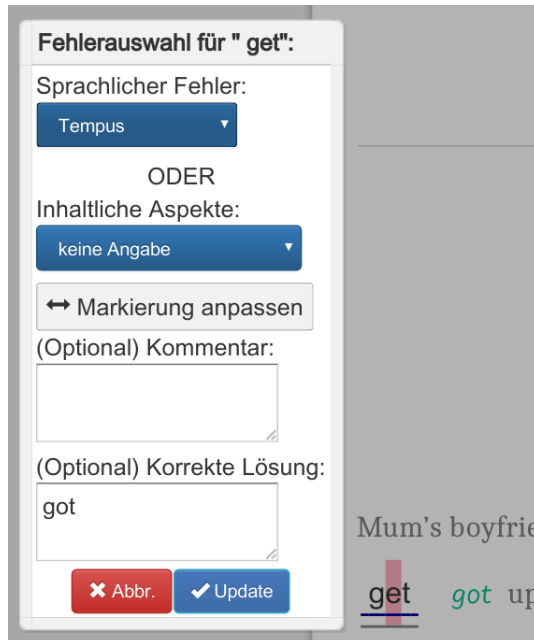


Figure 5.12: Teacher view: assigning an error annotation to a learner answer enhanced with the highlighting of the difference to the target answer. Taken from (Rudzewitz et al., 2017, page 6)

Language form errors	Content errors
phrasing, agreement, determiner, preposition, grammar, spelling, pronoun, tense, clause structure, word choice, missing word, word order, punctuation	problematic understanding, missing information, wrong information, lack of understanding, extra information, alternate answer

Table 5.2: Error types in FeedBook (taken from (Rudzewitz et al., 2017, page 6))

the system computes a visualization of a diff algorithm. This works by aligning each word in the learner answer to the closest word in the target answer in terms of edit distance on the surface form. In order to avoid that parts of the learner answer are aligned that only happen to share some characters, and given that the algorithm does not work on more levels of abstractions (like in Ziai (2018)), a heuristic is applied: only for words with half or more than half of the characters shared between the learner and target answer, a diff is computed and visualized. In Figure 5.12, the algorithm marked the character 'e' in the word "get" (target answer: "got").

As described above, annotations are saved with information about the task field and the teacher. This data is used for the second type of assistance, the feedback memory. If for the same task field the same learner answer was given already by another student, and if there is a previous annotation for it, then this annotation is loaded again from the feedback memory. This component works similar to a translation memory (cf. e.g. Somers (2003)) in that previous translations (in this case corrections) are loaded again, saving the teacher the time and effort to add the same annotation again. Teachers can configure via their profile page whether they want to use the feedback memory, and if so, whether they only want to load annotations by themselves or also annotations by other teachers.

In case there are no previous annotations, the system tries to suggest some annotations (which are modifiable or deletable by teachers). The system uses a lightweight variant of the feedback generation NLP pipeline to annotate learner answers and compare them with pre-annotated target answers having the same levels of annotations. Table 5.3 shows the components of the pipeline. These components are used for creating and annotating a Common Analysis Structure in UIMA (Ferrucci and Lally, 2004) with all the NLP tools providing linguistic annotations. The system then applies rule heuristics to detect spelling errors, wrong word choices, and tense confusions. If a rule is applicable, it adds a proposal for an annotation to teacher interface distinguished with a different color to make clear that this is an automatic annotations. This rule-based algorithm is less capable than the full feedback generation algorithm (cf. chapter 6), but still can provide assistance and paved the way to the more advanced processing techniques described later.

Below the exercise and annotations, there is a free text input fields in which teachers can provide a global comment and provide a star rating by selecting stars from a range of one to five. This information, along with the annotations, is visualized in the student result view.

5.2.7 Student Result View

The student result view allows learners to inspect the feedback provided by the teacher. While the exercise itself is rendered in the same way as during the practice mode, it contains additional feedback by the teacher.

⁴<https://emorynlp.github.io/nlp4j>

NLP task	Component Used
Tokenization, Sentence Detection, POS tagging	NLP Toolkit for JVM Languages NLP4J ⁴
Lemmatization	Morpha (Minnen et al., 2001)
Morphology	SFST with EMOR model (Schmid, 2005; Karp et al., 1992)
Dependency Parsing	MaltParser (Nivre et al., 2007)

Table 5.3: NLP components used in auto-correction (taken from (Rudzewitz et al., 2017, page 6))

For answers that are correct, a green check mark is displayed next to the learner answer. Incorrect answers, in contrast, are rendered without a check mark and potentially a teacher annotation. In an early version of the system everything that was not marked as false by the teacher was marked with a green check mark, but later we refrained from that since the assumption that teachers would mark every single error or mistake did not hold.

Teachers can mark parts of a learner answer (cf. section 5.2.6), assign an error category to this span, and potentially a comment and the target answer. Every error type is assigned a specific color which is used for an underline decoration under the marked span of the learner answer, and as the color of a small flag in the actual annotation.

In the student result view, these annotations are placed in a column to the right of the exercise display (see Figure 5.13). For annotations that come from the system’s auto-correction, the annotation is rendered with a yellow background, whereas annotations by the teacher are rendered with a blue background. This allows learners to see which of feedback was annotated by their actual teacher. Inside the annotations, a label indicates the type of annotation, followed by the target answer (if the teacher agreed to show it), potentially followed by a comment written by the teacher.

Due to technical reasons (responsive layout with multiple column), it was not possible to render annotations on the same line as the learner answer. Therefore, the reference of annotations to learner answer is rendered in the following way. When a learner hovers over an annotation, or when a learner touches an annotation on a touch device, then the learner answer this annotation pertains to is highlighted with a yellow background. This allows learners to hover over the list of annotations and then see which parts of their answers have been marked by the teacher.

Below the exercise and annotation display, learners see a star rating by the teacher, with five stars as the maximum. If a learner received less than five stars, the higher stars are rendered as empty. At the bottom of the page, the system displays a free-text comment by the teacher if present. The ”Gesehen“ button at the bottom takes the learner back to the student lobby.

FeedBook : Camden Town Workbook 3 Start Optionen Logout Lars (Schüler) Distroway


Hilfe

1 On the move

8.2 Gillian's point of view

Complete this version of the story from Gillian's point of view. Use the following verbs and put them in the simple past.

begin · come · feel · get · give · go · lie · make · not be · not listen · put · say · sit · suggest · try



Support:
Irregular verbs:
Textbook p. 255

Mum's boyfriend was coming to meet me so of course I got up in a bad mood. But Mum gave me a great big smile. She made me my favourite pancakes with maple syrup for breakfast but I wasn't hungry. She wanted to cheer me up and said that we go shopping. That usually puts me in a good mood but not today. So I said something about homework and stormed into my room. I laid down on my bed and felt really sorry for myself. Just then Mum came in. She sat down on my bed and put her arms around me. She wanted to talk about Gwynn but I refused.

Bewertung durch Lehrkraft: ★★★★★☆

Kommentar: okay

- Wortfehler; Korrekt: tried
- Ausdruck; Korrekt: suggested
- Autokorrektur; Korrekt: went
- Autokorrektur; Korrekt: lay
- Verstehen problematisch; Korrekt: began
- Autokorrektur; Korrekt: didn't listen

Gesehen

Figure 5.13: Exercise view: displaying the result of feedback by the teacher

5.3 System Environment around Exercises

Apart from a possibility for students to work on exercises, a web application in the form of a tutoring system to be adopted in an everyday teaching and learning context, needs additional functions and user interfaces to them.

In this section, we discuss what is necessary in the system apart from exercises and Learning Analytics to support what we actually want to do. We discuss different functions and user interfaces that fulfill some of the basic needs of users so that they will eventually use and adopt the system, interact with the exercises, and allow us to learn from their behavior.

5.3.1 Student Lobby

The student lobby is the central start page and the first interface learners see when they log into the system. The term 'lobby' is used in this context because, just like in a hotel reception lobby, this is the starting point from which students are forwarded or proceed by themselves to other functions of the system.

The lobby primarily provides students with a hierarchical index to tasks adopted from the workbook. Learners are able to see what tasks the system offers, and what the status of a task is for the current student (e.g. saved but not submitted or waiting for review by a teacher). FeedBook does not provide a linear index as it is the case for the normal workbook where it is possible to flip through the pages, but instead offers a hierarchical access to chapters, sections dividing the chapter, and group of tasks in sections. The alternative would be to provide all tasks in sequential order or via digital version of the book pages. A linear index, however, bears the danger that learners are overwhelmed if there would be too many tasks displayed at one time. And for a model in which the tasks are presented via digitalized book pages, a large amount of work in terms of layout needs to be put into it. The focus of the FeedBook project, in contrast, is to do computational linguistics and not digital publishing. A system going more in this direction is the BiBox⁵ by Westermann Gruppe.

With a world of web technology increasingly fragmented, it is important that a system uses responsive web technologies. In April 2020⁶, 54% of users use mobile clients, 42% use desktops, and 4% tablets. This is further fragmented into different browsers (with Chrome, Safari, Firefox the three biggest, in this order) and operating systems (Android, Windows, iOS, in this order).

In order to provide scalability across devices, FeedBook uses the Bootstrap framework⁷. In the student lobby specifically, the system uses Bootstrap's grid system for a responsive design. The grid system defines five responsive breakpoints that allow to change the layout depending on which viewport size is detected (extra small, small, medium, large, extra large). The core idea is that a web page defines rows, with each row consisting of twelve columns. For

⁵<https://www.bibox.schule/>, last accessed 2020-05-04

⁶source: <https://gs.statcounter.com/platform-market-share/desktop-mobile-tablet>, last accessed 2020-05-04

⁷<https://getbootstrap.com/>, last accessed 2020-05-02

each break point, user interface designers can define how many of the twelve columns are taken by an element.

In the case of the FeedBook, the list of elements corresponding to the six book chapters is defined as one row. For big viewports (e.g. desktop computers), each chapter display takes three columns, i.e. four chapters are displayed in one row (cf. Figure 5.14). For medium screens (e.g. tablets), each chapter occupies six columns (cf. Figure 5.15). For small screens (e.g. mobile phones), each chapter occupies the full twelve columns (cf. Figure 5.16).

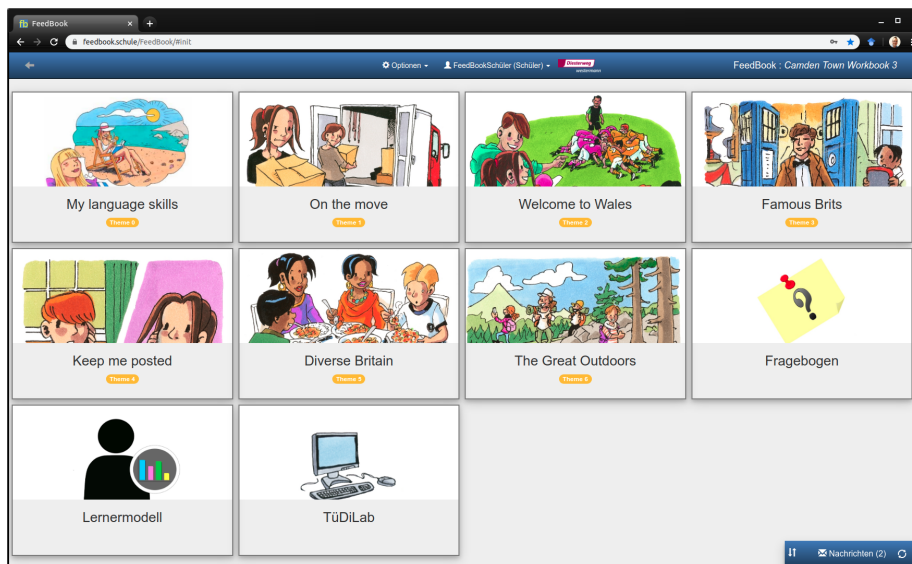


Figure 5.14: Student view: large viewport layout

Once a student selects a chapter by clicking on or touching a tile, a list of sections is displayed (cf. Figure 5.17). There is one tile for each of the book sections (A, B, C, Extra Training) and one section for tasks adapted from the additional material (Additional Practice).

Clicking on a section displays a list of tiles, each tile standing for a task from the workbook. In case an exercise consists of multiple parts, one tile is displayed for every sub task to make transparent to learners that an exercise has multiple parts and that the multiple parts (e.g. part a and b of an exercise) can have a different nature in terms of activity type and feedback support.

Each task tiles displays a thumbnail to users to give them some information about the task (cf. Figure 5.18). The task image in the middle indicates what type of activity it is from the perspective of the exercise modality (reading, listening, vocabulary, writing, grammar, mediation, speaking) As a means of creating a link between the printed workbook and the digital version, the subtask tiles indicate the task number (central badge) and page number (left lower corner).

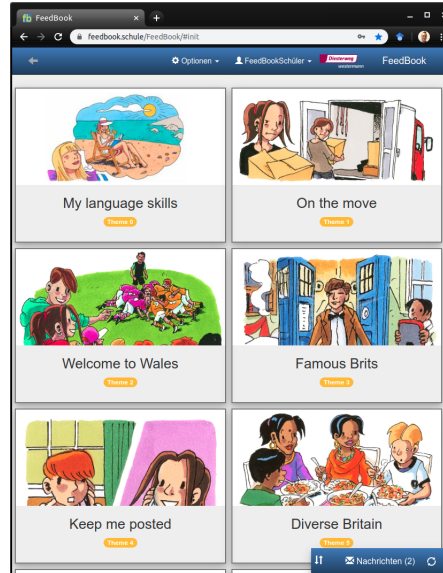


Figure 5.15: Student view: medium viewport layout

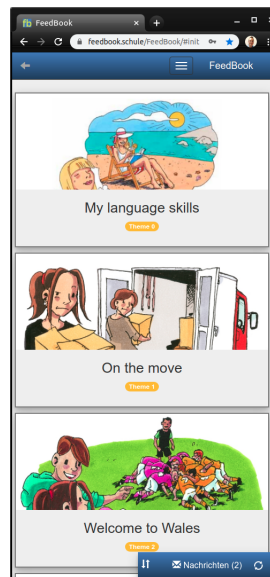


Figure 5.16: Student view: small viewport layout

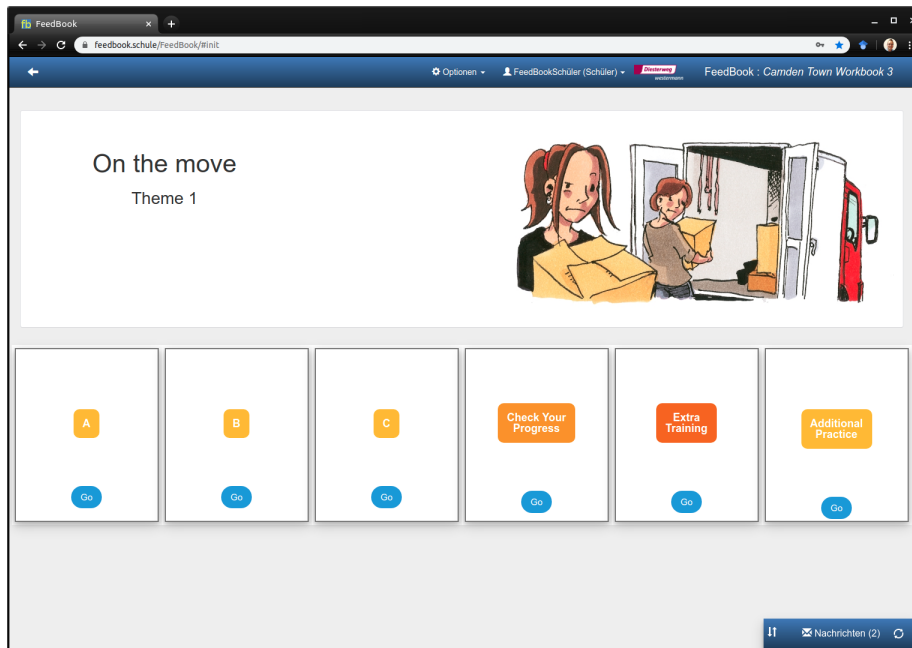


Figure 5.17: Student view: displaying sections

At the top of a task tile, a symbol indicates what type of feedback or support the system offers, with a light bulb standing for scaffolding feedback, a green eye symbol for general-purpose grammar checks, and no symbol for no feedback. In chapter 6, we explain each feedback type in detail.

Finally, each task tile gives learners information about the status of the associated task with a symbol displayed in the lower right corner. If this exercise has not been attempted, this space remain empty. In case a learner attempted and saved, but not submitted this task, a blue circle is showed. When a task has been submitted to the teacher and is pending for grading, an hourglass is shown. Once an exercise has been corrected by the teacher, a check mark is shown there in different colors corresponding to the star rating, from red (1 star) to green (5 stars).

Apart from providing access to chapters, sections and tasks, the student lobby furthermore contains tiles that offer access to the learner model (cf. Figure 5.14, bottom left) and special tiles for additional functions like questionnaires or individual difference tests.

5.3.2 Teacher Lobby

The teacher lobby is the central start page that users with a teacher role see when logging into the system.

It consists of a table of contents represented in a tree with sections and

The screenshot shows a web browser window displaying the FeedBook interface. The browser's address bar shows the URL `feedbook.schule/feedBook/#init`. The page title is "FeedBook : Camden Town Workbook 3". The main content area features a header with the text "On the move" and "Theme 1 Section A" next to an illustration of a woman and a man moving boxes into a van. Below the header is a grid of task cards. Each card has a yellow ribbon icon in the top right corner and a small yellow location pin icon in the top left. The first three cards are titled "Words: Problems" and are labeled "A 1 a", "A 1 b", and "A 1 c" respectively. Each of these cards includes an illustration of a child thinking about the words "BIG", "small", and "WALK". The fourth card is titled "Can you read these words?" and is labeled "A 2". Below the grid, there are three buttons: "Ask Ally", "Giving advice", and "Different tenses". A notification bar at the bottom right shows "11 Nachrichten (2)".

Figure 5.18: Student view: displaying tasks

individual tasks on the left side, and a task table with information about student submissions for a selected task on the right side (cf. Figure 5.19). Clicking on a chapter in the first level of the tree displays all tasks from the chapter's main sections (*A, B, C*). For the supplementary sections (*Extra Training, Additional Practice*), separate nodes are shown in the tree that can be unfolded to below it display the section's tasks. The tree thus represents a hierarchical index to exercises (as it is the case for students), but projected to be displayed on one single page instead of several pages.

On the right side of the page, the system shows a table listing all students in the class and the submission status for a task selected in the tree. This table displays submission data of all students in the class and displays the data in five columns. The first column lists the names of all students in the school class assigned to this teacher. In the next column "Abgabe", there is either a blank value if the given student has not submitted yet, or, for submitted tasks, it lists the date (not time of the day) of when the corresponding student submitted their answers to this task. The "Status" columns is used to display if for this student there is no submission yet ("nicht abgegeben"), if the submission is pending ("zu korrigieren"), if it is partly corrected ("teilweise korrigiert"), or if it is corrected and sent back to the student ("korrigiert"). A list with additional information, used for displaying whether reminders have been sent to students, gets rendered in the column "Anm.". (Anmerkungen). The last column has the label "Bewertung" and shows a rating for tasks that the teacher has corrected or that the student submitted as fully correct. It is rendered out as a check mark in a color range from red (one star) to green (five stars), visualizing the star rating assigned by this teacher to this submission, analogous to the visualization in the student lobby (cf. section 5.3.1).

Clicking on a row in the task table prompts the teacher forwards the teacher to the correction interface for tasks that are pending for correction. For students who did not submit yet, the system asks teachers if they want to send a reminder. For already corrected tasks teachers are prompted in a dialog window whether they want to correct the task, unlock the task again for revision by the student, or view their earlier correction.

As an alternative to the tree view visualizing submissions by chapter, section and task, teachers can get list of all student submissions for all tasks. Clicking on the top menu entry "Abgaben" loads an alternative view of the teacher lobby, but instead of a tree to select, it displays a version of the task table (which is by default shown right to the tree) with a list of all submissions across tasks. In this view, the students without submissions are not displayed, but only the actual submissions. This widget can thus serve as a ToDo list for teachers where they can see what submissions are pending for grading.

Shared for both views is a list of filters to hide information (cf. Figure 5.19 above the task table). This is especially useful for bigger school classes. The filters allow to hide or show only submissions that need to be graded or are already corrected. Another functionality worth mentioning are the buttons at the bottom of the task table: they allow to automatically accept corrections by the system for the given task and students (green button), send a reminder to

those who did not submit yet (purple button), unlock the task again for those who have already submitted (violet button), or mark all submissions as seen but not yet graded (yellow button).

At the bottom right of the teacher lobby, teachers can see a news feed that automatically minimizes when selecting a task in the tree view (cf. section 5.3.3). On the bottom left, for teachers who participated in the study (cf. chapter 7), teachers see an overview over the school class' progress on selected core tasks with one progress bar for every core task visualizing the percentage of students who worked on this task. At the top of the widget, the progress is displayed for the current and thus most recent chapter, older chapters are shown further down the list. This widget can be minimized, but not hidden with the intention to create a salient representation of the progress of students on core tasks that need to be done in the study.

In a nutshell, the teacher lobby provides an overview over who did an exercise when, what needs to be corrected, what has been corrected, and how good a submission was.

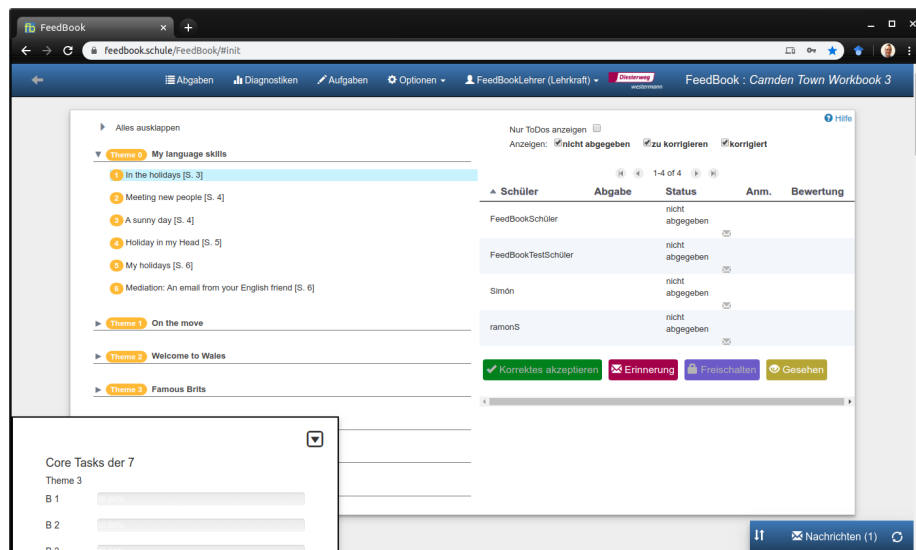


Figure 5.19: Teacher view: Start page/lobby

5.3.3 Administrative System Functions

User Management In FeedBook, users with one of five roles exist: *student*, *teacher*, *admin*, *material designer*, and *annotator*. Users with the role *admin* have the possibility to create *teachers* and associate school classes to teachers. The task of teachers when logging into the system is to create *student* accounts for every student in their class, who will be added to the school class associated with this teacher. *admin* users can furthermore create *annotator* accounts: these

users have the ability to modify exercises via the authoring tool (cf. section 5.2.3). While *admin* users have the ability to reset or change the password of all users, *teachers* can manage the passwords of the students in their class (cf. Figure 5.20).

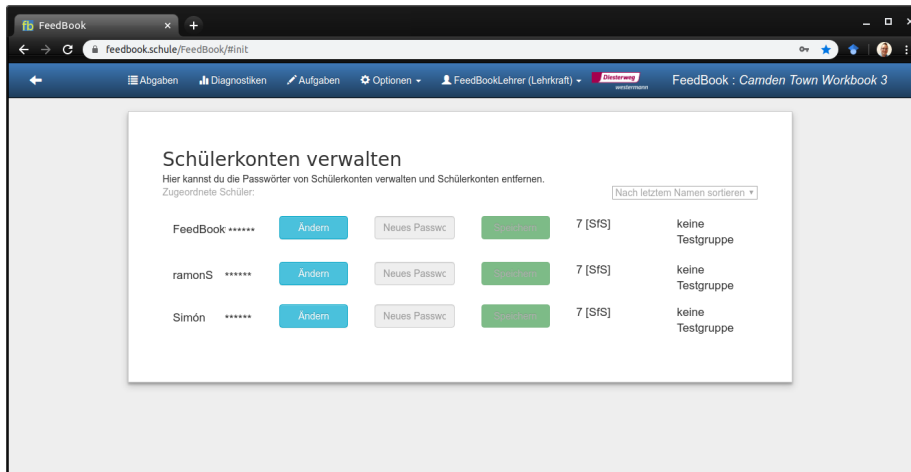


Figure 5.20: Teacher view: managing student accounts

Each user has a profile page where they can select an avatar picture, associate an e-mail address, and change their current password (cf. Figure 5.21).

Messaging system The system implements a messaging system allowing for both automatically generated messages and manually written messages by other users. Automatically generated messages are sent to a student when an exercise has been corrected by or submitted to the associated teacher, or when a teacher sent a reminder that an exercise has not been submitted yet. For manual messages, the recipient list is constrained. While every user can send a message to the admin and thus the developers, students can write to their teacher, and teachers can write to their students. An often requested, but not implemented feature, is the ability to add friends and have real-time chat. Whenever a message is created for a user, a news feed pops up and is displayed on the start page of each user at the bottom right. The messages are sorted from most recent (top) to oldest (bottom) and can be deleted individually or in a batch in the news feed. Depending on the type of the message, the background is rendered in a different color. In the example in Figure 5.22, this user received a reminder to do a specific exercise (orange message), and then submitted this exercise to the teacher (yellow message). On the profile page, users have the option to automatically receive an e-mail notification when a new message is sent to them. Clicking on a message opens the message writing view with the sender of the original message as the recipient, allowing users to quickly answer.

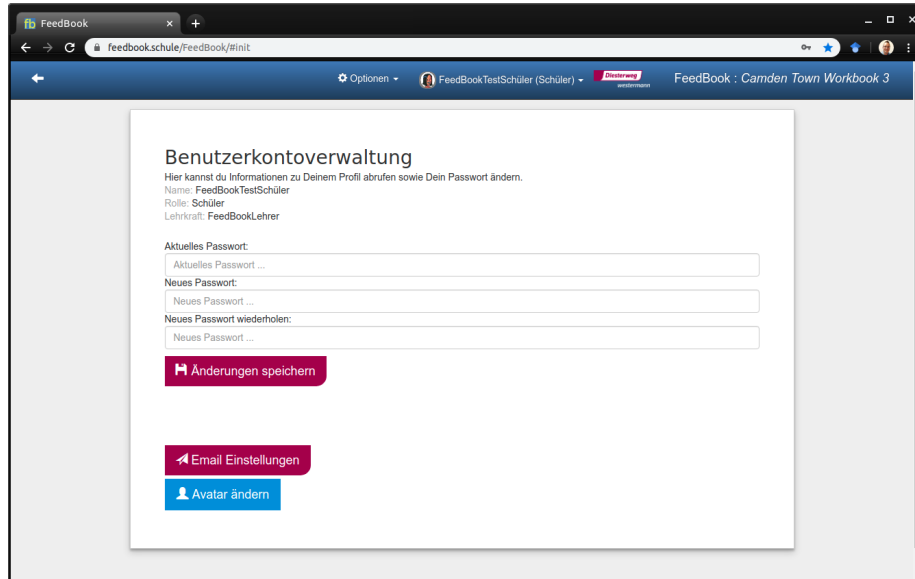


Figure 5.21: Profile page

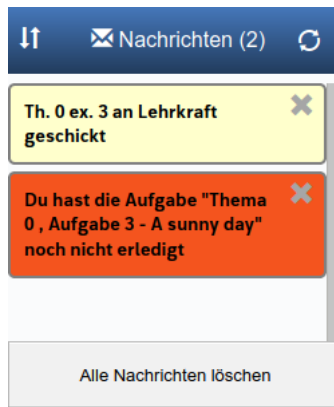


Figure 5.22: News feed with two messages: reminder (orange) and submission confirmation (yellow)

Navigation system In order to emulate a stateful experience, the system implements a navigation/routing system. In order to do this, the system stores the names of the last two interfaces a user selected. Based on this, and on explicit knowledge about which interfaces are accessible from where, the system can reload any previous pages. Clicking on the system's or the browser's back button leads the user back to the previous view(s). For interfaces requiring a lot of computations and/or network traffic, static results are cached to allow a faster reloading next time the user navigates there.

Login screen At the beginning of each session, users see the login screen shown in Figure 5.23. It prompts user to authenticate via user name and password. Also it includes a link to the support in case a user forgot their password. If a wrong password is entered 15 times in a row, the interface is locked, a message is sent to the admin, and the user is informed that they need to get into contact with the support. This measure was taken as a step to prevent bot attacks. Furthermore, the login page contains a link to the project web site for more information and a hint telling interested teachers how they can get an account.

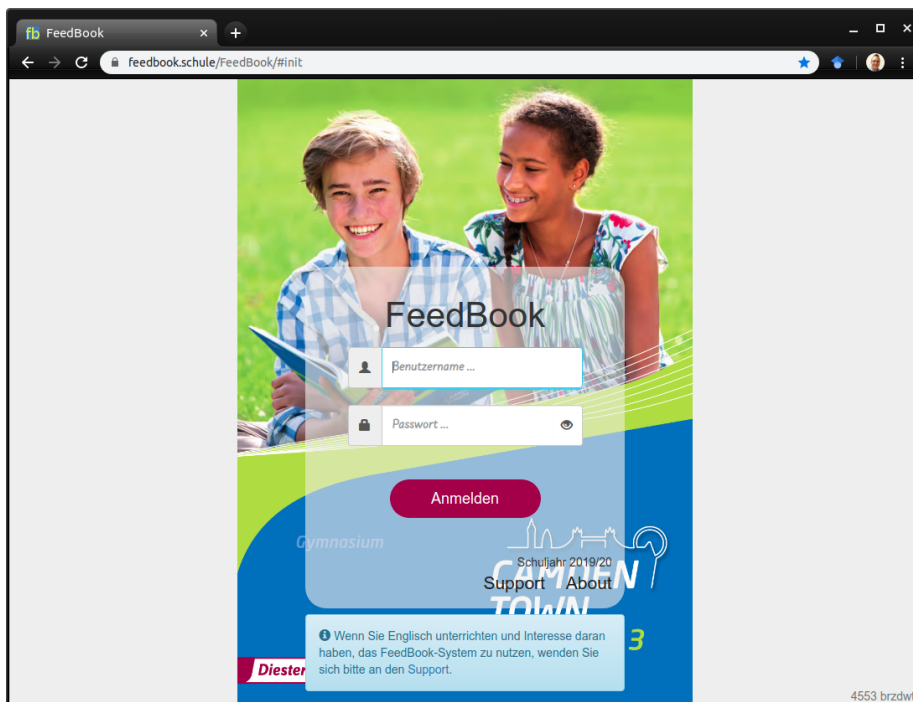


Figure 5.23: Login screen

Feedback form FeedBook was released as a research prototype under development and not a product in a final state. In other words, the system was not perfect in terms of usability and contained a range of bugs. As a means to enable users to provide feedback, we implemented a feedback form allowing users to provide feedback to the developers. While in an initial version it was a plain text field, we later adapted it so that users would need to provide more specific information about the type of problem (feature request/bug) and the specific location of the issue (e.g. which exercise). For every submitted feedback, a message in the admin's news feed as well as a message to the developers was created. Clicking in the admin interface on the feedback message allows the admin to type a response to the user who submitted the feedback, ideally to inform them that the reported problem has been solved.

Chapter 6

Interactive Real-Time Feedback in FeedBook

In chapter 2, we carved out the relevance and beneficial effects of feedback with special attention to interactive, real-time scaffolding feedback. FeedBook implements three types of feedback to provide learners with immediate feedback while working on exercises.

The system employs different feedback modules that provide feedback on different levels of representation of language. The spelling feedback (section 6.1) provides feedback on the level of individual characters. This type of feedback operates on the level of *orthography*. Form feedback (section 6.2) aims at correcting individual words with respect to grammar. This type of feedback operates on the *morphological* and *syntactic* level. The third feedback, meaning feedback (section 6.3), operates primarily on the level of entire sentences or clauses. This type of feedback operates on the *semantic* level.

Since the feedback modules operate on different levels of language representation, they do not exclude but rather complement each other. Orthographical feedback is given for tasks where by default grammar or meaning feedback is provided. It thus complements both other feedback types. Additionally, grammar feedback is provided for meaning-based activities once the learner and target answer are close enough, i.e. when quasi meaning equivalence is established.

A common trend when comparing the three strands of feedback is that all types use the task context to generate and select feedback. The feedback on orthography uses a task-specific weighting to guide the selection of target words for spelling. The grammar feedback algorithms operates by taking target answers from the task context and generating alternative forms from them. The meaning feedback highlights information sources in the task context to guide a learners' attention towards it. In the following, we will describe each of the feedback types in detail and put them into a context of related work.

Disclaimer: Specific sub parts of the contents of this chapter have been mentioned or referenced in Rudzewitz et al. (2018); Ziai et al. (2018, 2019); Meurers

et al. (2018) and are cited accordingly in the running text of the respective sessions.

6.1 Orthography

FeedBook contains a noisy channel spelling normalization component that is used in two application contexts. The first application is as a normalization module in the short answer assessment algorithm for meaning comparison (cf. section 6.3). In addition, it is used as a standalone module for providing scaffolding feedback regarding orthography to learners (Ziai et al., 2019).

In the following we will motivate the implementation of spelling feedback for learners (section 6.1.1) and explain the different error categories related to feedback on orthography (section 6.1.2). Then we will explain the noisy channel model and how it is tuned towards a specific task (section 6.1.3). In section 6.1.4, we describe the results of a validation experiment that demonstrated the impact of using a spelling normalization component for short answer assessment. We conclude the discussion by putting the approach into a context of related work in section 6.1.5.

6.1.1 Motivation

The spelling feedback approach was implemented for different reasons that we will describe in this section. The first reason is that short answer assessment approaches for meaning feedback typically work better with spelling correction component (cf. section 6.3.6 for a discussion and section 6.1.4). This insight is sustained by qualitative analyses of the data collected in the first and second year, which showed a lot of variation due to spelling errors. It became evident that a real-life usage of a tutoring system with free-text input creates a lot of variability in learner answers.

After the spelling feedback had been implemented, we made it available outside of the meaning feedback component for two reasons. First, when inspecting the data, we saw that default feedback is given frequently for cases where it would be easy for the system to provide more specific feedback and spelling correction. Secondly, this offered an opportunity to implement another feedback strategy that uses the task context to guide specific feedback, in line with the grammar feedback approach (cf. section 6.2).

6.1.2 Error Categories

The system uses two types of feedback templates to provide feedback on spelling listed in Table 6.1.

The casing template is more specific than the second template since it is only applied in cases where the system detected a match between a learner answer word and a reference answer word when ignoring case, but not when conducting a case-sensitive match. The second template is used for other types

feedback template	message
CASING_DEFAULT	There is a problem with capitalization in your answer.
SPELLING_DEFAULT	The following words in your answer seem to contain spelling errors: $\${SPELLING_ERROR}$

Table 6.1: Spelling error feedback templates and messages.

of spelling errors that do not fall into the first category. It contains a placeholder $\${SPELLING_ERROR}$. This placeholder is filled at runtime with the first word in the learner answer that contains a spelling error. In the first implementation of the system, all words with spelling errors were inserted into the placeholder separated by commas, but later a break was added to only give feedback to one word at a time.

6.1.3 Using the Task Context for Feedback on Orthography

The spell checking approach in FeedBook is based on the noisy channel model proposed by Brill and Moore (2000) enhanced with a task-specific weighting, described in the following. The implementation is based on a reference Java implementation by Adriane Boyd¹.

The noisy channel model is an approach that assumes that noise is introduced between what a person is trying to express and what this person writes.

While the model's task is to correct misspelling of individual words at a time, and not of a list of words, it nevertheless uses contextual information for returning the most appropriate correction.

The algorithms starts by generating all possible partitions inside words, i.e. splitting the characters of a word into groups of different lengths. Then it computes a conditional probability of corrections, not on individual characters, but on the chunks/partitions of the word. It computes an edit distance by counting the edit operations needed to correct the source to the target and ranking the corrections by the minimal edit distance. Positional information is included in this process by modeling edits not in isolation, but as adjacent edits. Furthermore, the approach incorporates a trigram language model that models the probability of the word sequence looking two words to the left to estimate how likely it is that the given correction appears in the context to the left.

In the noisy channel approach, the lexicon used in the system consists of two sets of words. On the one hand, there is a set of valid words, i.e. words that the algorithm is supposed to suggest. In the FeedBook system, we use the list of words provided in the school books of grade 5, 6 and 7 as the set of valid words, extracted from the vocabulary lists in the school books. As described by Ziai et al. (2019), we use the school book data to approximate the vocabulary known by the learners at this stage. This however only represents a minimum baseline

¹<https://github.com/adrianeboyd/BrillMooreSpellChecker>, last accessed 2020-06-02

since students most likely know more words in English from other domains via exposure to the language also outside of the classroom.

On the other hand, the approach needs a list of words that were misspelled together with target corrections. For this, we use data from the EFCamDat corpus (Geertzen et al., 2013). The data sample consists of approximately 10,000 misspellings with corrections, with the misspellings produced by German learners of English.

We tune the spell checker towards specific tasks in FeedBook by ranking candidates for corrections dynamically for every task. This works by assigning a weight of 1 to each (correction) word by default. This default weight is, however, increased if the word appears in the language material occurring in the specific task. The system then assigns a weight corresponding to the term frequency of the word in the associated reading text or transcription of the listening text.

This leads to a prioritization of the words occurring in the task context, making the spell checker task aware. Just as for the grammar feedback 6.2.2, the idea is to use the task context to provide feedback that is appropriate and dynamically tuned towards the current task the learner is working on.

6.1.4 Validation Experiment

Ziai et al. (2019) conducted an experiment to test the validity of the spelling correction approach. In this experiment, they tested the impact of using a spelling normalization for short answer assessment in the context of meaning feedback (cf. section 6.3).

They extracted learner data for all reading and listening comprehension tasks in the system. For 25 tasks with a total of 123 items, they extracted 3,829 unique answer types. Each answer type was rated as acceptable or not acceptable by an English teacher working in the project. Ziai et al. (2019) report a majority baseline of 62 % for the correct class.

In their experiments, they tested the impact of spelling correction by comparing the system performance with and without a spelling normalization. In order to derive generalizable results, they constructed three scenarios: testing with unseen answers, testing with unseen items, and testing with unseen tasks. The results show that the short answer assessment system with spelling normalization outperforms both the majority baseline and a system variant without spelling normalization. The results show that the positive accuracy impact of spelling normalization is biggest when testing on out-of-domain data in the scenario of unseen tasks. They also found out that task properties such as task format, task media type (text/audio), and expected answer length influence the effectiveness of the spelling normalization. Overall, the results show a clear positive effect of spelling normalization for the short answer assessment component in FeedBook.

6.1.5 Comparison with Existing Approaches

The origins of the noisy channel model and probabilistic models for spelling correction date back to the early 1990s (Kernighan et al., 1990; Church and Gale, 1991). Various extensions of the noisy channel model have been proposed to add more information sources, e.g. by including phonological information (Toutanova and Moore, 2002).

Another direction of extensions of the approach is described by Duan and Hsu (2011), who use a noisy channel normalization and prediction algorithm that goes beyond simple edits and explicitly models different types/sources of misspellings like keyboard adjacency, quick typing, inconsistencies between orthography and pronunciation, and more categories.

This line of research is further pursued in spelling correction approaches for search engine queries. Similar to the case of tutoring system, search engines need to deal with spelling errors by its users (e.g. Li et al. (2006)). The performance of such approaches is important in that feedback or corrections need to be very efficient in order to provide interactive and immediate feedback. In the context of search engines, like for tutoring systems that perform short answer assessment, the system needs to abstract from the form to the semantics of the query/answer. What is different between the application of spelling normalization in tutoring systems and search engines is, however, that for search engines, often not only normalization, but also the prediction of subsequent words is required.

Turning towards tutoring systems, spelling correction has been implemented in tutoring systems, but very little literature is available for it. Elmi and Evens (1998) implemented spelling correction in a tutoring system for medical students to provide real-time spelling correction and normalization with a special emphasis on the spelling of medical terms.

A problem is that language learners make different types of errors compared to native speakers. Rimrott and Heift (2008) showed that out-of-the-box spell checkers for native speakers do not cover the majority of spelling errors produced by language learners. Learners and native speakers commit substantially different spelling errors, since learners do not 'only' produce pure spelling errors, but often a mix between competence errors and spelling errors/mistakes.

In a study involving E-Tutor (cf. section 4.4.2 for a discussion), Heift and Rimrott (2008) tested different types of spelling feedback with different degrees of explicitness. Their experiments tested spelling feedback with or without suggestions for corrections, and with or without highlighting of the error location. They found out that more explicit feedback with highlighting and suggestions was most helpful for learners. Their data showed more uptake for feedback that included suggestions than for feedback without suggestions. Furthermore, they observed more correct learner responses for cases where the target word was in the list of suggestions.

Another line of research in conjunction with tutoring systems and spelling correction aims to put teachers into the loop. Laarmann-Quante (2017) describes a spelling error detection and classification tool geared towards teachers

to inspect the types of spelling errors that were produced by their students. The approach is not integrated in an actual tutoring system, but offers an interesting perspective since the target group are teachers, not students. The authors de Haan and Oppenhuizen (1994) present the *SPELLER* tutoring system that offers suggestions for corrections to learners in an interactive way. Their approach includes an early form of Learning Analytics that informs teachers about the individual keystrokes of their learners and how they interacted with the spelling correction mechanism.

6.2 Grammar

The grammar feedback algorithm in FeedBook is built on two pillars: an offline pre-generation and analysis part, and an online part that matches pre-computed analyses with learner answers (Rudzewitz et al., 2018).

The key idea is to use the task context for a pre-generation of the space of well-formed and ill-formed alternatives to avoid online processing of highly ambiguous learner language. Concretely, the system analyzes the target answer of a task in the context of the other language material provided in this task and generates alternatives of it based on errors observed in learner answers to exercises in the workbook. Since target answers represent native language productions, we avoid having to process interlanguage in the live system since when errors are introduced as part of the alternative answer generation process, the modification is explicitly encoded in a transformation rule and the resulting analysis is unambiguously determined by the rule.

Another reason to split the feedback process into two parts is to outsource part of the computational complexity from a live system to an offline generation approach. In a pre-generation algorithm, the entire computing power of the server can be used to generate analyses. In a live system, in contrast, the computing power needs to be split between many requests via concurrency and each user only receives a small share of the system's power.

In this section we start by talking about the development of error types used in the feedback (section 6.2.1), motivate the approach further from the perspective of the task context (section 6.2.2), before we talk about the offline generation part in detail. This includes the linguistic analysis and application of transformation rules (section 6.2.3), the combination of rules to rule chains (section 6.2.4), and the association of feedback templates with generated answers (section 6.2.5). For the online part we explain the flexible matching (section 6.2.6) and display of feedback, before we conclude with a comparison with related work (section 6.2.8).

6.2.1 Development of Error Types

The error types used in the system were developed based on two sources of information: paper work books and learner data from the first system usage.

The development started off with an analysis of a set of paper workbooks with learner answers and teacher corrections (Rudzewitz et al., 2017). The first set of error categories derived from there were first integrated in the teacher interface (cf. 5.2.6). In a second phase, we combined the workbook data with learner answers entered into the system by test classes in the first and second year of the project. Based on this data, a teacher employed in the project derived specifications for rules.

These specifications consisted of the learner answer, the target answer, a feedback code, and a feedback message. Based on the specification, we built transformation rules that transform the target answer into the specified learner answer and associate the given error code with it. We however did not only write rules that exactly work for the one example provided, but instead abstracted from the concrete example to a set of similar cases in a step of generalizing the errors from examples to rules that abstract and capture multiple examples. For example, the specification consisted of the confusion of certain selected tenses – we then wrote rules for the confusion of all possible tenses.

The final set of 88 fine-grained error categories thus constitutes a combination of bottom-up driven categories (what errors did students actually make on paper) and top-down driven categories (e.g. confusion of all tenses).

6.2.2 Basis for Feedback: Using the Task Context

The transformation rules use the target answer as the basis upon which analyses are conducted. As a first step, the system creates a text from the target answer plus the prompt context (e.g. the surrounding text in a fill-in-the-blanks activity). Both the prompt text and the target answer constitute the task context, since they stem from the exercise definition.

This combined text is stored in a UIMA JCAS (Java Common Analysis Structure, Götz and Suhre (2004)) object. A JCAS is a Java container around a CAS, which in turn is a wrapper around UIMA multi-layer standoff annotations. The annotations are defined in a type system. In FeedBook, we use the DKPro type system (de Castilho and Gurevych, 2014) enhanced with custom types. Building on DKPro makes the components interoperable and modularly exchangeable.

Once a JCAS is created, we run the NLP tools listed in Table 6.2 on the target answer and prompt. The analyses add word and sentence boundaries, part-of-speech tags, dependency annotations, lemmata, and morphological analyses. After these analyses provided by DKPro, we run over 40 KeyValuePair annotators to annotate properties that are shared across rules and that extend the existing linguistic analyses with specialized information like for example the type of comparative, whether words have regular or irregular tense forms, or signal words for specific tenses.

An important design feature of the feedback generation algorithm is that it analyzes language that is well formed and according to the grammar rules of the target language (cf. also Meurers et al. (2018)). This circumvents the problem that when analyzing learner language with errors, there is inherent ambiguity

task	tool
segmentation	ClearNLP (Choi and Palmer, 2012)
part-of-speech tagging	ClearNLP
dependency parsing	ClearNLP
lemmatization	Morpha (Minnen et al., 2001)
morphological analysis	Sfst (Schmid, 2005)

Table 6.2: NLP tasks and tools, taken from Rudzewitz et al. (2018)

as to how to interpret it. As discussed in section 4.2.1, for ill-defined domains like language, analysis tools developed for native language only work reliably for input in the native language. Given that the feedback generation algorithm always starts off with an analysis of native language, the tools are more accurate than in approaches where actual learner input needs to be processed.

6.2.3 Implementation of Transformation Rules

The target answers analyzed in context (section 6.2.2) are used as input to transformation rules that change one aspect of them. Each transformation rule inherits from an interface class that defines two obligatory methods: `isApplicable` and `apply`.

The `isApplicable` method returns a Boolean value and contains the logic to determine whether this rule is applicable for a given JCAS. By extracting `KeyValuePair` annotations, this function is used to ensure that the rule is only applied to JCASes in which the structures targeted by this rule are present. For example in tense substitution rules, the `isApplicable` function checks whether the tense to be substituted is present in the given input configuration.

If and only if the `isApplicable` returns true, the rule is applied. The `apply` function takes as input a JCAS and returns as output a JCAS again. The JCAS returned is not the same object, but a deep copy modified with one single change introduced by the rule. It does, however, contain a reference to its predecessor JCAS to make the rule application chain traceable. The change introduced by the rule conceptually amounts to introducing an error or variation pre-envisaging a response pattern by a learner. Each rule has a different logic for introducing errors, but most rules are configurable with parameters to make them general enough to model several related regularities. For example in the aforementioned tense substitution rule, it is possible to determine the source tense and the target tense, allowing this rule to be used for substitutions of arbitrary tense pairings.

The `apply` function not only edits the source text of a JCAS, but also adapts all the associated layers of linguistic annotation layers. Since there is a symmetry between input and output, and given that the output of one rule can be used as input to other rules, the rules need to ensure that their resulting analyses are

as complete as possible (with some difficulties arising in assigning annotations designed for native language to generated interlanguage).

There are two strategies for generating linguistic analyses of edited JCASes. For the output of rules which generate well-formed alternatives, such as for rules expanding contractions or substituting tenses, we perform a re-analysis of the edited part. In order to do this, we first re-analyze the entire edited JCAS document text in a temporary JCAS from a pool of temporary JCASes held back for this purpose. Then we copy all annotations that overlap with a window defined over the edited part from the temporary JCAS into the original deep copy, after which the temporary JCAS is reset and released back into the JCAS pool. This step is necessary since JCASes are not garbage collected and we can not create arbitrarily many new JCASes in memory. The idea of cloning and using JCASes in pools does not seem to have been an intended use case, but with the described tweaks it does the job.

The alternative strategy, namely to manually add new annotations as part of the rule logic, is only chosen when the rule yields a non-well formed source text, such as in a rule that constructs a regular past tense form for irregular verb forms. Since standard NLP tools can not handle these cases reliably, we manually encode the linguistic annotation layers for these forms to the extent that they are applicable.

The implementation of rules that are able to feed other rules is inspired by configuration-based parsers and the graph search algorithms implemented in them. The key insight and design pattern is the symmetry between the input representation and the output representation.

6.2.4 Combining Transformation Rules in a Configuration-based Transformation Algorithm

Transformation rules are organized in layers that control the order of their application. As shown in Table 6.3, there are four rule layers.

The **first layer** contains rules that do not add diagnoses, but generate alternative well-formed variants of the target answer, primarily the expansion of contracted forms. In the **second layer**, rules are put that primarily generate forms that are well-formed in the target language system, but are not appropriate in the given task context, such as a tense form. These rules are put in the second layer in order to allow the system to generate ill-formed derivatives from it. Rules that generate forms that are not well-formed are placed in the **third layer**. Output from this layer is never grammatical in the target language system. Finally, the **fourth layer** contains rules that generate normalized forms again, given that some forms have been expanded in the first layer.

Every non-final layer contains a *SelfCopyRule*, which is a rule that simply generates an identical deep copy of the input JCAS. This allows the original target answer to percolate through all layers, allowing rules in later layers to be applied to the original target answer. If this was not the case, the process would stop early if a rule is unable to pass through earlier normal rule layers.

As the starting point, the system attempts to apply the rules in the first layer. At least the *SelfCopyRule* can be applied. For every applied rule, the cloned JCAS is put on a stack and available for application in later rules. This means that the output of every rule application is put back to the input stack and percolates through all rule layers, either as a direct copy or as the output of a transformation rule. Figure 6.1 depicts this process.

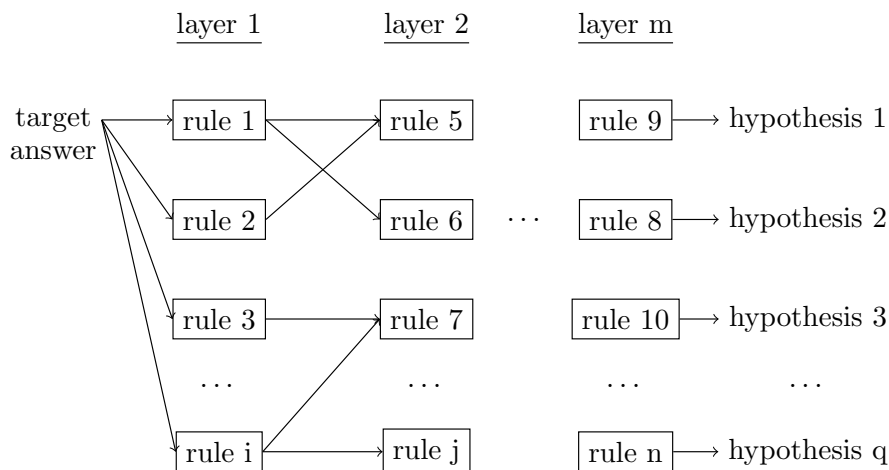


Figure 6.1: Multi-layered hypotheses generation process, taken from Rudzewicz et al. (2018)

For every output of a rule application, the system associates a feedback template and saves the result in the data base (cf. section 6.2.5). The first version of this algorithm was implemented as a single-threaded application with a stack of JCASes, but later we changed it in a second version to a scalable multi-threading implementation with a JCAS pool.

The approach is similar to breadth-first search algorithms in that the algorithm first tries to apply all rules in one layer before moving on to deeper layers. The algorithm thus traverses the layers first in the breadth before moving into the depth with higher layers. Conceptually, the system generates and searches a graph in a network representing the space of well-formed and ill-formed alternatives. Each reached node represents a state in which a target hypothesis is saved in the data base, i.e. the result of the successful application of a transformation rule. It avoids cycles by grouping rules into rule layers and employing a breadth-first approach to prevent that the same form is generated again, thus controlling the flow through the network.

layer 1	layer 2	layer 3	layer 4
NegationExpansionRule	WellformedTenseSubstitutionRule	VerbSuffixMalrule	ContractNotRule
ApostropheExpansionRule	Tense2BaseformRule	DoNegationTenseMalrule	ContractAuxFormRule
ContractionExpansionRule	InflectedBe2BeMalrule	ReciprocalReflexiveConfusionRule	
SelfCopyRule	AuxDeletionRule	PronounSubstitutionRule	
	Participle2BaseFormRule	WasFormConfuserRule	
	Participle2SimpleTenseRule	NegationDoDeletionRule	
	Tense2ParticipleRule	ExtraS_SimplePastMalrule	
	AuxSubstitutionRule	GerundExtraToMalrule	
	SelfCopyRule	MoreMostOmissionRule	
		DoubleComparativeMalrule	
		AdverbReductionRule	
		ExtraMoreMostMalrule	
		PronounSubstitutionRule	
		ComparativeYNotChangedMalRule	
		PresentNoAgreementMalrule	
		VerbOverregularizationRule	
		IrregularSimplePastEdMalrule	
		PronounMisspellRule	
		KeyValueWordDeletionRule	
		RelativePronounPluralPersonalPronounMalrule	
		RelativeClauseCommaMalrule	
		RelativePronounDeterminerRule	
		ExtraObjectPronounRelativeRule	
		WordDeletionRule	
		ComparativeAdverbReductionRule	
		AdverbCompExtraMoreMalrule	
		GerundMisspellMalrule	
		GerundToInfinitiveRule	
		IngFormVowelAbstrMalrule	
		GerundLooseRule	
		GerundExtraPronounInsertionRule	
		AdjectiveCompExtraMoreMalrule	
		ComparativeAdjectiveReductionRule	
		AdjectiveCompDoubleConsReductionRule	
		AdjectiveCompLongReductionRule	
		WordSubstitutionRule	
		InfinitiveTenseMalrule	
		AdjectiveBadExtraMoreMalrule	
		AdjectiveDeterminerComparativeMalrule	
		ComparativeAdjectiveSuperlativeThanMalrule	
		PronounMisspellRule	
		VerbOverReflexivizationRule	
		SelfCopyRule	

Table 6.3: Feedback generation rules ordered in layers

6.2.5 Computation of Template Specificity and Ranking of Candidates

The application of a transformation rule can yield two cases: for rules that introduce errors, a diagnosis is associated as an annotation on the edited part. For rules that do transformations such as contraction expansions, no diagnosis is associated, and the answer is treated as correct by the system.

Diagnoses contain both a diagnosis span and a list of side conditions. The side conditions can be empty, but if they are present, the specificity of the respective diagnosis is higher than if there are no side conditions. For certain phenomena, such as tense confusions, there are feedback templates with different specificity. Figure 6.2 shows a feedback template whose target form is simple past and the diagnosed form is simple present. It does not specify side conditions and is thus an error code used for tense confusions. Figure 6.3 shows a template with the same target and diagnosed form, but additionally a side condition that specifies that this template is only applicable if the diagnosed form is part of an if-clause. In this case, the system tells the learner about a rule of how to form if clauses. The same problem was diagnosed, but in a more specific context, and therefore the system can give more specific feedback.

Target form:	SIMPLE PAST
Diagnosed form:	SIMPLE PRESENT
Side conditions:	
Feedback message:	“This is the simple past You need to use the simple present here.”

Figure 6.2: Example error template

Target form:	SIMPLE PAST
Diagnosed form:	SIMPLE PRESENT
Side conditions:	IF-CLAUSE
Feedback message:	“With conditional clauses (type 2), we use the simple past in the if-clause, not the simple present.”

Figure 6.3: Example error template, taken from Rudzewitz et al. (2018)

During the feedback generation process, the system always associates the most specific feedback template applicable. It computes a specificity score for every generated form and applicable feedback. This score consists of three numbers: a binary indicator whether the target form of the template and diagnosis match, a binary indicator whether the diagnosed form is matched, and a natural number expressing the number of side conditions matched. This allows for flexibility in terms of matching, since not all conditions need to be met, but the most specific template can nevertheless be selected. In principle, this also allows to introduce a weighting function for the specific score to give higher weight to

certain parts, e.g. the diagnosed condition, but in the FeedBook, equal weight was given to all dimensions.

6.2.6 Flexible Matching

Since the system needs to deal with variations in learner input, especially for semi-open tasks where the task context constrains the input, but learners still need to provide a response in the form of an entire sentence, it is not feasible to generate out and anticipate all possible learner answers (cf. also Nagata (2009)).

The goal here is to allow for variation in the learner answer, but still be able to diagnose learner errors. In order to achieve this goal, the system uses a Lucene search index. In this index, the learner answer is treated as a query and the generated target hypotheses as documents. A 'document' is only found, i.e. the feedback is only selected, when the text of the diagnosis span of the generated hypothesis has been found in the learner answer. The system selects the closest match between the learner answer and the generated hypothesis by weighting words in the diagnosis span higher than words in the surrounding learner answer, telling the search index to prioritize generated hypothesis for which the maximum number of words associated with a diagnosis are matched. This in essence results in a search index for which the diagnosis is an obligatory part and the remainder an optional part, which is taken into account, but with priority given to diagnosed words. Technically, this is realized via Lucene's query boosting functionality, assigning a value of 5 to a diagnosed part of the learner answer and a value of 1 to the remaining text (Ziai et al., 2018). Once the most likely candidate is returned, the system stops and returns the template associated with the generated form. In an early version of the system, the Lucene index operated not only on surface forms, but also on the basis of soundex matches. This was however not as reliable and later disabled.

CYP 2 Grammar check: Problems
Everyone has got problems. What could these people do differently?

0. Gillian is sad. Her mother never has any time for her.
If Mrs Collins had more time for Gillian, Gillian wouldn't be sad.

1. Mrs Collins feels bad. She should listen more to Gillian.
If she listens to Gillian, she feels better

2. Gwynn is very disappointed. Gillian doesn't like Wildings

Feedback für "If she listens to Gillian, she feel..."

With conditional clauses (type 2), we use the simple past in the if-clause, not the simple present.

If she **listens** to Gillian, she feels better

Hilfreich?
 Ja Nein OK

Target answer (for reference):

If Mrs Collins listened more to Gillian, she would not feel so bad.

Figure 6.4: Student answer including multiple errors with feedback based on a partial hypothesis match, taken from Rudzewitz et al. (2018)

In Figure 6.4, the learner wrote "If she listens to Gillian, she feels better."

The target answer is “If Mrs. Collins listened more to Gillian, she would not feel so bad.” The learner used a pronoun (“she”) instead of a proper name, confused the tense form (“listens” instead of “listened”), omitted the word “more”, and wrote a different sentence in the main clause (“she would feel better” instead of “she would not feel so bad”). Despite all these changes, the system was able to recognize the tense confusion in the if-clause and provide specific feedback about it to the learner.

6.2.7 Combining the Functionality

Figure 6.5 shows a pseudo code version of the feedback computation algorithm. The system first performs a very efficient direct lookup check to determine if there is a direct match between the student answer and a target answer. It does also take generated hypotheses into account which do not have an associated template. If a match is found, the algorithm stops and tells the learner the answer is correct. If not, the system first performs another efficient lookup to see if there is a generated target hypothesis which has exactly the source text the learner provided as input. If this is the case, which frequently happens for fill-in-the-blanks activities, the system directly returns the feedback template associated with this generated form. Only if both the previous checks fail, the system performs flexible matching. In this case, the system searches for the best match and returns the feedback associated with the highest ranked generated target hypothesis.

```
if student input == target answer:
    visualize this with green check mark
    -> DONE
else:
    retrieve direct hypothesis matches
    if there are direct matches:
        show associated feedback
    else:
        perform token-level Lucene query
        if there are Lucene hits:
            for every hypothesis:
                if student answer matches criteria:
                    show associated feedback
            else:
                show default feedback
```

Figure 6.5: Feedback algorithm (simplified pseudo-code), taken from Rudzewitz et al. (2018)

6.2.8 Comparison with Existing Approaches

The traditional paradigms for generating feedback for a specific task consist of (a) mal-rules, (b) constraint relaxation approaches, and (c) pre-generation algorithms.

For the first approach, mal-rules, feedback is generated by running a parser on the learner input. For parts that can not be matched with standard rules, the algorithm introduces mal-rules which model non-standard patterns. The result of this algorithm is a parse tree of the learner answer with diagnostic information from the applied mal-rules. This approach requires a structural live analysis of the learner answer. As explained in section 4.2, the inherent ambiguity in learner language and the resulting differences in the analysis makes it difficult to arrive at an analysis or interpretation of learner answers that is as complete as in a mal-rule approach. In comparison, the FeedBook grammar feedback mechanism only requires a shallow surface analysis at run time. While structurally less rich, the potential for errors is reduced and the complexity outsourced to analyses of generated target hypotheses based on the analysis of native language in the form of the original target answer used as input to the transformation rules.

For the second case, constraint relaxation approaches, a similar pattern arises. A system based on such a paradigm attempts to analyze learner language by parsing it with standard rules, but relaxing some constraints if a student answer can not be analyzed successfully. Again, this requires live processing and implicitly assumes that the analysis from which constraints are relaxed is a sufficiently accurate representation and starting point for the constraint relaxation procedure. This assumption, however, can not always be assumed to be met due to the nature of interlanguage. As for case (a), FeedBook circumvents this problem by basing analyses on native language with controlled, explicitly modeled derivations for which the changes are made evident, and with a minimal live processing aspect in the system usage.

The third paradigm, pre-generation, is most similar to the approach described in the FeedBook. A pre-generation approach uses the target answer and thus the task as a starting point and anticipates possible alternative variants from there. The novel feature described here, and the feature that makes such an approach work in an authentic scenario, is the implementation of a flexible matching component. The system does not use the pre-generated forms as the solution, but as input to a matching algorithm, thus offering the flexibility that most hinders traditional pre-generation algorithms.

The FeedBook pre-generation algorithm integrates structural information in the form of a richly annotated JCAS, similar to information retrieved from approaches like mal-rules or constraint relaxation, but combines it with a pre-generation algorithm that outsources a large part of the complexity to an offline generation part. Its design offers a strong integration of the task context and rich analyses while reducing the required live processing to a minimum. As a hybrid drawing from all three approaches, it combines strengths from all three paradigms.

6.3 Meaning

FeedBook contains an extended version of the CoMiC (Meurers et al., 2011; Ott et al., 2012, 2013) meaning diagnosis component (Ziai et al., 2018). Originally developed in the SFB 833 A4 project, it was ported over and integrated into the FeedBook system. The A4 project created the theoretical and technical foundation for the alignment-based approach.

The design of the algorithm prioritizes meaning over form with respect to learner answers given as a response to semi-open questions. While the task constrains the space of semantic variability to a certain extent, the system needs to be able to deal with more variability on the morphological, syntactic, pragmatic, and semantic level. The approach is thus a short answer assessment (Ziai et al., 2012; Burrows et al., 2015) algorithm. Different from the grammar feedback, the algorithm performs a live analysis since it is not feasible to anticipate all possible learner answers in contexts with semi-open tasks.

The task of the meaning diagnosis component is to evaluate if a learner answer and a target answer are semantically equivalent or close enough to accept them as quasi-equivalent, and, if not, diagnose what kind of divergence there is between the learner and the target answer. The algorithm is capable of diagnosing missing concepts, extra concepts, and concepts that are not equivalent by aligning a learner answer with a target answer on different levels of linguistic abstraction.

While the algorithm prioritizes meaning over form, the two approaches do not exclude each other. Once the learner and target answer have been evaluated as semantically equivalent, the system attempts to diagnose form errors (cf. section 6.2) in a component for incidental focus on form.

The system offers three types of assistance to learners in conjunction with the meaning feedback. The first assistance comes in the form of an information source highlighting with different degrees of specificity. If a learner answer is not semantically equivalent to a target answer, the system replaces the default media (task text image or task audio clip) with a modified version with manually highlighted information sources. Concretely, for reading comprehension tasks, it replaces the photo of the task text with a version in which a teacher has added color markings for the part of the text that contains the answer to the current question. For listening comprehension tasks, the audio file containing the listening text recording is replaced by a shorter version that contains only the part which answers the current prompt. A second type of support is a visual highlighting of the parts of the student answer which diverge from the target answer (cf. also section 5.2.6). This allows learners to see which part of their answer needs to be changed. The third type of assistance is the textual representation of the feedback template and thus meaning diagnosis, telling the learner the result of the meaning comparison with the target answer.

In the following sections, we describe the approach in detail. Firstly, we explain how the linguistic analysis of a learner answer is conducted in the live system (section 6.3.1). In section 6.3.2, we turn to the alignment algorithm, which serves as the basis for the algorithm that diagnoses the type of divergence

(section 6.3.3). We then discuss how the generated feedback is displayed and enhanced with a display of information sources for the current task (section 6.3.4). Finally, we explain how form and meaning feedback can be combined in an incidental focus on form case (section 6.3.5) before concluding by discussing related work from short answer assessment (section 6.3.6).

6.3.1 Computational Linguistic Analysis of Learner Answer

For meaning-based feedback, the system performs a live analysis of the learner answer and extends the analysis of the answer to which it is compared. Since the system pursues an alignment-based approach on multiple levels of linguistic analyses, the first step is to add annotations on all levels of linguistic abstraction needed for the alignment (cf. section 6.3.2).

Table 6.4 lists the Natural Language Processing tasks and tools used in the meaning feedback analysis, listed in the order in which they are run. The first part of the pipeline up to the morphological analyzer is shared with the grammar feedback analysis and described in section 6.2.2. The next tool in the pipeline is a component that, for each word in the input, annotates the synset, listing the synonyms of each word extracted from WordNet (Leacock and Chodorow, 1998). Next, the system uses the OpenNLP chunker (Baldrige, 2005) to annotate multi-word units. In the next step, we use a component ported over from the CoMiC system (Meurers et al., 2011). It extracts keywords in the form of dependency heads extracted from the dependency parse tree annotated via ClearNLP (Choi and Palmer, 2012). The last step is to annotate word embeddings via ELMO (Peters et al., 2018). This is a distributional representation serving as an alternative, statistics-based representation of the meaning of a word – parallel to the manually annotated synonyms like in WordNet.

task	tool
segmentation	ClearNLP (Choi and Palmer, 2012)
part-of-speech tagging	ClearNLP
dependency parsing	ClearNLP
lemmatization	Morpha (Minnen et al., 2001)
morphological analysis	Sfst (Schmid, 2005)
synonym annotator	WordNet (Leacock and Chodorow, 1998)
chunking	OpenNLP (Baldrige, 2005)
keyWordFromDepExtractor	CoMiC (Meurers et al., 2011)
word embeddings	ELMO (Peters et al., 2018)

Table 6.4: Meaning-based feedback pipeline, originally adapted from Rudzewitz et al. (2018), but extended with meaning feedback analysis components

The second part of the NLP analysis is also performed on the reference answer selected as an alignment partner to the student answer since the stored linguistic analysis (UIMA CAS) of the target answer does not contain these

more abstract levels of analysis. Since these layers are needed to compute the similarity between the learner and target answer, these annotation layers need to be computed in the live processing step.

6.3.2 Alignment-based Matching of Target And Learner Answer

In order to align the learner answer to a reference answer, the first step is to select a reference answer to which to align to. In the most straightforward case, the reference answer is the target answer. Due to the high variability and the task-based approach of many exercises in the workbook combining functional goals with specific forms to realize these functions, the system needs to implicitly take into account form variants also in meaning-based tasks.

The system employs a two-stage strategy for selecting a reference answer. The goal of this process is to find the closest matching pre-analyzed answer in the system. First, it checks if the data base contains a reference answer that is string-identical to the learner answer. This includes the target answer as well as generated alternatives. Since the generated alternatives are variants with form changes, but only very limited meaning changes (for example meaning changes as a consequence of tense changes), the system does not restrict the alignment to the target answer only. If the system can align the learner answer to a reference answer, the system can interpret the learner answer more comprehensively. The system contains the option to perform the spelling normalization described in section 6.1 on the learner answer before performing the lookup, thereby allowing to account for more variability (Ziai et al., 2019).

Since the direct lookup is a very efficient data base operation, the system quickly returns a reference answer in this scenario. In many cases however, especially for semi-open tasks, the system is not able to find a direct match. The algorithm in this case uses a flexible matching approach with diagnosis weighting (cf. section 6.2.6 for a detailed description) to select the interpretable reference answer in the system which is closest.

Once a reference answer is selected, the system uses the NLP tools listed in the second half of Table 6.4 to add the linguistic annotation levels not present in the stored CAS analyses of the reference answer.

The annotations are used as input to a multi-level alignment process. In the first step, the algorithm defines which words are alignable and which ones are ignored. By adding a custom annotation type 'Mappable Annotation' to the CAS of the learner and target answer, the system initially marks all words, chunks, and dependency triples as alignable. The mappable annotation constitutes a 'shadow' annotation added behind every token, chunk, and dependency triple.

The system applies a punctuation filter to exclude punctuation from the alignment. As the next step, a 'naive' givenness filter is applied which excludes words from the alignment which are given in the prompt text by comparing their surface forms (cf. e.g. Ziai and Meurers (2018) for a discussion). Furthermore,

this component allows to filter pronouns and determiners, with only the pronoun filter active.

Since the alignment is conducted not only on the token, but also on the chunk and dependency level, both the punctuation filter and the naive givenness filter allow to exclude chunks that contain words which are excluded from the alignment. All words which have been marked as alignable are aligned on the lowercase surface level, on the surface level, lemma level, synonym (WordNet) level, and embedding level (ELMO). For each word in the target answer that can be aligned to a word in the learner answer, a potential match is marked. This configuration of potential matches is used as input to the Traditional Marriage Algorithm (TMA, (Gale and Shapley, 1962)) to select a globally optimal configuration in which each annotation is aligned to only one other annotation and the configuration, in which most words are aligned, is selected.

After the token alignment, the chunk aligner is applied, which takes as a basis chunk annotations and aligned words. For each nominal, adverbial, adjectival, or verbal chunk, it computes a score between the student and target answer chunk based on the similarity of the chunk lengths and the number and type of token alignments covered by the chunk annotations. A chunk which has a similar length and many alignments to the other chunk's words on less abstract levels (e.g. surface level) gets a high score. Again, the TMA is used for selecting a globally optimal chunk alignment configuration.

Finally, the dependency aligner takes all argument relations and checks if the tokens corresponding to the head and dependent are aligned to each other. If yes, the system adds a dependency mapping annotation to the learner and target answer.

6.3.3 From Alignment Configuration to Feedback Label

Based on the computed alignment configuration, the system determines whether an answer is accepted as correct, or if not, what type of problem there is. An answer is marked as accepted if there are no unaligned target tokens, no extra material in the form of open word classes (nouns, verbs, adjectives, adverbs) in the learner answer, and if there are no unaligned negation words in the learner answer. The last check is needed because in the case when a learner writes exactly the target answer with the prefix “not”, this would yield a perfect alignment from the target answer side since “not” does not fall into open word classes and does thus not hinder a perfect alignment score. To counter this, we added an extra check for negated statements.

As a next step, the system runs two rules to detect the type of divergence between the learner and the target answer. The *MissingConceptRule* checks if for open word classes, there are words in the target answer that fall into these classes and that are unaligned. The *ExtraInformationRule* is applied on the learner answer to check if there are words belonging to open word classes that have not been aligned to words in the target answer. For efficiency reasons, as soon as one of the rules finds a divergence, the system returns a diagnosis which is used to inform the learner about the problem.

If the answer is semantically accepted by the system, the system checks if the reference answer is a generated alternative with a diagnosis. If so, it generates a diagnosis for the form problem (cf. section 6.3.5). If this is not the case, the system checks if there are any spelling errors in the learner answer (cf. section 6.1), and if so, creates a diagnosis relating to the first diagnosed spelling error.

This approach of diagnosing a problem and providing feedback on it is conceptually different from the CoMiC approach (Ziai et al., 2018). CoMiC used a memory-based learner (Meurers et al., 2011) to derive a binary classification of the semantic correctness based on examples. This machine learning approach is not pursued here, but replaced with a rule-based approach for generating feedback diagnoses.

6.3.4 Displaying Meaning Feedback

Once a meaning diagnosis has been computed, the system displays it along with two additional types of assistance for the learner. Figure 6.6 shows different stages of meaning feedback with the exercise (frame 1), different stages of the highlighting of information sources along with diagnoses (frame 2 and 3), incidental focus on form (frame 4), and finally the correct answer (frame 5). In the following, we will explain these stages.

Meaning Diagnosis Display The system informs the user about what specific meaning divergence it has detected. This can be missing information, extra information, or non-equivalence of certain parts. The feedback templates are loaded and displayed in the same way as the grammar feedback, allowing a seamless integration of the meaning feedback into the grammar feedback frontend developed before it.

Error Location Highlighting The integration of the meaning feedback algorithm into the existing frontend means that it can benefit from the error location highlighting implemented in there. It computes a diff between the learner answer and the student answer and highlights those parts that are divergent (cf. section 5.2.5 for an in-depth discussion). In the case of incidental focus on form, the form diagnosis location is highlighted, in case of meaning diagnoses, all parts that are different from the target answer are visually distinguished. Different from the grammar feedback, meaning feedback templates that point to extra information, verbally mention the word(s) they refer to. Figure 6.7, frame 4, shows an example of a rule that points to an additional “no” in the learner answer and mentions this in the feedback message.

Information Source Highlighting Since questions in reading and listening comprehension exercises require the learner to locate the relevant information source in the input material, the system offers support to learners for detecting where they need to look to get the relevant information for answering the

The figure displays five sequential screenshots of a language learning interface. Each screenshot shows a reading comprehension task with a text passage and two multiple-choice questions. The interface provides different stages of meaning feedback:

- Screenshot 1:** Shows the text passage and the questions. The first question is "Gwyn thinks Widdings School would be great for Gillian because" with options "It is great" and "Gillian doesn't like Widdings School because".
- Screenshot 2:** Shows the same interface with a red error message: "Feedback: It is great". A tooltip indicates: "There seems to be important information missing in your answer. Please have a look at the highlighted passage in the text." The correct answer is "It is near the coast".
- Screenshot 3:** Shows the same interface with a red error message: "Feedback: It is near the coast". A tooltip indicates: "You still missing some information. Please look at the highlighted text again." The correct answer is "His sister goes there".
- Screenshot 4:** Shows the same interface with a red error message: "Feedback: The sister goes there". A tooltip indicates: "We are talking about something that happened in the past. Please use the simple past, not the simple present." The correct answer is "His sister went there".
- Screenshot 5:** Shows the same interface with a green success message: "His sister went there".

Figure 6.6: Display of meaning feedback, with different stages of assistance: coarse information source, precise information source, positive feedback

current question. In order to offer a scaffolded feedback to the learner without giving away the target answer, the system offers a two-stage information source highlighting. If a learner answer is not semantically equivalent to a target answer, the system replaces the default media (task text image or task audio clip) with a modified version with manually highlighted information sources. For reading comprehension tasks, it replaces the photo of the task text with a version in which a teacher has added color markings for the part of the text that contains the answer to the current question. Figure 6.6 shows an example of this. In frame 2, the system highlights the first two sentences in the text with green color to draw the learner's attention to this part. In frame 3, the learner still did not submit the correct answer. In this case, the system marked with yellow color the first part of the second sentence, significantly narrowing down the information source location.

For listening comprehension tasks, the audio file containing the listening text recording is replaced by a shorter version that contains only the part which answers the current prompt. Figure 6.7 shows the different scaffolding stages for a listening comprehension task. The system first presents the learner with a short version of the original audio clip (frame 2), which is 10 seconds short instead of 2:11 minutes (frame 1). Since the learner did not provide the correct answer in frame 3, the system displays the transcription of the audio file.

6.3.5 Incidental Focus on Form Feedback

The system combined meaning and form feedback in an incidental focus on form approach. If the meaning of the learner answer is similar enough to the target answer, or, concretely in this case, if there is a direct match between the learner answer and a generated target hypothesis, the system returns form feedback for meaning-based activities.

Since the generated target hypotheses contain form errors and not primarily meaning errors (apart from some changes via e.g. tense confusions), the system displays form feedback only if there are no meaning errors. The algorithm always prioritizes meaning feedback in meaning-based activities in order to avoid that learners are exposed to form feedback early on which leads them to a semantically incorrect form, given that the task in this type of exercises is to get the meaning right.

Thus, as soon as quasi meaning equivalence is established (quasi since e.g. different tense forms impact the meaning), but the interpretation of the divergence to the target answer is still explicitly known to the system via the generation algorithm, the system provides form feedback to learners in meaning-based activities. Figure 6.6 shows this process where the system first guides the learner to the correct meaning and then provides feedback on remaining form errors.

6.3.6 Comparison with Existing Approaches

The approach described here is an application of a short answer assessment algorithm in the context of tutoring systems. Short answer assessment is the

The figure displays four sequential screenshots of a digital learning interface for a listening comprehension task. Each screenshot shows a task titled "Talking to Gwynn" with a sub-instruction to listen to an audio clip and complete six statements. The interface includes a progress bar, a "Hörbuch?" (Audiobook?) button, and "Speichern" (Save) and "Abschicken" (Submit) buttons.

Screenshot 1 (Top): Shows the initial state. The first statement is: "1. Gwynn tells Mrs Collins that Gillian needs _____ to get used to the situation." The audio player is at 0:00 / 2:11.

Screenshot 2: Shows the first statement with the word "money" entered. A feedback box appears: "Feedback for 'money': There seems to be important information missing in your answer. Please listen to the following passage again." The audio player is at 0:00 / 0:10.

Screenshot 3: Shows the first statement with "no money" entered. A feedback box appears: "Feedback for 'no money': I'm still missing some information. Have a look at the text from the passage: Mrs. Collins: She's making it so difficult for everyone. Gwynn: You must give her time to get used to the situation. Remember, she has found out a lot of new things in the last few days." The audio player is at 0:00 / 2:11.

Screenshot 4 (Bottom): Shows the first statement with "no time" entered. A feedback box appears: "Feedback for 'no time': There seems to be some extra information in your answer, have a look at the following words: no." The audio player is at 0:00 / 2:11.

Figure 6.7: Display of meaning feedback for listening comprehension task, with different stages of assistance: shorter audio clip with information source, display of transcription of audio text, negation detection

automatic evaluation of short learner responses (typically one sentence) to activities that typically require the learner to understand and adapt contents from a task's context material, such as in a reading text or listening task. It is challenging for computers to perform reliable short answer assessment since prompts used as input to short answer assessment tasks typically do not prescribe words that need to be used in the answer. Instead, learners (re-)formulate an answer based on what they read. This typically leads to variation in both the meaning and the form of learner responses. The task of a short answer assessment algorithm is to abstract away enough from the surface to allow for form variation (primarily on the level of syntax and morphology), while keeping close enough to what the learner wrote to preserve and evaluate the semantics. Short answer assessment systems thus need to establish a balance between form abstraction and preservation. Ziai et al. (2012) and Burrows et al. (2015) provide comprehensive overviews over the field of short answer assessment and different approaches. While we refer the interested reader to these excellent overviews, in the following we will discuss only the most relevant approaches related to the current work.

The approach most closely related to the present work is the CoMiC system (e.g. Meurers et al. (2011)), since the short answer assessment system in FeedBook was ported from the CoMiC system and shares its key components. The CoMiC system in turn was a re-implementation of the CAM system (Bailey and Meurers, 2008). The CoMiC system served as a basis for the exploration of the design and effectiveness of meaning diagnosis components, such as an extension to use focus to replace the naive givenness filter (Ziai, 2018) or an adaptation to more open-ended community question answering tasks (Rudzewitz and Ziai, 2015).

Horbach et al. (2013) describe a re-implementation of the CoMiC system that takes into account the reading text and information source by using the closest sentence in the reading text as a feature. This approach is highly related to the approach described here since in both cases the information source is modeled explicitly. Their approach goes even one step further in the modeling of the explicitness of the information source in that the language material of the information source is actively used in the meaning assessment part. Another shared feature of the two approaches is that the information source is manually annotated from human annotators: in FeedBook it was annotated by a teacher in the project who highlighted the information source in the reading text images and cut the audio files into smaller parts, whereas in the Horbach et al. (2013) approach three human annotators marked the information source.

Leacock and Chodorow (2003) describe the C-rater system. This approach abstracts learner and target answers to abstract, canonical representations, which are used as input to rules serving to detect divergences between the two representations. This work is similar to the present work since Leacock and Chodorow (2003) also use a rule-based approach to detect specific types of divergences between learner and target answers, comparable to the rules in the FeedBook approach that detect extra or missing information.

In the approach by Mohler et al. (2011), the authors use dependency triples

as input to an alignment-based short answer assessment system. It is related to the present work in that both approaches use dependency triples as input to the alignment. The key difference here is that the system by Mohler et al. (2011) applies transformation rules during the alignment instead of before the alignment like with the target hypothesis generation approach in FeedBook. Another parallel is that they also use a givenness filter to exclude material given in the prompt. Instead of a surface form-based filter, they use a similarity-based variant.

Short answer assessment components are rarely integrated into real tutoring systems. Most approaches remain in the prototype stage and as stand-alone applications instead of being integrated into a larger tutoring system or application. There are, however, examples of cases where tutoring systems are enhanced with short answer assessment components.

Erickson et al. (2020) describe an extension of the ASSISTments system (see 4.3.2 for a detailed discussion) with a short answer assessment component. The component is used for an automatic assessment of answer to mathematical questions requiring free-text input. They make use of the large number of answers collected with the ASSISTments system to compare different approaches. In their experiments, a random forest classifier outperforms a Rasch model and a deep learning model.

Nielsen et al. (2009) describe textual entailment in tutoring systems for a range of different topics. Their system derives facets from learner and target answers that model dependency triples and thematic roles. These facets are compared and described with different labels related to textual entailment. Another parallel to the present work is that they use a spelling correction module to correct errors in student answers as a preparation for meaning comparisons.

Chapter 7

The FeedBook Study

In this chapter, we report on the FeedBook study. Conducted over the duration of one school year, and in an ecologically valid context, this study offers a rare opportunity to conduct research on a large scale. While by its nature such an endeavor results in less controlled settings and more noisy data compared to lab-based studies, it also allows to test the generalizability and robustness of theories about learning (e.g. Renkl (2013)).

Conducting studies in authentic, ecologically valid contexts is relevant in that the vast majority of learning and teaching takes place outside of labs. While there have been encouraging attempts to move from lab-based studies to actual classrooms (e.g. Cornillie et al. (2017)), there is still leeway. Conducting studies 'in the wild' not only allows to advance the understanding about learning processes, but also to put domain-specific technologies to the test. In the case of the FeedBook study specifically, the present study allows to test computational linguistic technology where it matters: by actual users in an end-to-end testing.

To our best knowledge, the FeedBook study is the first study testing an Intelligent Computer-Assisted Language Learning system for English on a large scale in Germany (cf. also section 4.5 for a discussion of tutoring systems in authentic contexts). While there are other, commercial products that have been adopted by a larger amount of users, e.g. as side products to printed workbooks, they are not used as scientific instruments and have a much looser connection to the current state of research.

The contents of this chapter are based on Meurers et al. (2019a). This chapter is divided into two parts. In section 7.1, we describe the method of the study. The second part in section 7.2 describes the raw data collected with this method.

We start the methods section by discussing the participants in section 7.1.1. As a next step, we introduce the diverse range of materials used as a basis for learning, testing, and data collection in section 7.1.2. This includes exercises (section 7.1.2), pretests and post tests (section 7.1.2), individual difference tests (section 7.1.2), c-tests (section 7.1.2), questionnaires (section 7.1.2), and a free-writing task (section 7.1.2). To conclude this part, we discuss the procedure in

the form of research question and study design in section 7.1.3.

Following the discussion of the method, we take a detailed look at different types of collected data in the system in section 7.2 as a source of data allowing to look at learning products and learning processes. For learning process data, we describe the different types of logs in section 7.2.1. We conclude this chapter with a discussion of learning products in the form of submissions for tasks to teachers in section 7.2.2.

Disclaimer: Specific sub parts of the contents of this chapter have been mentioned or referenced in Meurers et al. (2019a) and are cited accordingly in the running text of the respective sessions.

7.1 Method

7.1.1 Participants

The participants of the study are described in Meurers et al. (2019a). In the following, we will use this article as a reference for the reported information. The study started off with 14 school classes. Two classes dropped out early on due to work overload by the associated teachers. For one school class, the parents of the students in these classes decided to stop participating in the study. Additionally, two held-out control classes were recruited, but since they would not benefit from participating in pretests and post test, decided not to take part. The final sample of school classes that participated from the beginning to end thus comprised eleven school classes.

The test classes were distributed among four secondary schools (“Gymnasium”) in the state of Baden-Württemberg (Germany), with the two held-out control classes in another school. Three schools were public, one was private. Schools differed in their foci, with one school specifically focused more on sciences and one with an emphasis on arts and sports. The average class size was 25 students. One school class happened to only include boys, the rest a mix between genders. The technology focus also varied, with two classes working with tablets the entire year and the rest only occasionally using internet-capable devices. For several school classes, teacher trainees taught the school classes for certain periods of time during the school year.

In the system, 339 student accounts were registered for the participating school classes, with 284 students in the test classes and 56 student accounts for the two control classes. However, these numbers are maximum numbers that can not directly be used for analyses given that due to the ecologically valid settings, not all students participated in all data collection rounds. For example in one test for theme 2, the internet connection in the classroom broke. As a consequence, the students had to go to other rooms, changing the test conditions and making the monitoring of them difficult. Other reasons included cases where students were sick and could not attend school on some test dates, but rarely also technical problems. In a system used over a longer period of time, with a diverse sample of students, and in very diverse contexts, missing

data commonly occurs, needs to be expected, and dealt with. For each analysis, the number of participants depends on how many students completed all the required materials.

Of the 306 students who submitted their student questionnaire data, 172 reported male gender, 123 female, and 11 did not make an indication. The average reported age of the 296 students who submitted a value was 13 years. In sum, the sample is a vertical slice comprising of a diverse set

7.1.2 Materials

In this section, we describe the materials used in the study. The materials are divided into different categories. We first take a closer look at the exercises in the system that were used by learners. Then we describe the pre- and post tests. The conducted individual difference tests are presented in the next section, followed by the description of a c-test. Following this, we describe both a student and teacher questionnaire. We conclude this part with a discussion of a free writing task.

Exercises

Table 7.1 lists the number of exercises per theme and section (cf. section 5.3 for a detailed illustration of the exercise hierarchy structure). Since many tasks consist of subtasks (part a, b, etc.), we report both the number of tasks and subtasks. Theme 0 is special in that it does not have a subdivision into sections, but all tasks are on the same level. For themes 1 to 4, there is a section “Additional Practice”, containing a range of supplementary exercises (cf. chapter 5).

Table 7.2 provides an alternative perspective on the tasks in each topic. For every theme, it lists the number of tasks where the manually assigned pedagogical target constructs annotated by the material designers matched the given construct. The table shows that theme 0 and theme 1 focus on repeating tenses. This is relevant since, as the table shows, tenses are also a topic in theme 3 and 4. This also means that the learners from the intervention group of theme 1 did not see some feedback in higher chapters, since also there the tenses that are on their blacklist are a topic. On the other hand, Table 7.2 also shows that in every topic, new grammar topics are introduced. For example, gerunds are introduced in theme 1, comparatives and relatives in theme 2, etc (corresponding to the grammar topics listed in Table 7.3).

Pre-/Post Tests

Language Topics The basis for the analysis of the pre-post test learning gain analysis are eleven language topics. Table 7.3 lists all language topics covered in the pre-/post tests of the different chapters. For each of the topics, there is an identical pre test and and a post test about this topic.

theme	section	number of tasks	number of subtasks
0		6	14
1	A	5	9
1	B	7	9
1	C	6	10
1	Check Your Progress	6	7
1	Extra Training	6	6
1	Additional Practice	39	50
2	A	4	7
2	B	7	10
2	C	6	9
2	Check Your Progress	7	7
2	Extra Training	7	7
2	Additional Practice	40	45
3	A	3	5
3	B	7	14
3	C	6	10
3	Check Your Progress	5	7
3	Extra Training	5	7
3	Additional Practice	36	44
4	A	5	8
4	B	7	9
4	C	5	9
4	Check Your Progress	6	7
4	Extra Training	6	6
4	Additional Practice	37	40
5	A	4	7
5	B	6	12
5	C	5	11
5	Check Your Progress	6	6
5	Extra Training	6	6
6	A	5	8
6	B	6	7
6	C	5	12
6	Check Your Progress	6	6
6	Extra Training	6	8
total		329	439

Table 7.1: Number of tasks distributed across themes and sections

Theme	grammar topic	number of tasks
0	question	1
0	present perfect	1
0	simple present	1
0	simple past	2
0	present progressive	1
1	gerund	9
1	present perfect	4
1	question	6
1	past perfect	3
1	adjective	4
1	adverb	3
1	simple present	10
1	will future	4
1	past progressive	6
1	connective	1
1	simple past	15
1	present progressive	9
2	question	3
2	relative clause	7
2	adverb	4
2	simple present	3
2	comparative	4
2	modalverb	9
2	will future	3
2	simple past	9
3	present perfect	1
3	relative clause	4
3	past perfect	7
3	passive	8
3	will future	1
3	past progressive	1
3	simple past	9
3	connective	1
4	question	5
4	reflexive pronoun	7
4	reported speech	9
4	modalverb	3
4	indirect question	1
5	question	1
5	adverb	2
5	adjective	5
5	comparative	1
5	connective	1
5	superlative	1
5	indirect question	3
6	gerund	1
6	adjective	3
6	preposition	1
6	comparative	3
6	modalverb	3
6	superlative	2

Table 7.2: Distribution of tasks per grammar topic per theme

The topics covered in the individual chapters are different in their nature. While the tenses in chapter one that are presented in a contrastive manner are already covered in the previous year and constitute a repetition, the topic of gerunds is new in this grade. In chapter two, comparatives, relative clauses, and conditionals type 2 are tested. Since for conditionals it is necessary to know how to use tenses that are repeated in chapter one, the topic of conditionals type 2 in chapter 2 can be seen as a direct continuation. Chapter three introduces both a new tense (past perfect), but also a new voice: passive. Finally, the last chapter that included a pretest and post test is chapter four. In this chapter, reported speech and reflexive pronouns are covered.

chapter	task	language topic
1	1	present perfect vs. simple past
1	2	simple present vs. present progressive
1	3	simple past vs. past progressive
1	6	gerunds
2	7	conditional type 2
2	8	comparison of adverbs
2	9	relative clauses
3	4	past perfect
3	10	passive
4	5	reported speech
4	11	reflexive pronouns

Table 7.3: Language topics covered in the pre-/post tests

These topics are defined in the Bildungsplan (state curriculum, Kultusministerium (2016)). This document defines what contents and language topics have to be covered in specific types of schools (in this case Gymnasium) and in specific subjects (in this case English). The reason why these topics are covered in the pre-/post tests is that the respective chapters in the existing workbook contain exercises that allow to practice these grammar topics or language points.

Test Dates Table 7.4 lists the dates of the pretests and post tests for each school class. In addition to the four themes, the table lists the date of the delayed post test. This post test included a repetition of the first two subtasks of each pretest task. For each of the tests, a member of the project team was present in order to give instructions to students and monitor the process.

The teacher *H907* skipped chapter three of the workbook in FeedBook. For this reason, there is no pretest and post test for chapter three. When the post test for theme two was conducted, the pretest of theme four was conducted.

For the teachers *FQ8D*, *P7IB*, *M62N* and *FZJT* the time between the post test of theme four and the delayed post test would have been too small to have a real delayed test design. Therefore, no dedicated post test of theme four was conducted, but instead only the delayed post test covering the post tests of all themes.

teacher/ class	pre 1	post 1/ pre 2	post 2/ pre 3	post 3/ pre 4	post 4 post 4	delayed post
940B	18-09-25	18-11-26	19-01-21	19-02-18	19-05-03	19-07-10
1NVY	18-09-25	18-11-30	19-01-29	19-03-26	19-05-21	19-07-16
FQ8D	18-09-25	18-11-09	19-01-15	19-02-18	N-A	19-07-09
LTTS	18-09-25	18-11-27	19-01-29	19-03-26	19-05-28	19-07-09
M4SW	18-09-25	18-11-23	19-01-22	19-03-19	19-05-28	19-07-12
P7IB	18-10-10	18-11-26	19-02-01	19-03-13	N-A	19-07-12
YWL7	18-09-20	18-11-22	19-01-24	19-03-18	19-04-29	19-07-05
M62N	18-09-26	18-11-21	19-01-23	19-03-13	N-A	19-07-03
ZHOZ	18-10-08	18-11-19	19-02-11	19-03-18	19-05-28	19-07-05
H907	18-10-08	18-12-10	19-02-15	N-A	19-04-12	19-07-12
FZJT	18-10-01	18-12-10	19-02-14	19-04-11	N-A	19-07-09

Table 7.4: Dates of the pre- and post tests per school class. Classes with incomplete data are marked in gray.

Format of Tests Each pretest had the same format as the corresponding post test. The first part of each pretest consisted of a true/false task. Learners had to either perform grammaticality judgment tasks by ticking the correct options, or select sentences which contained a specific structure. Figure 7.1 shows an example of such a task. In this example, learners have to select all sentences in which the relative pronoun was used correctly.

The second part of each test consisted of a fill-in-the-blanks exercise with pre-defined options. Learners had to select one of the provided options. Initially, this was implemented as a fill-in-the-blanks task with options provided in brackets. Later, this was changed to a fill-in-the-blanks task with multiple choice drop down menus.

The third part of pretests and post tests consisted of prompts with less constrained input. In contrast to the second part, the options are not given here and students can type answers freely. This led to a significant amount of variation in the answer and allowed for task avoidance behavior.

Control Variables

OSPAN In order to test working memory, we used a variant of the OSPAN test with Klingon characters (Hicks et al., 2016; Ruiz Hernández, 2018). Each trial consisted of a simple math exercise (checking if they were paying attention) and a sequence of Klingon characters. In the following screen, learners had to select the Klingon characters in the order in which they recalled them. This complex span task involves performing two tasks jointly. The final OSPAN score takes into account the math accuracy as well as the correctness of parts of the recalled symbols in the sequence of symbols previously displayed.

MLATV We used a part of the Modern Language Aptitude Test (MLATV) test described in Ruiz Hernández (2018) to test verbal declarative memory.

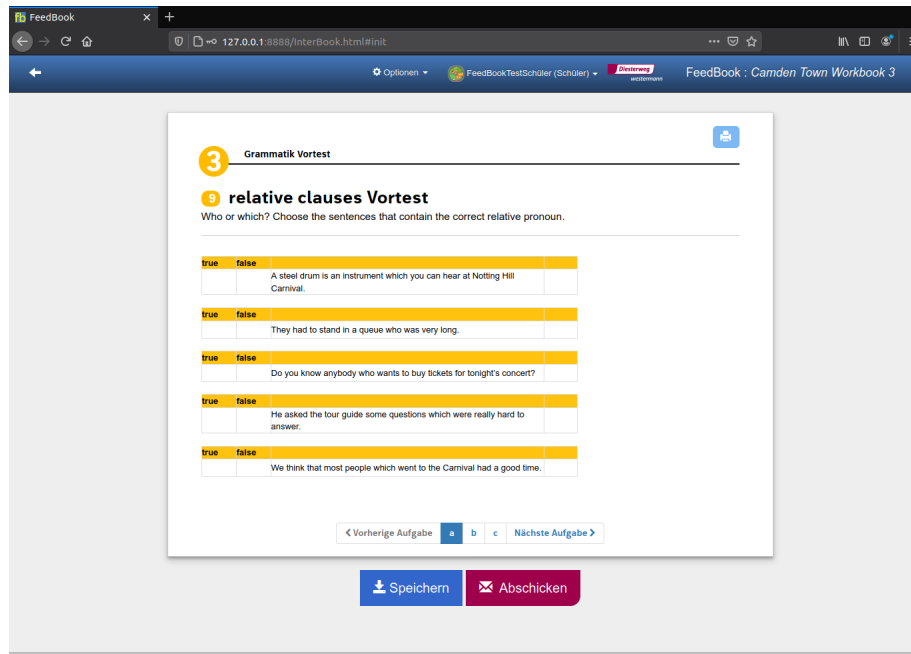


Figure 7.1: Pretest task part a: selection task implemented as true/false exercise

Learners were given a test in which they were supposed to learn translations of English words into a fantasy language. After a test trial as part of the test explaining to them how the test works, they essentially did a vocabulary test and the test score indicated how many words they recalled correctly.

Typing Tests In conjunction with the tests for theme 4, learners were asked to do a typing test. The test had three attempts, with each attempt requiring them to type words shown in a box above the input field. The test computed an accuracy for each of the trials, taking into account the speed of typing, but also the accuracy (typing errors). The test is an adapted version of an existing typing test ¹.

C-Test

At the beginning of the school year, and as part of the delayed post test, learners were asked to complete a c-test. C-tests can be used as a measure of general language proficiency as well as proficiency in sub domains (e.g. Eckes and Grotjahn (2006)). Figure 7.4 shows an example of one page of the c-test in the FeedBook system. While the first page explains the task, the remaining five subtasks consist of tasks of this form.

¹<https://github.com/anschwa/typing-test>, last accessed 2020-08-14

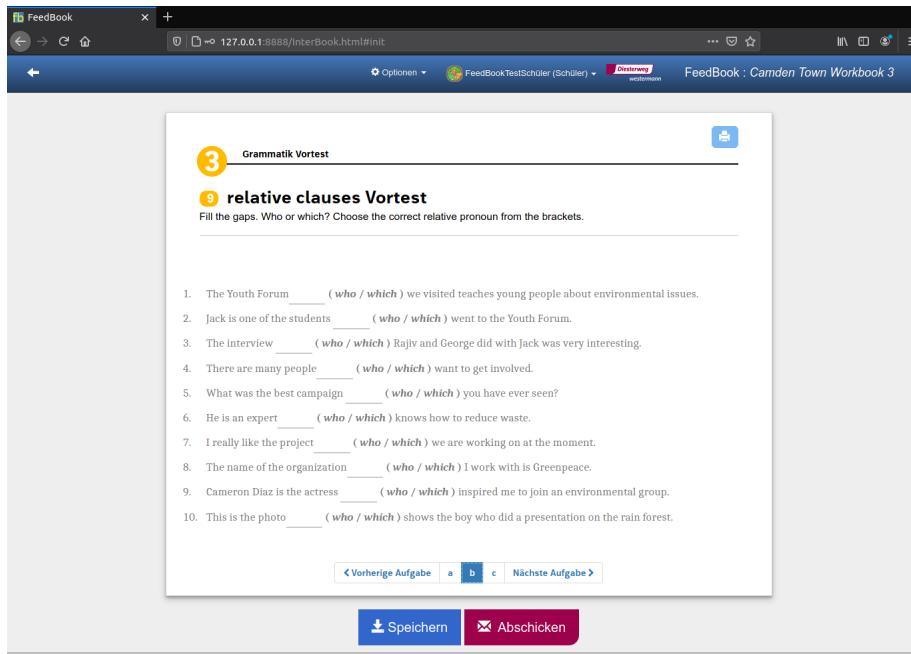


Figure 7.2: Pretest task part b: fill-in-the-blanks exercise with provided options

Personal Background and System Usage Questionnaire

Student questionnaire At the end of theme 2, learners completed a questionnaire asking about personal information. The questionnaire consisted of eight parts, implemented as eight tasks in FeedBook. Table 7.5 lists the different parts of the questionnaire, the topic, and examples for each of the questions in each part. Figure 7.5 shows an example of how the questionnaire is rendered in the system. Parts four and five of the test contains items from a previously established student questionnaire described in Gaspard et al. (2017).

Teacher questionnaire In addition to the student questionnaire, we sent a short questionnaire to teachers focusing on user satisfaction. It comprised 25 items and was sent out on paper to teachers when they were asked to send a list of the final English grades of their students assigned to the user pseudonyms (not real or user names). The test contains both multiple-choice single selection items and free text input items. The first six questions focused on the usability of the system, with question seven allowing to provide comments about how it could be improved further. The next questions asked about how and how long teachers used the system and to what extent they were satisfied with the tasks and task types implemented in the system. The questions explicitly asked how much time they invested into the system and how much time they could use for other activities. Further questions targeted technical problems and a

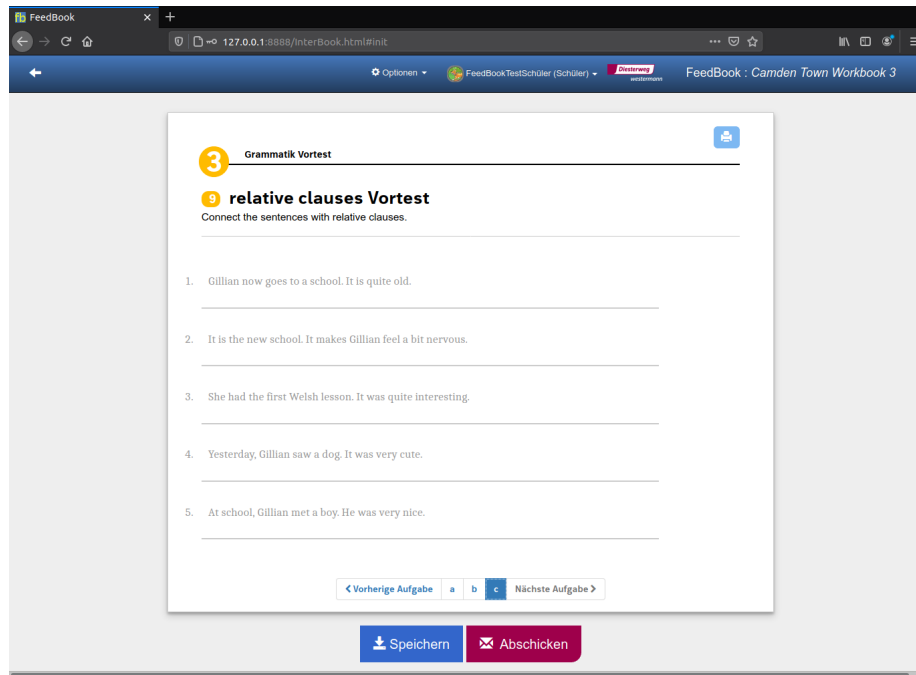


Figure 7.3: Pretest task part c: short-answer exercise with free input

part	topic	examples of questions
1	personal details	birthday, previous grade, language background
2	family background	languages spoken at home, origin of parents
3	English usage	parent support for English in school, sources of English input
4	skill self-assessment	estimated skills and attitude towards English
5	attitude	English in schools, English teacher
6	homework	effort, time, frequency, difficulty, attitude
7	FeedBook homework	same as homework, plus impression of system
8	internet usage	timing, devices, attitude towards computers

Table 7.5: Overview about student questionnaire

comparison of FeedBook with other digital learning systems. Finally, teachers were asked whether they would use the system again for teaching English.

Writing Tasks

At the end of the school year, students were given a free writing task that implicitly required them to use the constructs learned during the school year.

Sprachtest

Lies den Text erst einmal mit den Lücken durch, damit du ungefähr weißt, worum es geht.
Dann erst schließe die Lücken. Bei einigen wirst du ein bisschen knobeln müssen.
Lies dir den Text am Ende noch einmal durch!

My First Rabbit

My very first wonder of the world was a big, soft, Angora rabbit. Her na ___ was Sarah. I lo ___
this rabbit more th ___ I loved m ___ baby sister. T ___ rabbit never ope ___ her
mouth a ___ screamed and to ___ all the atten ___ away from m ___. The rabbit
ne ___ woke everybody i ___ the mid ___ of the ni ___. No, Sarah ju ___ sat there
un ___ a tree in our gar ___ and waited patie ___ for me t ___ come and gi ___
her more car ___ and more bread ea ___ day. And any ___, to have a rabbit w ___
something special. Nob ___ else at school had a rabbit, but baby sisters were two a penny and
everyone had them.

< Vorherige Aufgabe a b c d e f Nächste Aufgabe >

Speichern Abschicken

Figure 7.4: C-Test exercise

1 Fragebogen

7 Hausaufgaben mit dem FeedBook
Wie verhältst du dich bei der Erledigung deiner Englischhausaufgaben im FeedBook?

Mein Verhalten bei Englischhausaufgaben im FeedBook:	stimmt gar nicht	stimmt eher nicht	stimmt eher	stimmt genau
Ich tue mein Bestes bei den Hausaufgaben im FeedBook.				
Ich bearbeite in letzter Zeit die Englischhausaufgaben im FeedBook so gut ich kann.				
Ich versuche immer meine Englischhausaufgaben im FeedBook vollständig zu erledigen.				
Bei den Aufgaben im FeedBook ist mir oft nicht klar, was ich machen soll.				
Die Aufgaben im FeedBook sind klar formuliert.				

< Vorherige Aufgabe a b c d e Nächste Aufgabe >

Speichern Abschicken

Figure 7.5: Example screen of the student questionnaire

Students were asked to compare two of their last holiday trips (requiring tenses and comparatives), describe their next holiday trips (requiring the use of future tense), and imagining what they would do with a lot of money during their next holiday trip (requiring the use of conditionals). This led to a substantial amount of learner responses, the analysis of which lies beyond the scope of this thesis.

7.1.3 Procedure

The FeedBook study was conducted over the course of an entire school year from September 2018 to July 2019. The central research question of this study was whether immediate formative meta-linguistic scaffolding feedback by a tutoring system was beneficial for learning, and if yes, for which linguistic constructs and outcomes.

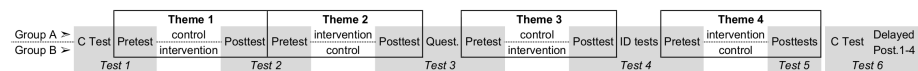


Figure 7.6: Experimental design with rotating groups (taken from (Meurers et al., 2019a, p. 13) and slightly adapted)

At the beginning of each chapter, before the tasks were unlocked in the system, learners took part in a pretest that tested prior knowledge about the linguistic constructs introduced in this chapter (cf. section 7.1.2). The same test was administered at the end of the chapter when the teacher decided to move to the next chapter. In conjunction with the post test, the pretest for the next chapters was conducted. This procedure was repeated for themes 1 to 4. Overall, the study tested whether for different linguistic constructs, introduced in four chapters across a school year, scaffolding feedback by the FeedBook system was beneficial for explicit learning of these grammar constructs.

The learners were divided into an experimental and a control group, with the assignment rotating with each chapter. For each school class, the system randomly assigned each student to test group A or B, but ensuring that the number of students in each experimental group was balanced for each school class (within-class randomization). This balance of participants with respect to experimental groups per school class was conducted in order to account for effects of between-teacher effects.

Figure 7.6 illustrates the rotating design of the randomized controlled field trial study. The intervention group of theme 1 became the control group for them 2 and vice versa. Furthermore, the graphics shows supplementary materials collected, such as individual difference tests, which are described in section 7.1.2. The difference between the control and the intervention group was that for the control group, feedback templates that target the constructs introduced in the respective chapter, were blocked (i.e. computed, but not displayed). The rest of the feedback was still accessible to the control group. Only for tasks in

which the only constructs covered were exclusively from the set of new grammar topics introduced in this chapter, the feedback was deactivated altogether for the control group. For items or tasks where the system did not provide feedback to students in the control group, teachers could still provide feedback. This was, however, only possible after learners submitted an exercise to the teacher, similar to a submission on paper that a teacher collects, grades, and hands back. The feedback for the control-group for language topics on their blacklist was thus delayed (versus immediate) and manually provided (versus automatic).

This worked by implementing a blacklist consisting of language constructs. For each feedback template, we know what target language construct is associated with it. Whenever a feedback message is generated, the system can check whether this targets one of the constructs on the blacklist for this experimental group, and if yes, block it from the end user. The blacklist is thus construct-specific, and not chapter-specific.

Students and teacher were free to use the system in their own way. The only requirement was that teachers were asked to assign a set of core tasks in each chapter to their students, with core tasks focused on the new grammar topics. From informal conversations with the teachers, we learned that the system was used in a range of different ways. While the system was mostly used for homework assignments, some teachers also used it (partly) in the classroom. In terms of devices, the school classes differed. While some classes were using tablet computers on a regular basis, others used stationary computers. The input devices and conditions thus differed between the individual school classes and schools.

Ecological Validity The study was conducted under ecologically valid conditions. Schmuckler (2001) defines ecological validity as a combination of three criteria: setting of an experiment, stimuli used, and type of responses elicited. In terms of experimental setting, the object under investigation should be observed in an authentic, naturalistic context, and not detached from real-life or in artificial contexts. This also includes an authentic sample of participants. As described above, in the FeedBook study we analyzed the data from a horizontal slice of schools and participants in actual school contexts. The second criterion for ecological validity pertains to the stimuli. According to Schmuckler (2001), research should be conducted on stimuli in the experiment that occur in the same way in real life. Since we used exercises from an existing, state-approved workbook already in use, this criterion is fulfilled. As the final criterion, it is necessary that what is measured in the experiment as the responses to tasks is what is also produced by the participant's. Given that both in the exercises in FeedBook, in the pretest and post test, and in the exams in school, the same type of responses were collected and evaluated, the nature of the response and the measured responses are similar in their nature, fulfilling also this criterion.

7.2 Collected Data

7.2.1 System Log

FeedBook logs the actions of users in two ways. On the one hand, the system contains a general system interaction log (section 7.2.1). It provides general system usage information that allow to quantify the usage. For example, it stores when users logged in, where they navigated inside the system, and when they logged out. This is a relevant source of information in distance learning, since it allows to see whether learners had problems with the system (e.g. when the system crashed). Complementing this, the system also contains an exercise-specific interaction log (section 7.2.1) providing information about the interaction of learners with exercises and especially with the feedback mechanism. In contrast to the general interaction log, the exercise-specific log specifically holds language information. What differentiates the two logs is thus their contents: general interaction and interaction with language. In the following, we will describe both types of logs. We conclude this section with a description of how to filter the data by interaction dates and the output of these processes in section 7.2.1.

General System Server Log

The general system server log contains all the information logged by the web application container in which the system was running. It contains information that stems from the container, such as instances in which the application we undeployed or deployed, but also information that is specific to the FeedBook system. The log data in this category can be subdivided into pre-envisaged statements and incidental log entries. The latter consists of exceptions logged by the system, such as run time exceptions when unexpected input is encountered or resources are unavailable. The former consists of statements that are intentionally generated when certain actions are performed in the system. While initially this consisted of statements that are called when users navigate between different interfaces (e.g. navigating between chapters and exercises), it was extended over the course of the study to include more fine-grained distinctions such as learners actively requesting or clicking on feedback messages or the time-on-task between opening and leaving an exercise.

In sum, it contains general usage information only indirectly related to language such as navigation patterns, login and logout information, and error messages. The system log can answer questions like who used the system when, what are typical navigation patterns in the system, or did learners attempt to do their homework.

Figure 7.7 shows an excerpt of the system log. The format of the log is one line with a time stamp and the logging tool, and then one line with the actual log message. The two lines together form one logging entry. In the example provided, the first entry indicates that a learner logs into the system. As the next step, this user sees the start page (mosaic lobby). meanwhile, another

```

Apr 29, 2019 10:03:33 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server
INFO: [redacted] logs into the system
Apr 29, 2019 10:03:33 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server
INFO: [redacted] opens the mosaic lobby (top level)
Apr 29, 2019 10:03:36 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server
INFO: user [redacted] spent 4 seconds on the exercise Theme 14 , SubTask 2 SubTask 0, new total time on this task: 4
Apr 29, 2019 10:03:36 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server
INFO: user [redacted] spent 0 seconds WORKING on the exercise Theme 14 , SubTask 5 SubTask 0, new total time on this task: 0
Apr 29, 2019 10:03:36 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server
INFO: Subtask id: 10093574
Apr 29, 2019 10:03:37 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server
INFO: [redacted] opens the mosaic lobby (top level)
Currently online users: [redacted]
Apr 29, 2019 10:03:37 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server
INFO: [redacted] registers user?
Apr 29, 2019 10:03:37 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server
INFO: [redacted] opens the mosaic lobby (top level)
Apr 29, 2019 10:03:37 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server
INFO: [redacted] logs into the system
Apr 29, 2019 10:03:42 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server
INFO: [redacted] opens the message fullscreen view
Currently online users: [redacted]
Apr 29, 2019 10:03:42 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server
INFO: [redacted] registers user?
Apr 29, 2019 10:03:44 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server
INFO: [redacted] opens the mosaic lobby (top level)
Apr 29, 2019 10:03:45 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server
INFO: [redacted] opens the practice interface with task Theme 14 , SubTask 3
Apr 29, 2019 10:03:46 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server
INFO: Subtask id: 10093572
Apr 29, 2019 10:03:46 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil log$Server

```

Figure 7.7: General system server log (user names blinded)

user exited an exercise after having looked at it for 4 seconds and worked on it for 0 seconds (log entries 3 and 4). This user (blinded for anonymization) then goes back to the start page. The system then lists all the currently online users. Subsequently, more users log in, and one of them opens the interface in which all messages are displayed. Later, even more users log in and work on different exercises in the system. While it constitutes only a small excerpt from the log, it illustrates the format and type of information present in the log. For any analyses, we replaced the user names with pseudonyms before analyzing the data.

Exercise Interaction Log

The system generates an interaction log that stores the actions of learners performed when working on exercises. The content of this log consists of learner answers, diagnoses of misconceptions, system feedback, and time stamps. This is complementary to the general system log in that it contains language data.

The subject of the general system log is a domain-independent system usage and could be extracted from or implemented in every type of web application. In contrast, the subject of the exercise log are domain-specific learner interactions on exercises designed to practice language. For this reason, the exercise log contains language data.

For every answer a learner submits, either by hitting the enter key, hitting tab to change the focus on another input field, or clicking on the feedback request question mark symbol, the system generates a *LoggingToken* and adds it to the interaction log of this student for this exercise. This is also the case for instances in which the feedback was blocked via the blacklist.

Each *LoggingToken* contains the following fields: learner answer (what the learner typed), taskfield (reference to which item the answer was provided to), feedback string (feedback message text plus degree of specificity), feedback template (internal code for error type, or null if answer is correct), time index (relative position in log), and time stamp (server time when the feedback was logged). Additionally, the *LoggingToken* contains a Boolean variable "auto-

matic“, which is set to true if the LoggingToken was triggered by an automatic check (when entering the exercise, clicking the earlier existent check-all button, or before submitting to the teacher). In essence, it expresses whether a learner answer was checked in isolation from this specific item or on conjunction with other items. Later, we added an explicit indication of whether the feedback was blocked via the blacklist.

```

Apr 29, 2019 6:33:53 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil logOnServer
INFO: user is allowed to see the feedback FeedbackInfo [diagnosis=Diagnosis [span=Span [beginIndex=14, endIndex=15], construct=ConstructValue [construct=contact clause, value=], correctConstruct=ConstructValue [construct=relative pronoun who, value=who], additionalConditions=[], templateString=REL_OBJ_ALL_01], feedbackStrings=[FeedbackString [message=Please read the instruction again and use a relative pronoun., specificity=1]]]
Apr 29, 2019 6:33:54 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil logOnServer
INFO: user got the following LoggingToken LoggingToken [answerText=Caroline asked Leon if he like english girls., taskField=TaskField [promptText=1. Do you like English girls?, inputType=SENTENCE, targetAnswer=Caroline asked Leon if he liked English girls. | Caroline asked Leon whether he liked English girls., sortIndex=1, example=Caroline asked Leon, questionType=null, requiredNumItems=null], feedbackStrings=[FeedbackString [message=<a class="lflink" href="/media/lif/WebContent/LIF1R.html">Simple past</a> with regular verbs: verb + -ed., specificity=1]], timeIndex=8, template=VP_SPA_LE_M_01, date=Mon Apr 29 06:32:31 GMT+200 2019, automatic=true, blacklisted=true, languageTool=false]
Apr 29, 2019 6:33:54 AM com.google.gwt.logging.server.RemoteLoggingServiceUtil logOnServer
INFO: user got the following LoggingToken LoggingToken [answerText=After that she asked, taskField=TaskField [promptText=3. Who's your favourite British musician?, inputType=SENTENCE, targetAnswer=After that she asked him who his favourite British musician was., sortIndex=3, example=After that she asked, questionType=null, requiredNumItems=null], feedbackStrings=[FeedbackString [message=Please read the instruction again and use a relative pronoun., specificity=1]], timeIndex=9, template=REL_OBJ_ALL_01, date=Mon Apr 29 06:32:31 GMT+200 2019, automatic=true, blacklisted=false, languageTool=false]

```

Figure 7.8: Exercise-specific interaction log (user names blinded)

Figure 7.8 shows an excerpt of the exercise log. In this example, the learner typed “Caroline asked Leon if he like english girls”. The target answer is “Caroline asked Leon if he liked English girls”. The feedback, which is the eighth (time index 8) in this interaction, was based on the feedback code *VP_SPA_LE_M_01* and read “Simple past with regular verbs: verb + -ed.” It’s specificity is 1 (in contrast to 0 for default feedback) and the feedback was generated automatically (e.g. when the learner opened this exercise). However, due to the student being in the experimental group for which feedback on tense formation was blocked, this specific feedback was blacklisted and not displayed. The next feedback (time index 9) was shown for the input “After that she asked” for the target answer “After that she asked him who his favourite British musician was.” This LoggingToken also shows that the example answer of this item is “After that she asked”. This means that in this case, the automatic feedback was triggered on an item in which only the already provided example answer was present, i.e. an item which the learner has not worked on so far. This message, which relates to relative pronouns, is not blocked by the blacklist and accessible to the learner. This is also expressed in the first message of this log at the top of the figure.

While the data of LoggingTokens is straightforward, the interpretation is not trivial. Difficulties arise when looking at the granularity of the data structures. An open question is whether to count individual instances of logging tokens or modeling sequences of them. While the data structure expresses interaction with feedback, this does not happen in isolation, but as part of a bigger process of working on this task. Then, while the LoggingToken refers to an individual item, the task context constitutes of more than only the item. In order to interpret the LoggingTokens in the context in which they were generated, they need to be interpreted with respect to explicit task models. Another question is whether to weight information differently over time. Related to this is the question of how

to handle periods of non-activity (i.e. the time between interactions) or pauses (when learners left a task and opened it again later to continue working). Then there are differences between school classes and chapters, with consequences on hidden vs. displayed feedback (cf. section 7.1.3 for a detailed discussion of the blacklist filtering and section 8.2 for the underlying data model).

Filtering the Interaction Logs by Test Dates

As Table 7.4 shows, the pretest and post test dates varied depending on which school class learners belonged to. If analyses have the goal of linking interaction data from one theme to the pretest and post test results, then it is necessary to filter the interaction data to only include the data between the test dates.

For the general system log, due to the variations between the test dates, the log can not simply be cut into slices where the split points are the test date. Instead, we implemented a script that splits the log by linking each participant to a test date and filtering out all the interactions of this user that took place before the pretest and after the post test. The result is a log with the same format as before, but filtered by test dates individually for each student.

Table 7.6 lists the number of lines in the separated system log files. The number of log statements is relatively stable across the themes, with theme 3 having the smallest number of log entries.

theme	log file line count
1	303,119
2	286,902
3	280,694
4	303,820
total	1,174,535

Table 7.6: Number of lines in logs filtered per theme

theme	number of LoggingTokens
1	140,893
2	75,422
3	49,648
4	35,827
total	301,790

Table 7.7: Number of LoggingTokens (recorded interactions) per theme

Table 7.7 shows the number of LoggingTokens by theme. The number of LoggingTokens decreases from theme 1 to theme 4. An important factor that partly explains this is the fact that the check-all button was disabled after having been initially active. This button had triggered a check for all the items in a given exercise, inflating the number of LoggingTokens. However, the fact

that the number of LoggingTokens still decreases for later chapters still shows a reduction in exercise activity later in the school year.

7.2.2 Submissions

Table 7.8 lists the number of tasks submitted to teachers by learners for each chapter and section. The table shows that for the optional chapters 5 and 6, which were not part of the study, the number of submissions is lower than for other chapters. Also for chapter, 4, which was covered towards the end of the school year, there are less submissions. In total, there are 8,608 submissions for the exercises in theme 1 to 6 and 8,432 for themes 1 to 4.

theme	section	number of submitted tasks
0		72
1	A	249
1	B	847
1	C	468
1	Check Your Progress	331
1	Extra Training	425
1	Additional Practice	403
2	A	306
2	B	628
2	C	113
2	Check Your Progress	224
2	Extra Training	454
2	Additional Practice	701
3	A	117
3	B	403
3	C	326
3	Check Your Progress	353
3	Extra Training	377
3	Additional Practice	342
4	A	245
4	B	232
4	C	162
4	Check Your Progress	238
4	Extra Training	253
4	Additional Practice	163
5	A	46
5	B	73
5	C	1
5	Check Your Progress	17
5	Extra Training	15
6	A	10
6	B	12
6	C	2
6	Check Your Progress	0
6	Extra Training	0
total		8,608

Table 7.8: Number of tasks submitted to teachers for each chapter and section

Part III

**Learning Analytics in
Practice**

Chapter 8

Modeling Individual Learners

In this chapter, we present Learning Analytics functions for individual learners. This perspective is complemented by Learning Analytics tools for groups of learners in the form of school classes (chapter 9) and by Learning Analytics functions that take into account the entire learner population (chapter 10).

We start off by analyzing the needs of students in an educational context in section 8.1. Learners need to know, among other factors, how well they are performing for a specific language construct, how their competence developed over time, and what information a system has about them. In order to provide such an open learner model, a tutoring system needs to be able to diagnose construct-specific competence and errors related to a specific target construct. We use section 8.2 to explain how this competence mapping was conducted in FeedBook.

In section 8.3, we turn to the actual open learner model implemented in FeedBook and discuss the individual components of it. We start by showing the top level of the learner model in section 8.3.1, which provides learners with a quick overview about their competence in the areas of a specific language topic. In section 8.3.2, we show a visualization of errors per linguistic target construct before we turn to a visualization of the longitudinal development of competence and uptake of learners in section 8.3.3. We conclude the chapter with a discussion of an adaptive proficiency-dependent and item-specific exercise sequencing algorithm in section 8.3.4.

Disclaimer: Specific sub parts of the contents of this chapter have been mentioned or referenced in Rudzewitz et al. (2020) and are cited accordingly in the running text of the respective sessions.

8.1 Needs Analysis: Students

Students in a school context have different needs with respect to Learning Analytics in the educational context (Rudzewitz et al., 2020). Since students constitute by far the largest user group in a tutoring system, it is important that their needs are satisfied. In the following, we will outline the most important needs with regard to Learning Analytics for students.

First of all, learners need to know what they have already learned. This includes information about what has been practiced and how much practice has been put into a specific topic.

The next need emerges from the first need. Learners not only need to know what and how much they have learned, but also how well they performed. In essence, this amounts to a visualization of the degree of mastery of a concept. With mastery offering a positive perspective on the competence data, the other side is equally relevant. Learners need to know which misconceptions have been diagnosed and which errors or mistakes they have made.

Moving from functions that aggregate past data to estimate and visualize the current state, students have needs relating to the future. Students need guidance and suggestions for the next steps, especially which exercises to practice on next. The selection of exercises should take into account the degree of mastery and misconceptions. An adaptive sequencing tailored towards the state of individual learners needs to serve the goal to overcome the misconceptions, as well as offer an individualized access to new material.

Finally, from a general point of view, learners (like other user groups), need to know what the system knows about them. Especially in the light of recent data protection regulations, and given how rich learning interaction data is, it is essential to provide learners with access to *their* data.

Without Learning Analytics, students are confronted with a black box that collects but not outputs data as well as a lack of adaptivity and transparency. Given the needs and state of research it is irresponsible to not implement Learning Analytics functionalities in a tutoring system.

8.2 Diagnosing Construct-Specific Positive and Negative Evidence for Competence

In order to diagnose both competence and a lack thereof for the open learner model, we implemented a bidirectional competence mapping. The mapping links individual items with multiple linguistic target constructs on the one hand with feedback templates associated with one linguistic target construct on the other hand. Especially for items with longer input fields, often multiple constructs from different areas need to be produced by the learner. For example, in a sentence like “Paul is usually driving faster than Dom”, in an imaginative exercise with the pedagogical goal of comparatives, learners need to produce a correct form of the present progressive, produce correct agreement between the subject and predicate, and obey the correct word order.

This leads to the challenge of how to interpret learner actions that do not result in a correct answer with respect to the competence modeling. While an exercise is typically designed to practice a certain target language construct, learners need to actively produce constructs that are not sub constructs of the pedagogical set of target constructs, but rather side constructs. However, independent of whether a construct is a pedagogical target construct or not, the learner model needs to diagnose whether a learner produced it correctly or not. For this reason, it is not sufficient to associate an item with only the pedagogical target construct, but instead it is necessary to model all language constructions that make up a specific item.

When a learner produces an answer that is identical to a target answer, the learner model update is trivial. In this case, all target constructs associated with an item can be updated with a positive increment. The interesting case is when an answer is (partly) (in)correct. In this scenario, the system needs to diagnose which of the set of target constructs associated with the item are affected by the learner error(s) and which ones not.

The computation of this delta between the learner answer's constructs and the item's target constructs is enabled by associating each feedback template with an explicit target construct where applicable. In addition to modeling an error with the diagnosed construct and error code (see Figure 6.2 for an example), the feedback template also expresses what the linguistic target construct is. This allows to see which of the constructs tested in the item are affected by the error in the learner answer.

Concretely, we build an index of target constructs per task field once during the feedback generation process for form feedback (section 6.2). Since rules add one diagnosis associated with a feedback template, and since the feedback template is associated with a positive target construct, we store the target constructs of all feedback templates along with their side conditions added by all applicable rules in the feedback pre-generation process. They are stored in a special data structure *TargetConstructsByTaskField* in the data base for an efficient lookup at run time. This process only conducted for specific form feedback, i.e. not meaning or spelling or default feedback, since in the latter cases, no explicit target constructs are available that could be compared to an item's target constructs.

We use this mechanism to update the learner model in real time for every interaction with the feedback mechanism, i.e. every time a learner sends an answer to the server for checking its correctness. While we record simple error template frequencies by learners with a time stamp, the described bi-directional competence mapping is used to compute fine-grained diagnoses. Table 8.1 lists all the categories used in the competence modeling in FeedBook. Every time one of the categories is applicable, it is stored in the data base along with the target construct and side conditions, a time stamp, and a reference to the learner. In the following, we will explain them one by one.

uptake/evidence type	description
correct at first try	positive feedback for all target constructs
answer fully correct after feedback	correct evidence for all target constructs after feedback
different error for same construct after feedback	different error, but same target construct
different error for different construct after feedback	feedback to something else
correct but not necessarily at first try	something right, but the system doesn't know if it was correct already before
answer ultimately correct	answer is identical to target answer when the student submits it
answer ultimately false	answer is not identical to target answer when the student submits it

Table 8.1: Uptake types recorded in the learner model in FeedBook

Correct at first try When a learner submits an answer that is evaluated as correct, and the learner has not interacted with the given item before, positive evidence at first try is recorded for all target constructs associated with this task field. We distinguish this category from correct after feedback since it expresses the fact that the learner did not need feedback to arrive at a correct answer.

Answer fully correct after feedback When an answer is first incorrect and a learner receives specific feedback related to one of the target constructs, then manages to implement this feedback and produce a correct answer, the system records for the specific target construct associated with the feedback that the learner addressed this specific construct. Essentially, the system records an instance of helpful feedback for a specific construction.

However, it is possible that the learner answer contained more errors that were not targeted by the feedback, but nevertheless addressed by the learner in the revision process of the answer following the feedback. For this reason, the system additionally records uptake of type “correct, but not necessarily at first try” at the same time for all constructs associated with this item, in contrast to the uptake type “Answer fully correct after feedback” only recorded for the construct targeted in the feedback template.

Different error for same construct after feedback Since many feedback template are associated with a target construct, we can map errors to specific language constructs. This allows to determine whether two feedback templates displayed in succession target the same language construct or different constructs. Since the same construct can be produced in many different, linguistically incorrect ways, it is possible that a learner makes a sequence of errors working on the same form or construct. For example in a tense exercise with the target answer “I got up today”, a learner can write first write “get” and then “gots” instead of “got”. In this example, the same target construct (first person simple past for an irregular verb) is the target of two errors that express a different error (tense confusion vs. agreement), but they both map to the same target construct.

Different error for different construct after feedback This type of reaction to corrective feedback is similar to *Different error for same construct after feedback*, but instead of the same target construct, two subsequent templates refer to different linguistic target constructs in the learner answer.

Correct, but not necessarily at first try Since when feedback occurs it is only clear that the specific target construct referenced in the template is wrong, but not whether other constructs in the learner answer are also wrong, it can only be diagnosed with certainty that the other constructs are right once the learner submits a fully correct answer. Simply taking the difference between the current target construct and all target constructs is not valid since learner language contains ambiguities and we only know that one construct is wrong,

but it does not imply that all other constructs are correct. Therefore, this type of competence modeling is stored in the data base only once a learner submits a fully correct answer.

Answer ultimately correct This type of reaction to feedback is technically not uptake in the strict sense, but modeled internally in the system in the same data structure and thus described here. When a learner answer is submitted to the teacher, the system evaluates whether the learner submitted a correct answer or not. This is not as trivial as comparing a learner answer with a target answer, but requires specific treatment depending on the task type and feedback. Especially for meaning-oriented tasks, the correctness of a learner answer is determined based on a similarity computation that uses an alignment configuration as a basis. In interfaces such as the material designer perspective for task performance (cf. chapter 10), the system needs to quickly fetch the ultimate correctness status of a large number of learner answers. An implementation that re-evaluates and re-analyzes all learner answers would be highly inefficient and unusable in an authentic context with big amounts of data. For this reason, the system offers a shortcut by storing once whether an answer has been evaluated as correct.

Answer ultimately false This feature is similar to *Answer ultimately correct*, but a specific learner answer, at submission time, was not identical to one of the target answers and has not been evaluated as correct during the interaction process.

8.3 Open Learner Model

The FeedBook system contains a learner model implemented to address the need of learners outlined in section 8.1. The learner model is accessible from the start page of the system (see Figure 5.14 bottom left) to all learners. The learner model is open to end users in that it presents all information to individual learners that the system has collected about them. As discussed in section 4.1.2 in detail, there are many reasons for opening learner models to learners.

In order to prevent that learners are cognitively overloaded by too much information, the learner model is organized in a hierarchical structure. This allows to display information on demand when learners request it. The hierarchy consists of three levels: top level grammar topics, performance data for grammar points under a top level category, and detailed information for an individual grammar point. The grammar points represent all linguistic constructions listed in the state curriculum for English 7th grade. Purely linguistic in nature, these grammar points do not contain pedagogical information. They thus do not directly mirror the curriculum, but offer a model of it from a linguistic perspective closely tied in with the feedback generation algorithm.

Figure 8.1 shows the learner model at the top level as it is displayed when learners first enter it. It displays ten categories covering all constructs from

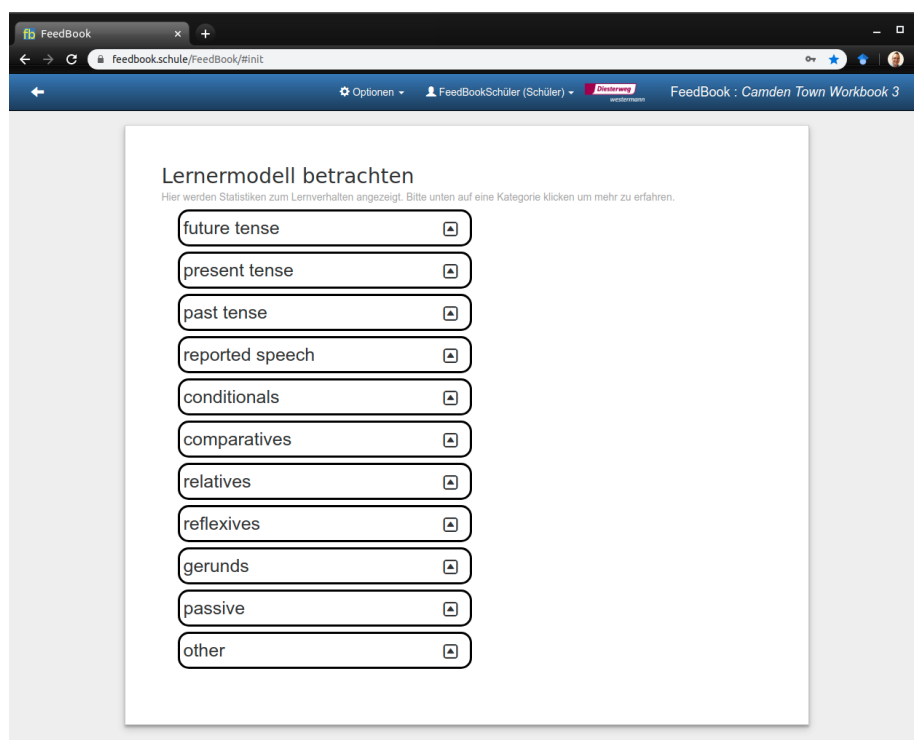


Figure 8.1: Learner model top level

the state curriculum. Additionally it contains one category “other” for further constructs that do not fall under the other top-level categories. This category was, however, not used in the system, i.e. does not contain sub categories.

Upon clicking on one of the top level categories, the learner model displays detailed information. The respective Learning Analytics interfaces are discussed in the following. In section 8.3.1, we describe the second level of the learner model offering a spider web chart that presents aggregated information about learners allowing learners to see at one glance where strengths, weaknesses, and gaps for a top-level category are. We then proceed with information on the third level. Section 8.3.2 illustrates a visualization of error frequencies by target constructs. We continue with a discussion of the longitudinal development of competence for the specified construct (section 8.3.3). We conclude by explaining how the open learner model implements an adaptive, item-specific and proficiency-dependent sequencing of material (section 8.3.4).

8.3.1 Language Area Competence Visualization

Figure 8.2 shows the second level of the learner model after clicking on one of the top-level categories (in this case *present tense*). The first type of information the system presents to the learner when opening a top-level category is a wind rose chart offering the learner an overview over his or her performance on this topic at one glance. The wind rose chart offers an easy-to-interpret summary of the learner performance for this grammar topic by indicating for each sub topic (each corner in the diagram) with green color instances of successful usage of this construct and in red instances of errors observed for this learner and this target construct. The size of the red or green part differs depending on the number of observations and on the proportion of correct vs. incorrect usage. The chart does not only allow to compare correct vs. incorrect usage, but also gaps in practice or underuse of a construct. If there is no or very little performance data available (like simple present with regular verbs in Figure 8.2), the absence of red and green bars in contrast to the other existing data shows to the learner that he or she did not do exercises for which the respective construct needed to be produced. This allows to identify areas in which, without an open learner model, learners could get into a situation in which they do not know that they do not know something because they did not encounter this phenomenon before.

Technically, the system adds the instances of *answer fully correct after feedback*, *correct at first try*, *answer ultimately correct* and *correct but not necessarily at first try* to generate the green data points, and counts the number of misconceptions related to this grammar topic to generate the red data display. This approach was chosen since misconceptions (error frequencies) are clearly assigned to an error, whereas (partly) successful uptake can be related to multiple problems and correct forms. Therefore, the system needs to account for multiple cases here.

Every corner of the spiderweb chart is associated with one of the language topics assigned to the top level topic. In the example in Figure 8.2, this is the *simple present* and the *present progressive*, along with several special cases

where these tenses occur in regular or irregular verb forms or signal words. For each of the grammar sub topics, the spiderweb contains an edge, but in addition, the system renders a graphical element below the diagram. It is rendered with a background color and information text, the label of the grammar topic, and a star rating.

The background color is directly related to the spiderweb diagram and renders a label in red, yellow or green. Red is selected if a learner made more errors than correct instances for a grammar topic. Yellow is chosen when the positive and negative evidence is equally sized. Green is selected if the positive evidence outweighs the negative evidence. For example in Figure 8.2, the grammar topic *present progressive* has a red label, and the spiderweb chart shows for this topic that the red bar is higher. Compare this to *present progressive with present progressive signal*, where the green bar outweighs the red bar and the graphical element related to the language topic is rendered in green.

The star rating indicates how often this learner was able to show successful uptake for errors related to this language construct. Similar to the computation of the red, green and yellow color assignment, the system computes a ratio of *answer fully correct after feedback* plus *answer ultimately correct after different error for same construct after feedback* plus *answer ultimately false*. It thus puts instances of successful uptake for this construct in relation to unsuccessful uptake, both locally after one feedback message and globally when submitting an answer.

The learner model contains an additional feature: if a construct has not been practiced for a specific amount of time (in the system: 30 days), the system adds a gray filter on the construct box. The learner model thus makes use of temporal information in this specific case. A potential drawback of the learner model is related to his point: the spiderweb chart does not only take into account recent data, but all data observed since the beginning. This implies that with continued usage and an improvement of a learner with respect to a language construct, the red parts get smaller with more positive evidence, but never completely disappear. Since every data point used as input to the diagram is associated with a time stamp, it would be possible to only take data into account for a certain time window.

Upon clicking on one of the boxes, the system allows learners to look at more fine-grained information like error frequencies (section 8.3.2), the temporal development of performance measures over time (section 8.3.3), and an adaptive selection of exercises (section 8.3.4).

8.3.2 Target Construct Specific Error Frequencies

When learners request more information about a specific language topic, they can visualize information about the most common errors they made related to the given language topic. For every type of error where the target construct of the feedback template was the given language construct, the system adds a bar in a bar chart (see Figure 8.3). By looking at the height of the bars, learners can easily see which errors they made most commonly in their productions related

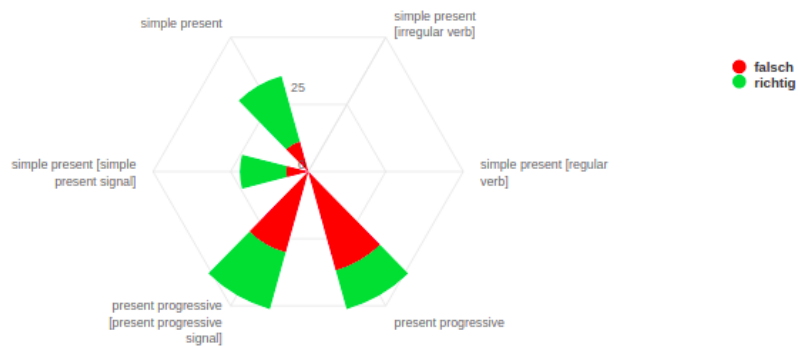
Lernermodell betrachten

Hier werden Statistiken zum Lernverhalten angezeigt. Bitte unten auf eine Kategorie klicken um mehr zu erfahren.

future tense

present tense

Richtig vs. Fehler



Highcharts.cc

simple present irregular verb

simple present regular verb

present progressive
Bitte üben! ★★☆☆

present progressive present progressive signal
Hervorragend. Weiter so! ★★★★★

Figure 8.2: Learner model, taken from Rudzewitz et al. (2020)

to this grammar topic. For example in Figure 8.3, this learner's most common error with respect to the present tense was a confusion of the will future with the simple present.

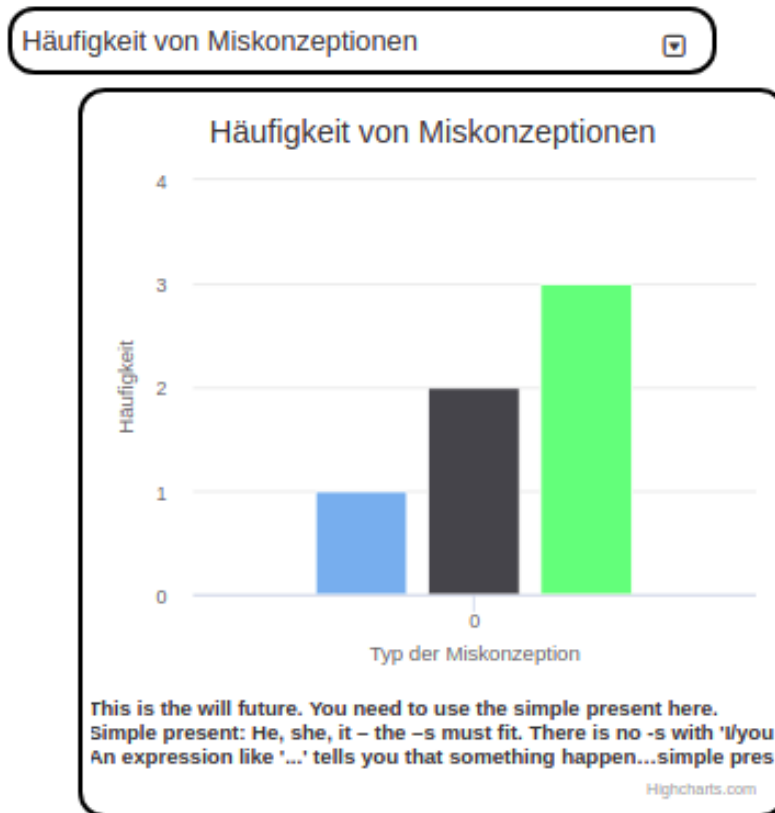


Figure 8.3: Typical errors visualized for the language topic *simple present*, taken from Rudzewitz et al. (2020)

8.3.3 Longitudinal Modeling of Learning Paths

Learners can inspect how their observed performance in the system developed over time for individual grammar points. Figure 8.4 shows an example of the development of the grammar topic *simple present* for a learner over time. The x-axis has a unit of 1 and has one value per day since the day this construct was used for the first time. In the given example, the learner used the construct for the first time 150 days ago. The y-axis shows the number of observations on the given day, with red showing incorrect usage (recorded misconceptions), green indicating correct usage at first try (uptake type *correct at first try*), and yellow representing successful uptake (uptake types *answer fully correct after*

feedback, correct, but not necessarily at first try, and answer ultimately correct).

Initially, the learner required some feedback, but got the language form correct after feedback (yellow bars). For some forms, the learner also got them correct at first try without needing feedback (green bars). Then, between around the 20th day, the learner suddenly made a number of errors related to this form. There can be many reasons for that; it could be that the learner went from receptive to productive tasks, or it could be that this language form was used in the context of another, new or complex language form, e.g. conditionals type 1. From day 40 to day 70, the learner did not practice this grammar construct. When he or she practiced it again for some days around day 80, the results were mixed with some incorrect forms and some feedback required initially, but later the positive evidence outweighed the negative by far. After a long break, this learner used the construct again 150 days after the first usage and was able to produce the form correctly.

8.3.4 Adaptive Sequencing of Material

The learner model in FeedBook contains an interface that suggests to learners exercises that target a selected grammar topic and that match, in their difficulty, the current proficiency level of this learner for the targeted grammar topic. As described in section 5.2.1, the system contains 154 exercises from accompanying material to the school book with every exercise available in epistemic difficulty levels 1 (easy), 2 (medium), and 3 (hard).

Every exercise in the system was manually assigned a set of grammar topics that represent the *pedagogical* goal of the exercise. Additionally, the feedback generation mechanism recognizes which grammar topics are targeted for every item (see introduction to chapter 8). The combination of this information allows to determine which exercises either explicitly via the pedagogical goal, or implicitly via the required language forms, target a specific language form.

The learner model makes use of these information sources to make proficiency-dependent, construct-specific suggestions for exercises to learners via the open learner model. When learners click on “Vorschläge für Übungen”, the system presents to them exercises that are selected based on the language construct selected and the learner’s observed difficulty level. If the positive evidence outweighs the negative evidence, the system thinks that the learner is good at this topic and suggests exercises of difficulty level 3. For example in Figure 8.5, the system marked the grammar point *simple present* with green color. For this reason, all the exercises suggested by the system are at difficulty level 3 (hard) so that the learner is challenged and not bored. If the language construct has been marked as yellow, the system proposes exercises at difficulty level 2, and for cases where the negative evidence outweighs the positive evidence, the system presents exercises at the easiest level for the learner, allowing them to start practicing with introductory exercises. In essence, the system attempts to suggest exercises that lie in the Zone of Proximal Development (Vygotsky, 1978) for this learner given the specific language construct(s).

This interface is adaptive since when learners are working on one of the

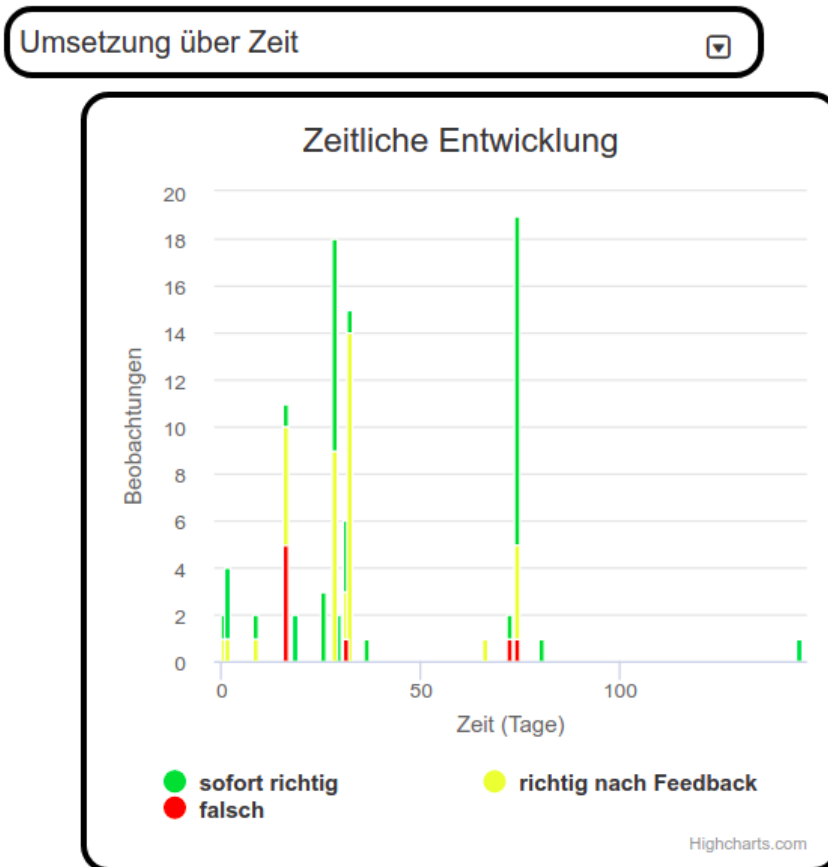


Figure 8.4: Development of competence over time (green: correct usage, red: incorrect usage, yellow: successful uptake), taken from Rudzewitz et al. (2020)

simple present

Hervorragend. Weiter soll ☆☆☆

Vorschläge für Übungen

Empfohlene Übungen:

<p>Some advice 3</p> <p>Theme 1 AP 6</p>  <p>S.12</p>	<p>👁️</p> <p>IT CAME FROM OUTER SPACE 3</p> <p>Theme 3 AP 3</p>  <p>S.83 ↻</p>	<p>PRESENTATION DOS AND DON'TS 3</p> <p>Theme 2 AP 6</p>  <p>S.51</p>
<p>💡</p> <p>Writing a txt 3</p> <p>Theme 1 AP 27</p>	<p>💡</p> <p>A FAB CLASS TRIP 3</p> <p>Theme 2 AP 16</p>	<p>💡</p> <p>IFS ABOUT THE INTERNET 3</p> <p>Theme 4 AP 6</p>

Figure 8.5: Proficiency-appropriate sequencing of material in learner model, taken from Rudzewitz et al. (2020)

suggested exercises, the learner model collects traces about the performance of learners for this grammar construct. When the learner uses the suggestion mechanism again next time after having worked on a task, the suggestion mechanism takes this evidence into account and potentially presents different suggestions to the learner. For example, it is possible that a learner starts with more negative than positive evidence and gets exercises at difficulty level 1. Since these exercises are the easiest, the learner is able to work on them successfully. After having shown enough positive evidence, the system shifts to suggesting exercises on the medium level. Once the learner was able to produce correct forms at this stage, the system finally suggests the hardest exercises at difficulty level 3.

Chapter 9

Modeling School Classes

In this chapter, we present Learning Analytics tools that make interaction data of school classes available to teachers in a way meeting their demands. In chapter 8, we presented Learning Analytics tools for individual learners. For this chapter, we will zoom out from an individual learner to a group of learners in the form of school classes. Since school classes are managed by teachers, the tools presented here are designed with teachers as the target user group.

We argue that it is crucial to provide teachers with Learning Analytics tools since they need insights into the learning process and not only the learning products. In a situation with distance education (with spatial and geographical distance between teachers and learners), as well as tasks traditionally performed by teachers such as the provision of feedback or revision of tasks outsourced to a tutoring system instead of to the teacher, most of the data that allow for diagnostic information about students is part of the interaction data. In a tutoring system with helpful feedback, more correct answers will be submitted to teachers, enabled via the assistance by the system. However, while many answers will be correct, the learning path towards these answers can differ. For example, learners can reach a correct answer with a correct attempt at first try, or it can take them a long interaction path with a range of different errors and reactions to feedback by the system. We get back to this observation in chapter 11.

In this regard, Learning Analytics tools are crucial to shed light on the learning process that otherwise would constitute a black box to teachers. We explain a range of different Learning Analytics tools that provide teachers with different interfaces that offer different perspectives on the interaction data of school classes by making data accessible from the perspective of learners, from the perspective of items, and from the perspective of tasks.

In section 9.1, we start off with an analysis of the needs of teachers in the educational context. Section 9.2 presents a Learning Analytics interface in FeedBook offering an item-specific, fine-grained analysis of learner interaction with the item so that teachers can see how specifically the learners in the school class interacted with this item. Section 9.3 takes a different turn by allowing

teachers to see which and how many tasks the learners have completed so far in the school year, and which tasks are still pending. We show an interface that enables teachers to inspect learner logs in section 9.4. In contrast to section 9.2, the data is not aggregated per task field, but visualizes the actual interaction logs over time as produced by the learners. Finally, in section 9.5 we showcase an interface that allows teachers to see at one glance before class what the most frequent problems were for a specific task and inspect selected interaction quantification metrics like different types of errors, successful uptake, time-on-task, effort, and more. In the following, we will present school class Learning Analytics tools offering perspectives on data per learners, per items, and per tasks.

Disclaimer: Specific sub parts of the contents of this chapter have been mentioned or referenced in Rudzewitz et al. (2020); Meurers et al. (2019b) and are cited accordingly in the running text of the respective sessions.

9.1 Needs Analysis: Teachers

The basic need of teachers in an educational context is to get information about the proficiency of learners with respect to the topics the teacher covered in class. Apart from knowing about who does well, teachers especially need to help learners who have problems. In order to address the problems, teachers need to know about problems of specific learners and the school class as a whole. Ideally, they can get this information for specific items, for exercises comprising of multiple items, and across exercises.

In a traditional approach with submissions on paper, teachers can (to a certain extent) see what learners write on their first attempt. These answers are valuable for the teacher as a diagnostics instrument since the correctness or errors in the answer provide the teacher with clues about where the learner shows strengths and has misconceptions.

This is not the case in tutoring systems, since a tutoring system like the FeedBook is designed to guide learners towards a correct solution before submitting to the teacher. As a consequence, assuming that the tutoring system provides helpful feedback, more correct than incorrect answers will be submitted to the teacher. This raises a need for teachers to get insight into the learning process that led to the correct solutions in order for the teacher to still get some diagnostic information about the students in class. If a tutoring system does not provide this process information, this creates a 'black box' in which many learners submit correct answers, but potentially needed a lot of attempts to arrive there, without the teacher knowing about which problems occurred on the way.

Since one task of teachers is to help weaker students to catch up with stronger students in the subject at hand, teachers should know by themselves or via a tutoring system which students are at risk (and in which areas), so that an intervention by the teacher can be planned. A student-at-risk detection system needs to know what the expectations are so that it can compare the learner

model to the reference and detect deviations. In a milder version, teachers at least need to know where they need to put in effort next, i.e. grade submissions and provide feedback to learners with open submissions.

Another need of teachers is to see where learners are in terms of their progress. This can include local aspects like how many students did how much of the assigned homework for the next session, probably also with respect to the next exam, but also globally over the school year and the material that needs to be covered in the school year and respective curriculum.

During the school year, and especially in a system like the FeedBook where learners have a free choice to work on exercises even if they have not been assigned explicitly by the teacher, teachers need to know the progress of the school class for a specific chapters so that they know when the learners have invested sufficient work so that they can move on to the next chapter.

Finally, a need shared by learners and teachers is to provide an opportunity to reach the actual person sitting in front of the screen. This includes being able to receive and send messages for a range of purposes like class management or asking and answering clarification questions. Teachers need to be reachable as a partner, also through the system, since some of the interaction that previously took place face-to-face is now mediated via the computer.

While addressing all the outlined needs would go beyond the scope of this thesis, in the following we will explain how some important needs are addressed. Section 9.2 presents an interface that allows teachers to zoom in further and get analytic information on the level of individual items. In section 9.3, we present interfaces that inform teachers about the progress of learners. This is complemented by an interface showing log data to teachers for specific exercises (section 9.4). Finally, we show an interface that allows to visualize the interaction data of learners on a specific task in section 9.5.

9.2 Item-Level Performance Analytics

In a typical setup, teachers assign one or two exercises to learners for homework. When a paper workbook is used, the teacher would ideally collect the test papers and grade them. Due to time constraints, often homework is discussed in class and individual students need to present their answers to the teacher. In the case when a learner presents an incorrect answer, the teacher would explain to the students why the answer is wrong and how to fix it.

In a situation with a web-based tutoring system like the FeedBook, teachers can assign exercises in the same way. The difference, however, is that the system provides formative feedback during the exercise completion process, guiding learners towards a correct solution. What the teacher then sees as learner submissions is not comparable to the submissions on paper since it represents the result of an interaction of potentially many turns between the system and the learner. This leads to the situation that a teacher can receive only correct submissions, but on the way towards reacting the correct solutions, the learners showed a lot of problems and potentially were even attempting to guess the

correct answer. In this scenario, when teachers only look at the final result, they will be misled into thinking that the learners are well-prepared on this topic, e.g. for the next exam.

What is instead needed to address this problem is a Learning Analytics interface that presents not only the learning *product* data, but additionally the learning *process* data. In FeedBook, this is addressed in an interface that provides a fine-grained analysis of the interaction of learners of a school class with the feedback mechanism on the level of individual prompts. This level of granularity relating to prompts was chosen since this allows teachers to zoom in on individual items, just like in a traditional setup for discussing homework results.

The interface shown in Figure 9.1 provides a list of all tasks that offer scaffolding feedback. When a teacher selects one of the tasks, the system loads the exercise in the exact same way as the learners saw the exercise, with one notable exception: instead of the question mark symbol for requesting feedback, there is an arrow symbol. This symbol is associated with a click handler that allows to load the learning process data related to this item for the students in the school class of this teacher.

Upon clicking on a gap or the symbol next to it, the system loads a popup for item-level learning analytics. At the top of the popup, the system shows the target answer(s) of the item in green. Below this, a pie chart is rendered. Every slice of the pie chart represents a feedback template that was computed in the interaction with this item. This also includes feedback that was computed, but not displayed due to the blacklist filter. The size of the slice depends on the frequency of the provision of the feedback template, with more frequent templates taking up more space. This allows the teacher to see at one glance what the most common problems of the class were for this item. For example in Figure 9.1, the biggest slice with 45.38% is associated with a template about the usage of the past progressive in relation with the signal word 'while'. The teacher thus immediately knows that the class struggles with the fact that 'while' is a signal word for the past progressive.

The figure, however, shows more: the second biggest slice of the feedback pie with 42% is default feedback, i.e. cases where the system could not meaningfully interpret the learner utterance. This points in the direction of limitations of the feedback capabilities by the system and could potentially be relevant information for system designers (see chapter 10). However, as a trained professional, a teacher can recognize what the problems were. To this end, the system allows to zoom in further for every slice of the feedback pie chart. Below the chart, the system offers a responsive table that, for every slice in the pie chart, allows to query more detailed information. The elements in the table are ordered by decreasing frequency, thus mirroring the pie chart information at the top level. In this example, the feedback related to the usage of the past progressive with the signal word 'while' is consequently displayed at the top. In addition to the feedback message, the top level elements contains a badge with the absolute frequency of the computation of this feedback for this item. In the example, this feedback message was computed 393 times. Upon clicking on the element,

the system displays all the types of learner answers that triggered this feedback template. The types are ordered by the frequency of occurrence, which is displayed in square brackets behind the actual learner answer. In the concrete example, the most frequent answer with 15 occurrences was “*Charlie was buying Arsenal tickets while Gillian sit on the bus.*”. The fact that the most frequent type occurred only 15 times out of 343 cases, representing only 4% of the learner answer tokens, illustrates (a) how much variability there is in authentic learner answers and (b) that the system’s flexible matching algorithm is both necessary and working (cf. also Meurers et al. (2019b,a)).

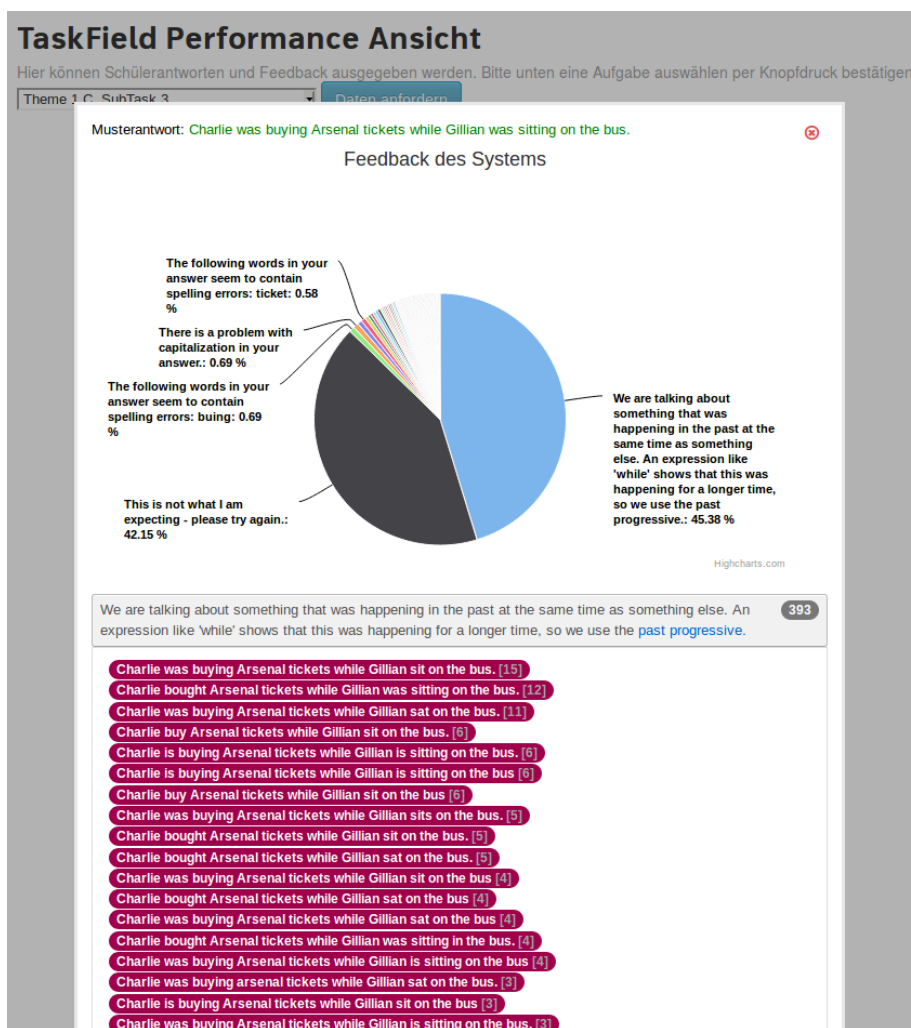


Figure 9.1: Prompt-specific learner answer analysis, taken from (Rudzewitz et al., 2020)

9.3 Learner Progress Visualization

Teachers need to know how much of the material assigned has been covered by students in their class. In a traditional face-to-face teaching setup, teachers can collect filled out work sheets or they can walk through the rows and put a check mark for every student who shows that they completed the assigned tasks. In a tutoring system, teachers also can get insights into who submitted what. While the teacher lobby view (cf. section 5.3.2) lists all the submissions for one specific task, but not a list of tasks. While teachers can also click on “Abgaben” at the top menu (cf. Figure 5.19) and receive a list of all submissions sortable by tasks, this list does not include the delinquent students who did not submit, but only the actual submissions. So in order to display a list of exercises and the progress of learners in terms of the completion status on them, an additional Learning Analytics interface is needed. Figure 9.2 shows a full version of the widget displayed in minimized form at the bottom left of Figure 5.19. While the widget can be minimized, it can not be closed and is loaded in a maximized version every time a teacher logs into the system. The content of the widget is a completion progress overview for the ‘core tasks’ of specific chapters (cf. chapter 7). The core tasks of the current chapter are displayed at the top to increase the salience, with the other core tasks displayed in the order of completion of the chapters below. In the example shown in Figure 9.2, this teacher was working on theme 3 of the workbook at the moment. The interface allows teachers to recognize at one glance for which of the tasks that need to be completed in this chapter the completion rate is significantly different from 100%. For the tasks with identified lack of coverage, teachers can then look into who specifically did not submit the tasks yet via the list of submissions per tasks described in section 5.3.2.

9.4 Learner Log Perspective

While interfaces such as the item-level performance view (section 9.2) or the task-level performance analytics (section 9.5) present information aggregated across all interaction steps, for an interpretation of learner data from a pedagogical (and also from a system development) perspective, an approach different from aggregating data across the interaction steps is to visualize the interactions as a process. Figure 9.3 shows an interface which, after selecting a specific exercise from the system, visualizes the interaction steps. In this interface, the interaction of learners is “replayed”. Unlike in the teacher correction interface (section 5.2.6), in which only the final submitted answers are displayed, this interface shows the steps taken by learners towards these final submitted answers. The system displays the data per learner, with learners separated by line breaks and whitespace. For every logging token of a learner, the system displays one line. The first element is a time index, which represents a number that is counted up every time a new `LoggingToken` is added to the interaction log. This number allows to reconstruct the order in which the learner answers

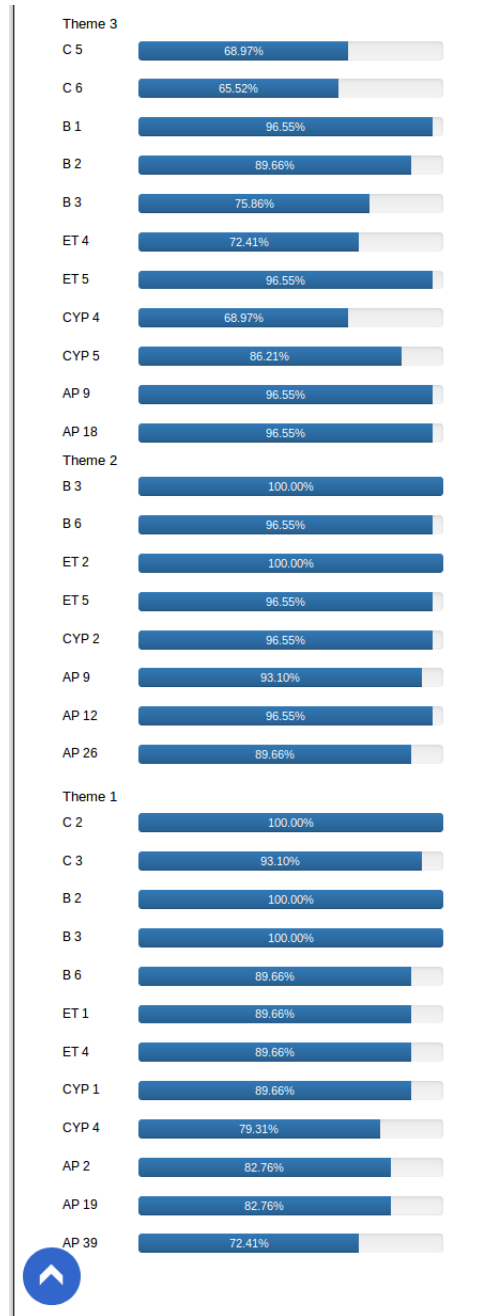


Figure 9.2: Visualization of learner progress over core tasks

were entered. While the logging token data structure also contains a time stamp for every interaction, the readability of the interface would suffer if instead of the time index a time stamp would be displayed. The second element for each interaction is the index of the task field. This allows to see for which specific item this answer was given. Again, an index is used for readability of the interface. After this, the system displays the actual learner answer, followed by a green check mark if this interaction took place for an answer that was evaluated as correct. If the submitted answer was not correct, a red cross is displayed. In this case, after the red cross, the system displays the feedback message that the system computed for the given learner, item, and learner answer.

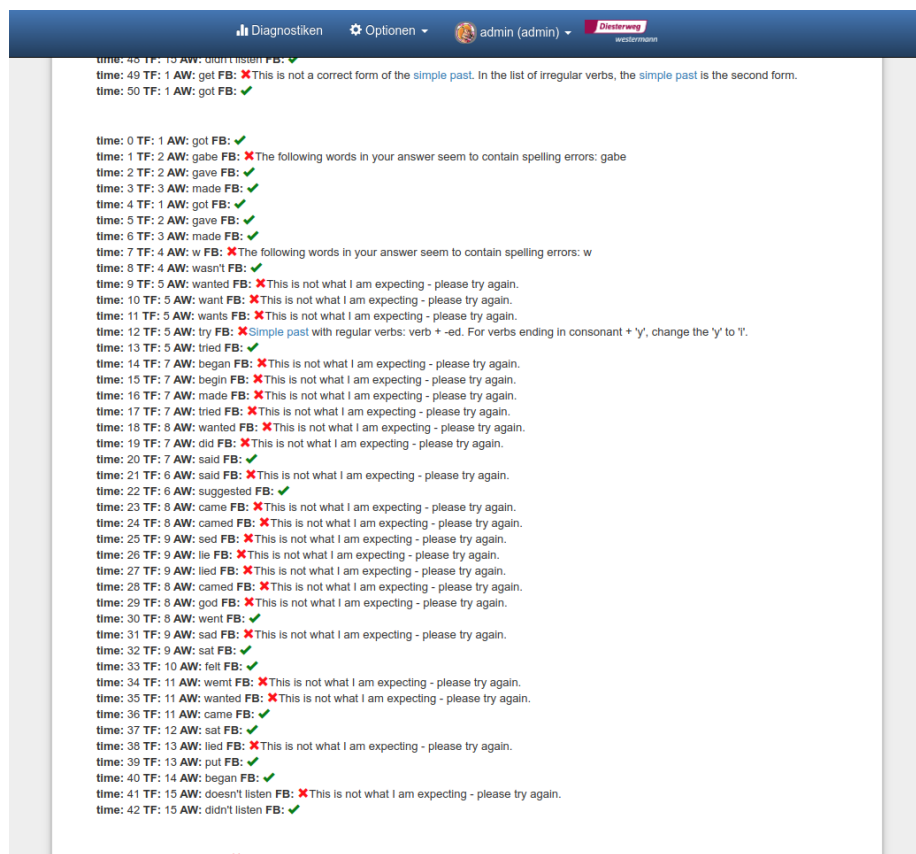


Figure 9.3: Log data view

This interface allows to see *how* learners interacted when working on a given task. It visualizes the steps they took for reaching their final submissions and how they, in negotiation with the system, worked step by step to arrive at what they submitted. It also allows to get insight into task strategies employed by learners. For example, in the example in Figure 9.3 at the top, it can be seen

that first the learner works his or her way up to item 15 (time point 48), but in the end of the interaction (time points 49 and 50), jumps back to the first item and manages to provide a correct solution for it, after before having not managed to solve it. While this interface can give teachers the capabilities of coming up with fine-grained diagnoses of the problems of learners for a specific task, this interface can also provide valuable insights to material designers (see chapter 10) about which exercise design triggers which interactions on the task and of the learners with the feedback mechanism.

9.5 Task-Level Performance Analytics

Figure 9.4 shows an interface that visualizes interaction metrics for sub tasks (i.e. individual parts of tasks in the system). At the top, the system allows to select an exercise, along with two options. the first option allows to restrict the data only to a specific group of learners (no filter, only control group, only intervention group). The second option in the form of a check box serves to indicate whether automatic logging tokens should be included or filtered out for the interface. Confer section 5.2.5 for a discussion of different types of logging tokens.

At the top of the interface, the system displays a pie chart that indicates the percentage of each type of feedback computed while learners were interacting with the exercise. Each slice of the pie chart stands for one feedback template, with the size visualizing its frequency. In the example in Figure 9.4, the default feedback was the most frequent template with 61.5%. The most frequent specific template with 10.25% out of 5,374 interactions, equaling 551 instances, is a template that indicates an incorrect formation of the simple past for an irregular verb. The second most frequent specific template is concerned with the formation of simple past forms for regular verbs. Followed by this is a template that tells learners that they confused the simple past and the simple present form of a verb. Note that this pie chart is computed for all task fields/items of this task. While in section 9.2 we show a pie chart visualizing interaction data in the same way, the difference here is that in the interface shown in Figure 9.4, the data takes into account all the interactions for a task and not only for a specific task field. With this interface, teachers can see globally for one task what the most common problems are. This visualization of complex interaction data in a simple way for an entire exercise was implemented in this way since the time of teachers is very limited and before class, they might only have a few minutes to prepare the discussion of homework, underlining the need for a simple and easy-to-understand user interface.

In the second part of the interface, a bar chart with interaction metrics is shown. The metrics displayed there quantify the interaction in terms of errors, correct answers, time-on-task, uptake, and effort. A detailed explanation of every metric displayed can be found in Table 10.1 in section 10.3.2.

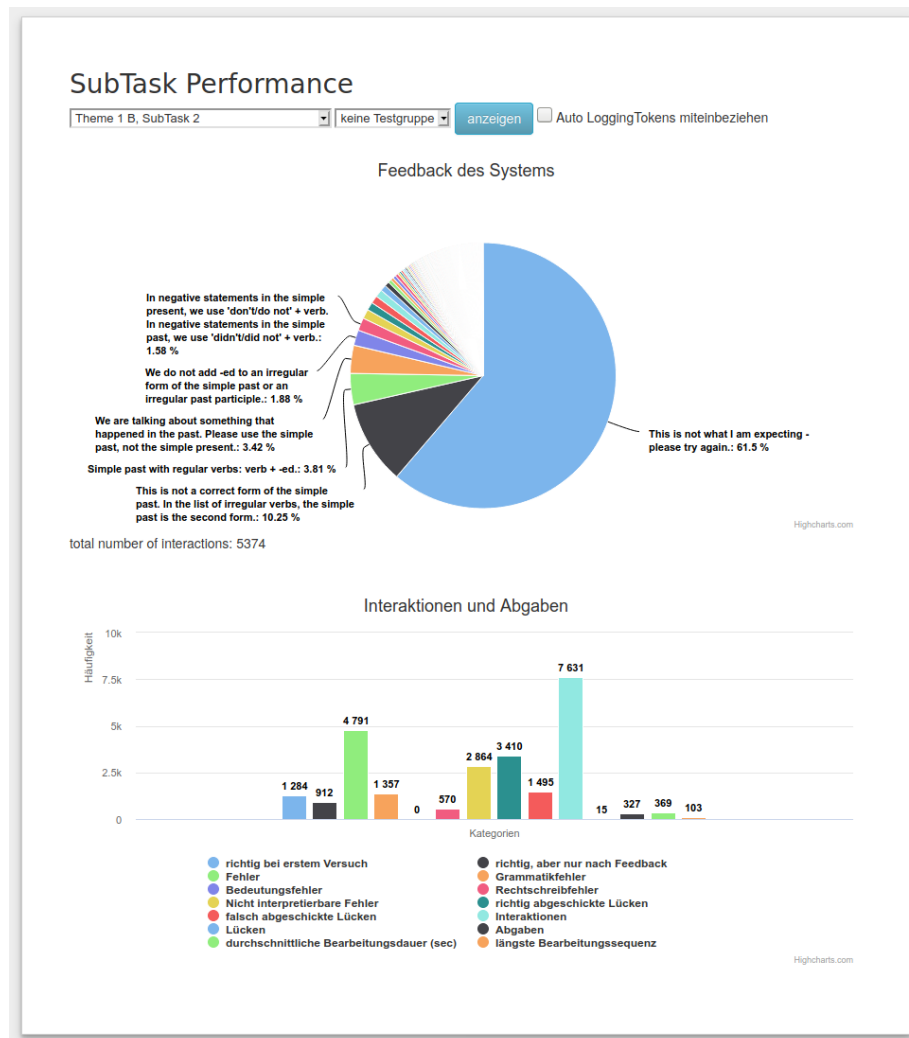


Figure 9.4: Task Performance View

Chapter 10

Modeling the Learner Population

Moving from Learning Analytics modeling single learners in chapter 8 over Learning Analytics about school classes that form groups of learners in chapter 9, in this chapter we now turn to explaining Learning Analytics for modeling the entire learner population in this chapter.

The Learning Analytics functionalities are designed for different stakeholders involved in the educational context, but with a shared interest for Learning Analytics that model more than one school class. In section 10.1, we outline the needs of material designers, primarily related to functions that allow them to evaluate how well tasks are designed based on the interaction data from learners working on these tasks. In section 10.2, we take a different perspective and analyze certain needs of educational administrators. Their needs center around observing the progress of school classes over time to ensure persistent high-quality teaching and coverage of materials by a set of school classes.

Having delineated the needs of the different stakeholders for which these Learning Analytics tools were designed, we look at interfaces that address these needs. In section 10.3, we showcase an interface visualizing the interaction data of learners. Part of this interface is the functionality described in section 10.3.1 that puts effort in relation to success. Another component of this interface displays key interaction metrics informed by SLA research (section 10.3.2). In section 10.4, we explore how to display to educational administrators how learners in school classes develop over time by displaying anonymized excerpts of learner models. Section 10.5 offers a different perspective on the learner population in the form of a user interface for material designers that enables them to inspect the well-formed and ill-formed variability elicited by tasks, and how the system handled it. As a follow-up, section 10.6 shows an interface designed for a data-driven extension of target answers based on actual learner answers. Based on the observed variability, material designers can add language forms that were previously not anticipated in the task design. Finally, section

10.7 concludes the chapter with an interface visualizing the completion rate of learners over selected mandatory core tasks to educational administrators and researchers conducting studies.

Disclaimer: Specific sub parts of the contents of this chapter have been mentioned or referenced in Rudzewitz et al. (2020); Meurers et al. (2019b) and are cited accordingly in the running text of the respective sessions.

10.1 Needs Analysis: Material Designers

Material designers are users that take on the role of creating educational materials in the system and can provide input on the system's behavior with respect to its feedback behavior. The role of material designers can be shared between persons who work(ed) for a publishing house as authors and write exercise texts etc., and system designers that make the system behave and make the exercises look as specified by the content authors.

Material designers, like for example representatives of publishing houses or users who create content, need to be able to make observations that are informed by data from as many users as possible to be able to distinguish trends from outliers. This implies they need to be given the opportunity to look at the data aggregated across school classes without being able to identify or track individual learners or teachers.

An important task of material designers is to detect and correct situations with misunderstandings related to the design of exercises. In order to be able to find answers to these questions, teachers first need to know which exercises learners work on or which exercises are avoided by the learner population. The next question is to get insights into whether an exercise is too hard, just at the right level, or too difficult for the learner population. Closely related to this is the need to have means to answer the question of whether learners actually benefit from working on an exercise. If yes, they should be enabled to answer the question of why an exercise worked well, and how this knowledge can be transferred to the design of other exercises. If not, they need to be able to distinguish whether the feedback mechanism or the design of the exercise, or a combination of both are responsible for the problems observed with this exercise. Approaching the system designers with data-driven observations and being able to edit an exercise on-the-fly allows to improve the system and solve the observed problems.

10.2 Needs Analysis: Educational Administrators

Educational administrators are people in positions to decide about the contexts for teaching. In the school context, this could be the headteachers of a school. In their role, they need to satisfy a range of users including the learners, teachers, parents, and potentially decision makers on a higher level of administration like

school supervisory boards. The latter group includes decision makers that look at performance indicators like grade averages.

Needs that administrators can have in conjunction with a tutoring system can be to see whether, across school classes, the learners actually benefit from using the system. When learners benefit, both the learners and their parents will be satisfied with the education and thus the school. Since the administrators need to satisfy and care for the teachers employed by them, a relevant question with respect to teachers is whether teachers save time working with the system in comparison to colleagues not using such a system. If a system can help keep work time in a reasonable frame, it is beneficial for all stakeholders. Closely related to that is the need to know if the system is easy to integrate into the regular teaching and whether it creates hurdles for teachers, learners, or potentially parents.

Not in the case of FeedBook, but in general for systems with a commercial license, educational administrators decide whether the school spend money on the acquisition of commercial licenses for a tool/systems. In order to answer this question, they need to be able to see how effective a system is, and how effectively it is used. One indicator for this is the completion of exercises. Educational administrators should be able to get insights into the progress of school classes in the curriculum. To this end, they need to see if some school classes are lagging behind other school classes in this school.

In the end of a school year, the decision makers in the educational context should be able to see how the final grades of learners using the system behave in contrast to learners from previous years or learners from school classes that did not use it.

10.3 Interaction Performance Metrics Perspective

Figure 10.1 shows a Learning Analytics interface that visualizes performance metrics based on interaction patterns. The interface consists of three components: a selection of a specific task from the system at the top, a line chart visualizing effort in relation to task success in the middle, and a bar chart plotting different error and uptake metrics next to each other. In the following, we will explain the different components in detail.

10.3.1 Effort in Relation to Success Perspective

The line chart in Figure 10.1 visualizes the effort learners put into an exercise in relation to their ultimate success or failure on the task. As a first step, the interface makes explicit what the data source used in the current view consists of. In the example in Figure 10.1, the data underlying the view consists of a total of 264 submissions by learners, of which 170 were submitted without any errors in the final submission to the teacher, and 94 submissions with at least one deviation from one of the target answers. The relation of ultimately correct vs.

Taskperformance

Theme 1 C, SubTask 3 [short answers] anzeigen

264 Abgaben (170 vollständig korrekt, 94 fehlerhaft)

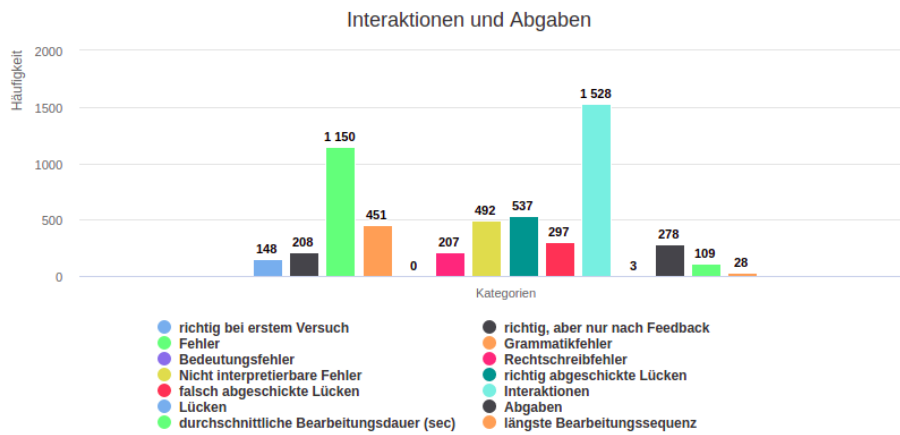
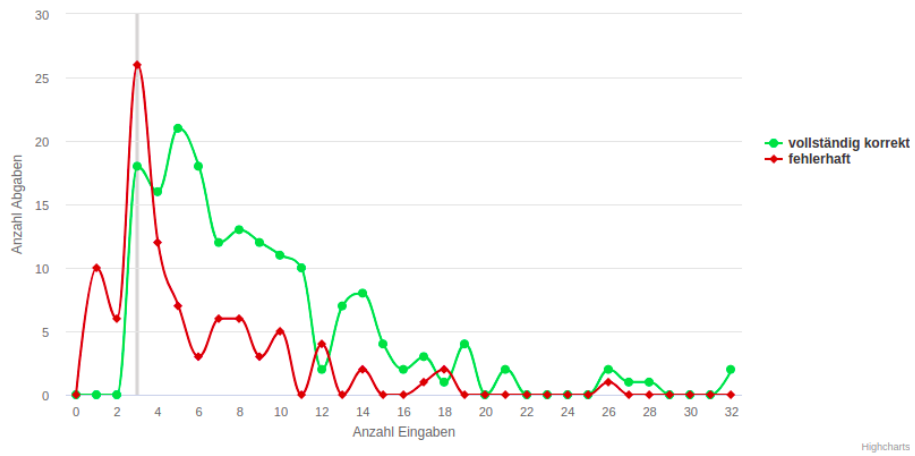


Figure 10.1: Interaction performance metrics, taken from Rudzewitz et al. (2020)

incorrect submissions can be a first indicator of how well this exercise matches the ability of the involved learners.

Below this, the system renders a line diagram. On the x-axis, the system visualizes effort operationalized as the number of attempts learners spent on this task. Every interaction with the feedback mechanism increases the value on the x-axis by one. On the y-axis, the system plots the number of submissions with as many attempts as the x-axis indicates. It is a frequency of submissions with the indicated number of attempts and expresses how many learners submitted this task to the teacher with this many attempts. In essence, this interface visualizes how many people submitted with how many attempts and whether they were successful.

Since every exercise can have a varying number of items, in order to be able to interpret the displayed values, material designers using the system need to have the opportunity to put the number of attempts in relation to the number of items – ten attempts on a task with ten items should be interpreted differently compared to ten attempts on a task with two items. To this end, the system indicates the number of attempts in the chart with a vertical gray line. In Figure 10.1, this is visualized at the x value of 3, meaning that this task consists of three items that require learner input.

The actual values for the line chart are plotted in green and red. Data points visualized in green represent learner submissions that were submitted with all learners answer evaluated as correct by the system. For grammar tasks, this implies identity to one of the target answers. For meaning tasks, this can also be the case, but since the evaluation of learner answers for meaning-based activities allows for answers that are treated as equal due to similarity measures rather than string identity, the system in order to enable this visualization checks in the interaction log whether there has been a logging token without an associated error code for a specific task field. Given that it is important that an interface like this loads fast, a re-evaluation of learner answers with the memory-intensive alignment approach (cf. section 6.3) is not feasible at runtime. Instead, the system uses a shortcut by checking whether an answer, and projected from that a submission for a task, has been evaluated as correct in the past and uses this as evidence. For all submissions for which at least one answer has not been evaluated as correct, the system adds a data point in red into the chart (cf. also Meurers et al. (2019b,a)).

Let's look at the example in Figure 6.3 in more detail. All the data points left to the gray line are marked in red. The reason is that it is not possible to submit an exercise with all items answered correctly if the number of interactions is not at least equal to the number of items. In other words: these data points represent submissions of learners who did not provide an answer to all items. At the x axis value 3 indicating the number of items, there are 18 fully correct and 26 not fully correct submissions. This means that 18 learners managed to submit a correct answer for every item on their first attempt. In contrast, 26 learners submitted after interacting with each item only once (another hypothetical explanation would be they interacted with fewer items but more often). To the right of the gray line (i.e. $x > 3$), the trend is that the green line is above

the red line, with only two exceptions at $x = 12$ and $x = 18$. The interpretation for the observed values is that for the majority of learners, interacting with the exercise led to ultimate success. In other words, this trend shows that it pays off for learners to interact with this task since the likelihood of reaching a correct submission in the end is higher than the likelihood of submitting something to the teacher that contains errors. Another observation related to this interface is that the maximum of correct submissions is at $x = 5$, followed closely by $x = 6$. 22 learners submitted a correct submission for $x = 5$ versus 7 learners with an incorrect submission. This shows two things: firstly, the exercise is designed in a way that in tendency is not too easy in the sense that the highest number of submissions can be observed for learners who on average had two interactions per item. Secondly, the effort learners invest into interacting with the task is not in vain since it is much more likely to get it right in the end than otherwise.

Compare this to the chart for a different exercise depicted in Figure 10.2. First of all, this exercise was not remotely as popular as the other task, with only 15 submissions in total. A reason for this could be that it was part of the *Additional Practice* section with optional tasks. Secondly, the maximum number of submissions is at $x = 1$, meaning learners only interacted once with this exercise offering eight items and decided to submit directly. At $x = 11$, the only two correct submissions for this task can be observed. One learner interacted 19 times, but was still unable to reach a correct submission. The visualization for this exercise shows that learners were both not motivated to put effort into this task, and for those who were motivated, the effort did not pay off. The likelihood of reaching a correct submission is much lower than reaching an incorrect submission. This can be an indicator for material designers to check the validity of the design of this exercise.

This interface can answer the question of whether an exercise is appropriate for a specific learner population and whether effort of learners invested in this specific task pays off for them.

10.3.2 Uptake Metrics Visualization

In addition to the line chart, the system plots different quantities that express features related to interaction patterns of the given exercise in a bar chart. The quantities are plotted in different colors to visually distinguish them, and plotted in one graph in order to make the quantities comparable. The purpose of visualizing these quantities is to give material designers an accessible and easy-to-interpret overview about how learners interacted with this task that goes beyond only ultimate success or failure. In chapter 11, we analyze the predictive power of (some of) these features with respect to learning gains.

Table 10.1 lists the features in the order in which they are visualized in the chart from left to right. Along with the description, the table lists a short explanation for the conceptual motivation of the respective feature. Every feature is computed on all available data (i.e. across all school classes) for this task in the system.

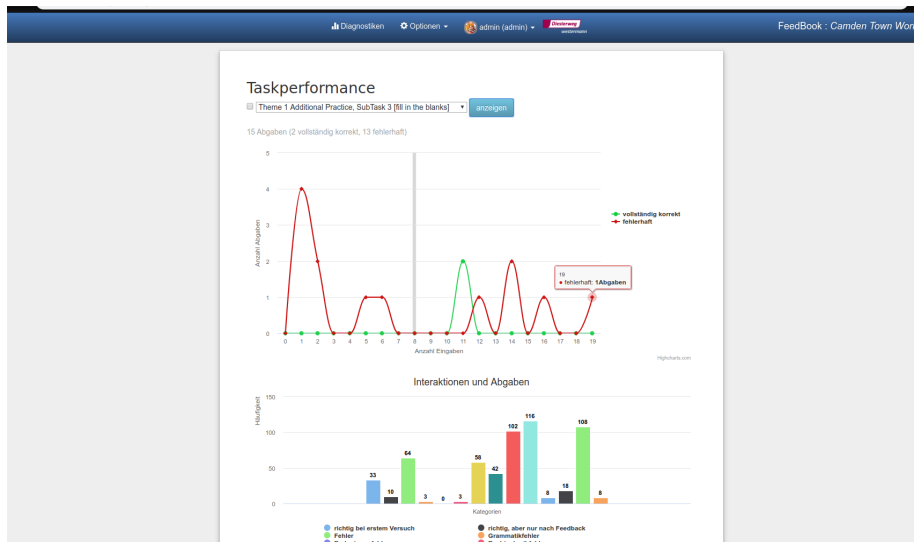


Figure 10.2: Interaction performance metrics for an exercise that shows problems

The first feature, *correct at first try*, counts the number of uptake observations of the associated uptake type, as described in section 8.2. It can be interpreted as a proxy for previous knowledge since these are cases where learners only need to interact once and in this one interaction show that they already know the solution and can thus use the associated grammar construct in this context correctly.

The next feature, *errors*, expresses the total number of learner answers that were submitted during the interaction that were evaluated as not correct. Since this category includes all the different types of errors handled and not handled in the system, it expresses how many errors in general were committed in this exercise. Material designers can see whether this task in general elicits many errors or not and can put this into relation with the number of items visualized in the chart described in section 10.3.1.

In contrast to the feature *errors*, which expresses the errors committed during the interaction, the feature *incorrect answers* indicates how many incorrect answers (or to be more precise, how many answers not recognized as correct by the system), were submitted to the teacher. Technically, this is a count of uptake types of the sort *answer ultimately incorrect* (cf. section 8.2) The interpretation of this feature is the number of instances (on an item level) where the system could not provide feedback that led to a correct solution, including cases where the feedback was disabled for the control group. The alternative interpretation is that the system provided feedback that could have led to a correct solution, but learners did not use the feedback to correct their answers before submitting. This again can have a range of reasons (see also section 2.2), including motiva-

tional issues or feedback that is not formulated in an understandable manner for a specific student.

Similar to that is the next feature, *correct answers*, given that it expresses the opposite case compared to *incorrect answers*. These are instances in which, for a specific item, learners submitted an answer to the teacher that was evaluated as correct by the system. The interpretation of this feature is that these are cases where learners either had previous knowledge or showed successful uptake to reach a correct solution, i.e. cases where the system offered helpful feedback leading to a correct solution. This feature thus expresses for how many cases it was possible to ultimately succeed in getting a correct answer.

The feature *interactions* indicates how many times in total learners interacted with the feedback mechanism. This feature is a summary statistic of the data points displayed in the chart described in section 10.3.1 visualized above. The same holds for the feature *submissions*, which simply indicates the absolute number of submissions to teachers for this task. This can be a proxy for how popular an exercise was (if it was not a mandatory task).

The next feature, *gaps*, is a measure of how many answers (on an item level) were submitted to the teacher that were not empty. A common problem is that learners submit tasks before having filled out all the gaps. This is evidenced in for example Figure 10.1 with the data points in red left to the gray vertical line. Another problem can be that learners submit their answers for one subtask without realizing that there is a second subtask they could switch to. This feature thus expresses the coverage for learner submissions for this task.

The feature *average time* is a measure of the average time-on-task learners spent for this exercise. Comparing the number of items with the average time-on-task allows to see how long the learners on average spent on the items in this task. Similarly, comparing the number of interactions with the average time-on-task allows to get insight into whether learners are putting effort into an exercise or just trying out a few answers and then submit quickly. Related to that, combining this with the line chart visualized above, it allows to relate the number of interactions with ultimate success and the time invested. For example, if the line chart shows it is unlikely that learners who invest effort into the system will reach a correct solution, plus the time-on-task is high, then this is an indicator that the design of an exercise could be problematic.

Encoded by the feature *correct only after feedback* is the number of uptake observations of the type *answer fully correct after feedback* (cf. section 8.2). This encodes a sequence of two interaction steps in which a learner went from an incorrect answer to a correct answer. In essence, this feature thus expresses instances of successful corrections by learners. Since the interaction takes place with the system as the interaction partner, these are cases in which the system was able to provide helpful feedback. With this feature, and especially in relation to the feature *errors* expressing the intermediate steps with errors, material designers can evaluate whether the feedback capabilities of the system are compatible with the exercise design, i.e. whether the feedback can guide learners to a correct solution in this specific task context.

In order to give material designers insight into the proportion of different

feedback types, the system visualizes the number of the sum of feedback instances for groups of templates. An important feature for material designers is the quantity of *default feedback*. Default feedback is only triggered in cases where the system is unable to interpret the learner answer. While it contains highlighting capabilities to indicate which parts of the learner answer need to be changed, which can be helpful for e.g. lexical substitutions, this type of feedback is the least specific version. Every instance of default feedback means that the system was unable to deal with the learner answer in a more specific way, i.e. did not find a rule or target hypothesis to interpret the learner answer. If this number is high, it is an indicator for material designers to look into more detail into what specific aspects of the learner answers the system was unable to interpret given the lack of coverage of specific feedback. This can be achieved with Learning Analytics functions like the item-level performance analysis perspective described in section 9.2.

The feature *meaning errors* expresses the number of problems related to the semantics of learner answers. This is completed with *grammar errors* representing morphological and syntactical problems, as well as *spelling errors* that tell material designers how many instances of orthographical problems the system detected. The amount of and relation between the different feedback types can give material designers insight into what types of problems the learners have in this exercise. For example, if the amount of spelling feedback and default feedback is very high, but the amount of meaning and grammar feedback very low, the exercise is unlikely to provide opportunities for meaningful interaction on the pedagogical target constructs associated with this exercise. In essence, these features allow to compare whether the frequency of different groups of errors triggered by the learners are in line with the pedagogical design of the exercise.

The last feature visualized in the bar chart is the *longest sequence*. It expresses the maximum length of a learner interaction sequence on one of the items in this task. The motivation for this feature is to offer a measure of perseverance of learners for reaching correct answers on this specific task. While it is sensitive to outliers and could in principle be complemented with measures of central tendency related to interaction sequence lengths, this feature allows for material designers to see if the exercise design on the item level is appealing enough so that at least one learner attempted in a continuous manner to solve an item. Both extreme cases, only a very short longest interaction, as well as a very long longest interaction, can be warning signs for material designers that an item is either not appealing enough to invite longer interactions, or that learners need to put in many interactions on one or more items. Note that this feature is conceptually different, being on the item level, from the line chart that aggregates attempts across items for one task.

While there are many features that could be visualized in addition to the presented features, a consideration in the design was to only visualize key information to material designers and to keep the interface interpretable in a reasonable time frame.

feature	description	interpretation
correct at first try	number of instances students filled in a correct answer at their first attempt	previous knowledge
errors	number of diagnosed errors	misconceptions
incorrect answers	number of answers submitted to teacher with errors	cases where the system was unable to lead to a correct solution
correct answers	number of answers submitted to teacher without errors	previous knowledge or successful uptake
interactions	total number of interactions with feedback mechanism	number of times learners requested feedback
submissions	number of submission to teachers by students	size of data set underlying visualization
gaps	number of gaps filled out	coverage of learner submission for task
average time	average time taken by learners for exercise	time on task
correct only after feedback	number of prompts where feedback led to correct solution (uptake)	number of cases where system feedback was effective
default feedback	learner answers with no associated diagnosis	lack of coverage of specific system feedback
meaning errors	diagnosed missing or additional information	semantic misunderstandings
grammar errors	diagnosed grammatical misconceptions	morphological or syntactic misunderstandings
spelling errors	diagnosed misspellings	orthographic misunderstandings
longest sequence	maximum number of steps for prompt	perseverance of learners for reaching correct answers

Table 10.1: Uptake metrics presented to material designers, taken from Rudzewitz et al. (2020)

10.4 Learning Path Perspective

Educational administrators need to keep track of how school classes are performing over the school year. At the end of the school year (and in the middle after one term has ended), each student receives a grade. Depending on the educational system, administrators might be obliged to report grades to authorities or higher educational institutions. Having grade averages that are below the standard or comparative schools can then have an impact on funding and benefits the next time resources are distributed.

Among the primary concern to foster the learning of students, this can be an incentive to ensure a high level of quality in the education of students resulting in decent grade averages. While it does not need to be (and should not be) a concern of administrators about who specifically performs how, identifying trends when comparing school classes can be an understandable need.

Figure 10.3 shows an interface that fulfills this need. It allows to select a specific school class at the top along with a language construct of interest. The interface then loads the temporal development of all learners of this class for this construct – without displaying the name of any of the students. These components constitute parts of the learner models (see section 8.3.3) and allow administrators to identify trends with respect to how the learners in this school class are developing on this language construct over time. Loading the perspective with different school classes allows to draw comparisons, identify successful instances for a specific construct, but also to see if a school class is underperforming for a specific grammar topic.

10.5 Answer Variability View

The type-token ratio (Richards, 1987) indicates how many strings versus how many *different* strings are present in a specific scenario. A low type-token ratio indicates that there are not many different variants, whereas a high ratio indicates that there is a lot of variability.

Figure 10.4 displays the answer variability for one item in terms of types and tokens. The interface displays the information for both well-formed and ill-formed variability. For grammar tasks, the number of correct answers is identical to the number of target answers (or only has minimally higher numbers due to variance caused by expanded contractions etc.). In contrast, for meaning-based activities, the correctness of an answer does not depend on a string identity with one of the target answers. In this case, the number of correct answer types or tokens can be higher.

Ideally, the variability observed in an answer contains systematic patterns covered by the feedback algorithm, or, if the patterns are not covered, systematic patterns that could be covered if the feedback algorithm would be extended. On the other hand, if the variability shows that learners did not know what to do in this specific task, with answers that contain parts unrelated to any of the target answers, then this shows a problem with the design of an exercise that



Figure 10.3: Visualization of uptake over time

needs to be tackled by re-designing (parts of) this exercise. The difference between the two types of variability is that in the first case, these are errors that are pedagogically anticipated (e.g. confusions of tenses), constituting a form of variability that is desired, versus the second case, where the errors do not constitute pedagogically meaningful ways of intervention to remedy the errors.

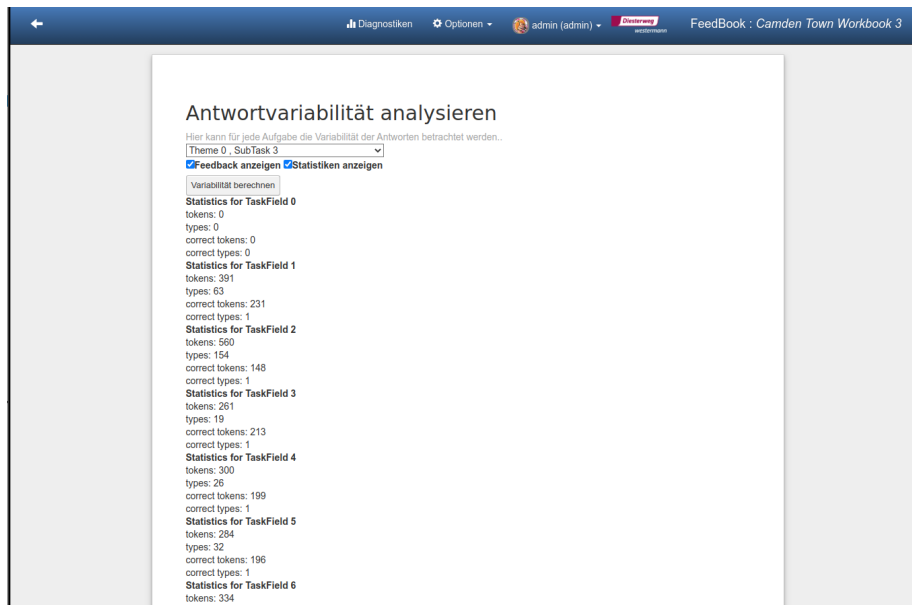


Figure 10.4: Answer variability view: type/token visualization (part 1)

The next follow-up question related to observed variability is how the system deals with it. Concretely, system designers need to be able to interpret which parts of the variability are covered by the system. To this end, Figure 10.5 shows a screenshot of the lower part of the interface displayed in Figure 10.4. This is in essence a flat representation of the information encoded in an interactive way in the item-level performance analytics view (section 9.2).

In the example displayed in Figure 10.5, the target answer is 'is'. The interface shows that this answer has been submitted 52 times to the teachers (token count of submitted answers), and appeared 147 times in the log. This number is higher since when learners re-open an exercise or before they submit answers, their answers are re-evaluated. These instances are counted in the log, resulting in the higher numbers. This is different from the item-level performance analytics view (section 9.2), where these duplicates are filtered out before rendering the data. The most common answer in the discussion is the correct answer, followed by the form 'was', where the system was able to provide specific feedback to learners in the form of the feedback template "This is the simple past. You need to use the simple present here." The next position of the most frequent



Figure 10.5: Answer variability view: aggregated log data visualization (part 2)

answer is shared between the form 'been' and 'will be'. For the latter case, the system was able to provide specific feedback ("This is the will future. You need to use the simple present here."), but in the former case the system could not provide specific feedback. Since a form like 'been' is related to the target answer via lemma identity, this shows an instance of a problem for which a rule that generalizes this pattern should be written so that this type of error in which an inflected form is replaced by the past participle will be recognized and handled for all exercises in which inflected forms are target forms.

10.6 Data-driven Target Answer Extension View

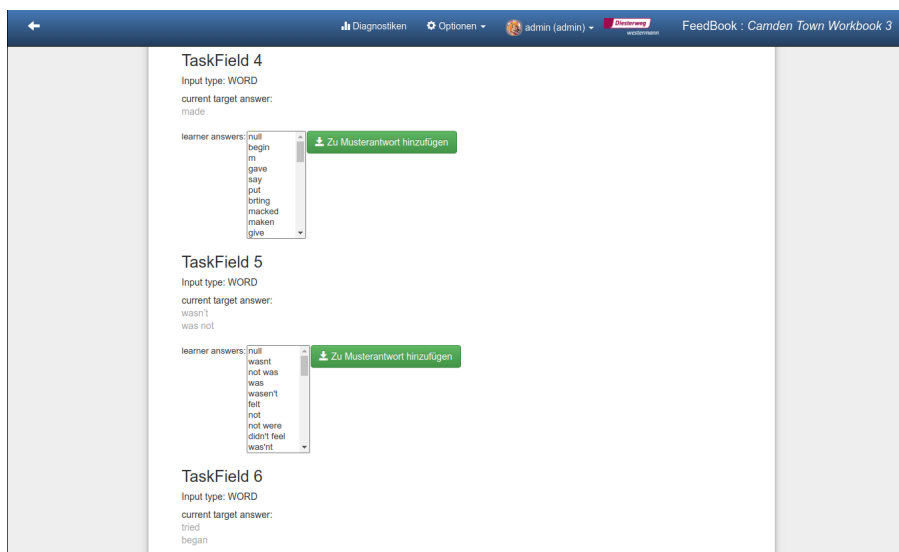


Figure 10.6: Interface for data-driven extension of target answer

When a system is used for a longer period of time, the amount of collected data increases. These data can help identify where the system does not behave in a manner expected by a significant number of the learner population. Problems could be a lack of coverage in the feedback algorithm, exercises that are defined in a problematic or unclear way, or simply errors in the specification of exercises such as additional characters in a target answer. In addition to errors, a continuous usage of a system also allows to look at data from the space of well-formed variability. Typically, for grammar-oriented tasks, FeedBook only contains a few alternative answers. This is reasonable for answers of limited length, such as in fill-in-the-blanks tasks. However, for longer tasks there is more variability in the expression of answers by learners. This most likely includes alternatives that are acceptable, but were not anticipated by the original authors of the respective exercise. Keep in mind that all exercises in FeedBook

were originally designed on paper and the entire evaluation of the correctness of answers was supposed to be done by human teachers who can deal with the exhibited answer variability. The target answers were entered as they were present in the printed workbook, in which they were entered from a top-down perspective and not in a bottom-up approach based on collected data. In order to allow for more variability in the target answers, we implemented an interface which allows material designers to extend target answers with types of learner answers observed in the interaction of learners with a specific item. Figure 10.6 shows an example of this interface. Material designers can see the current target answer and then select one or more target answers from the observed types. Clicking the button extends the target answer with all those selected types that are not yet present in the current target answer.

10.7 Progress View per School Class

Educational administrators, as well as researchers conducting a study, should be able to know about the progress of school classes on the materials. Figure 10.7 shows a user interface in the system where all the mandatory core tasks are listed as rows. The interface has one column for every test school class. Each cell contains a percentage indication for the completion rate of this task for the given school class. The completion rate is visualized in a traffic light system with three sections: the cell is colored in red if less than one third of the school class have completed the respective task, yellow if the completion rate is between one third and two thirds, and green if less than two thirds of the class have completed the specific exercise.

The interface allows to filter the values so that the data is only displayed for either the control or the intervention group for a specific chapter. Furthermore, the interface allows to select for which chapter the data should be displayed.

While the interface was designed to display the progress on a small set of core tasks for the context of the FeedBook study, it can be straightforwardly extended to include any set of tasks, accommodating the specific needs of educational administrators and/or potentially teachers. For example, Figure 10.8 shows an interface from Didi, the follow-up system on FeedBook, that is based on this interface and shows how it can be altered. It visualizes the progress of learners not based on core tasks, but based on linguistic phenomena the tasks are assigned to. Another alternation that was made was to aggregate the data across school classes and render the numbers in a bar chart instead of a table (cf. also Figure 5.19).

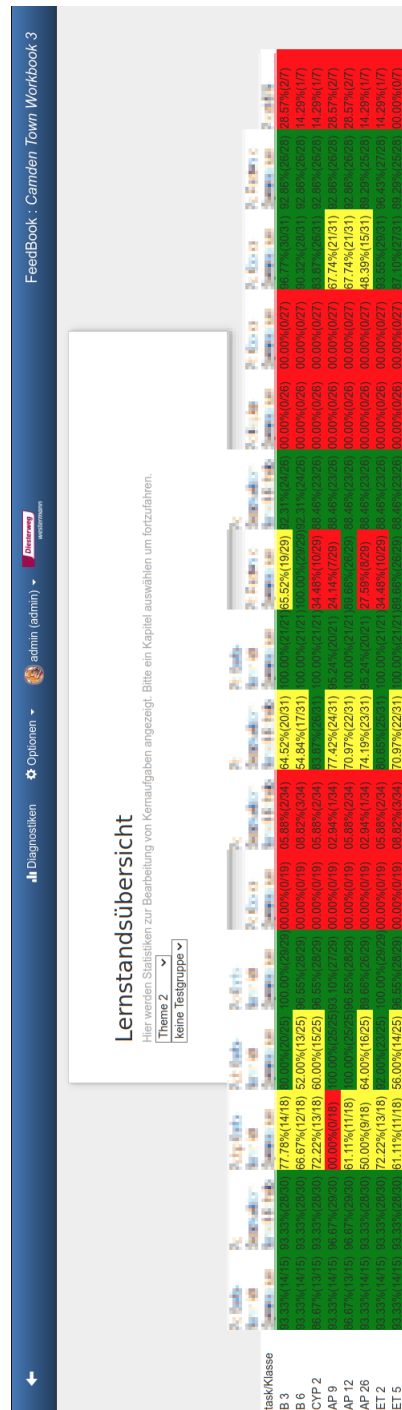


Figure 10.7: Progress view per school class (rotated 90 degrees to accommodate for long format)

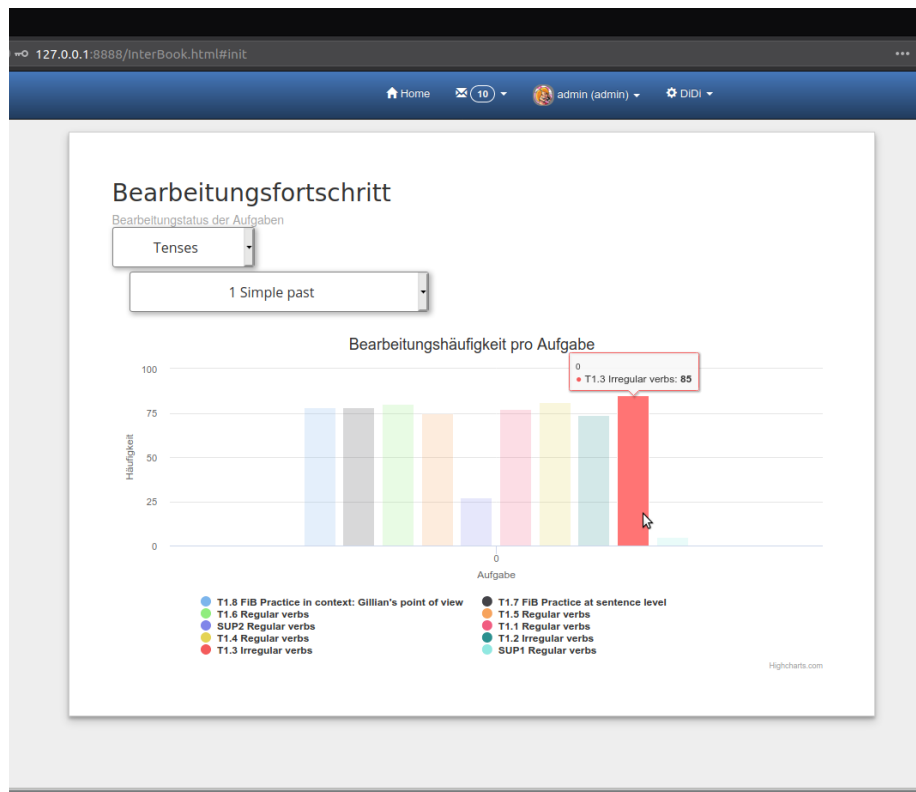


Figure 10.8: Progress view of tasks across school classes

Chapter 11

Learning Analytics for Second Language Acquisition

In this chapter, we analyze and interpret data collected in the context of the FeedBook study (chapter 7). While research in SLA (chapter 2) serves to guide the operationalization of features, the very concrete computation going from an abstract, theoretical description of a construct to concrete numbers, leaves room for interpretation and thus multiple potential operationalizations. Due to fuzzy, non-technical definitions of certain constructs, instead of making decisions beforehand which variant of a feature we test, we simply output the different variants and test them all. This allows us to see the effect of different operationalization for an abstract concept.

For example, time-on-task can be defined as the time learners spend on exercises. Does this include the time learners spend reading the instructions or only the time learner spend filling gaps in an exercise? And should the unit be seconds, minutes, hours, or only an abstract counter for the sessions instead of the duration?

In line with testing different variants of features, we pursue an exploratory analysis in which we aim to unveil relationships between learning process variables and learning products. The key question is: Why did the learning we observed happen? Meurers et al. (2019a) showed that for theme 2 of FeedBook, learners started off with similar pretest scores, but in the post test, they scored significantly higher. The intervention group significantly outperformed the control group. We know from a top-down perspective that the difference between the control and the intervention group is the provision of specific feedback. In this chapter, we however pursue a bottom-up approach in which we output learning process features and explore which differences in the data we see. The idea in this chapter is thus to go from learning processes to learning products by linking observations of interaction with learning outcomes.

We start by discussing the method, procedure and participants sample in section 11.1. The main part of this chapter is the data analysis in section 11.2, in which we explore the effect of feedback, uptake, correct at first try, and time-on-task. In section 11.3, we discuss the results and which conclusions and implications we can draw, with special focus on SLA research and the design of tutoring systems. We conclude with discussing limitations and opportunities for future research in section 11.4.

Why should we care? Before we dive into the analyses, let's take a step back and ask ourselves why we are doing this. There are two reasons for conducting the analyses presented in this chapter. The first reason is that we are interested from a scientific point of view which factors are correlated with eventual learning performance. Very few ICALL systems have been used in ecologically valid environments (cf. section 4.5), and thus it is not clear which learning process features play a role in predicting the change in grammatical knowledge in individuals. The second reason is that the ultimate goal of the development of tutoring systems should be to scale up research prototypes so that they are used by a broader audience and outside of study contexts. In this case, teachers are responsible for guiding learners in the usage of the system. Since there are many learning processes or paths that lead to the same learning product, Learning Analytics functions are needed to inform teachers about the learning process of learners. If we know from these analyses that certain behaviors are associated with exceptional or underwhelming performance, this allows future system designers to implement Learning Analytics functions that present these metrics to teachers. Having such information available allows teachers to diagnose especially students at risk of failing assignments or the course, thus prompting the teacher to intervene and help the struggling students with extra interventions like for example special recap sessions. In order to know which metrics to display to the teacher, we however first need to find out what we can learn from the logs and which metrics are useful and robust predictors of learning gains.

Disclaimer: Specific sub parts of the contents of this chapter have been mentioned or referenced in Meurers et al. (2019a) and are cited accordingly in the running text of the respective sessions.

11.1 Method

Meurers et al. (2019a) reported that for theme 2 of the FeedBook data, there is a significant change in explicit grammatical knowledge from the pretest to the post test. They furthermore reported that learners in the intervention group learned 62% more than learners in the control group. Given that the only difference between the intervention and control group is the provision of specific feedback, they indirectly assumed that specific feedback is beneficial. However, they did not provide a direct proof of any learning process measures that could explain the learning products. In this section, we aim to address this gap by

relating learning processes to the observed and reported learning products. The key question is what it is in the behavior – as observed in the logs – that has an effect on the learning gains. In order to find answers to this question, we compute regression models that extend the models proposed by Meurers et al. (2019a) while keeping the factors found to be significant in the models so that our models are comparable. Concretely, we include a factor *group* that expresses whether learners are in the intervention or control group, and a feature *pretest score* that expresses the number of correctly answered items in the pretest, thus expressing prior knowledge of the grammatical structures. In the following, we conduct an exploratory analysis in which we test and explore various learning process features that in the literature have been proposed as relevant factors in determining learning gains.

First, we motivate the focus on theme 2 in section 11.1.1 and explain in more detail how the scores were computed (section 11.1.2). Then we turn towards the individual experiments testing groups of learning process features. We start by exploring a broad range of feedback frequency features in section 11.2.1. Based on the findings there, we turn towards answers submitted correct at first try (section 11.2.5) before exploring different features related to uptake in section 11.2.3. Finally, we present analyses related to the time on task in section 11.2.6. We conclude with a discussion in section 11.3.

11.1.1 Focus on Theme 2

For the subsequent analyses, we focus on theme 2 of the work book. We thereby link up with and extend the work by Meurers et al. (2019a). For theme 1, the system was still under development and contained several severe bugs that compromised the experimental design. On the one hand, the blacklist for filtering constructs was incomplete and certain specific feedback messages for this chapter could pass through and got displayed. On the other hand, there was a bug in the implementation of the check-all button and the check of all answers when submitting a task to the teacher. This led to specific feedback being computed and available for inspection by learners by clicking on the question mark. This bug compromised the experimental design in that both the intervention and control group received the specific feedback. In theme 2, the bugs had been fixed and everything worked as expected. As described in Meurers et al. (2019a), learners in the intervention group ($n = 104$) learned significantly more (mean change score 7.82) than the learners in the control group ($n = 101$, mean change score 4.81). For theme 3, the system worked as expected as well, and even more information was logged, for example instances in which learners actively clicked on question mark symbols to request specific feedback. However, when analyzing the learning gains, we noticed that there were no significant learning gains for the intervention or control group, and also no significant differences between the pretest and post test. For the intervention group ($n = 93$) in theme 3, we observed a change score of 2.8, for the control group ($n = 96$) a change score of 2.19. A Welsh two-sample t-test revealed that there are no significant differences ($p = 0.3492$). In theme 3, the topic were passive constructions. From

the change scores, and from informal conversations with students and teachers, we have to assume that the passive is too difficult at this stage of development. For theme 4, the system worked properly as well, but, as Table 7.4 shows, there are only very limited amounts of data available. Due to the proximity of the chapter to the summer break at the end of the school year, we lack the post test results of four school classes. Since one school class skipped theme 3 altogether, we lack the pretest for one other school class.

11.1.2 Answer Comparisons beyond String Matches

We exported all answer types learners submitted in the pretest, post test, and delayed post test. A member of the project who was involved in the creation of the tasks labeled all answers that had not been marked as correct previously as either *false* (default), *correct alternative*, or *nonsense answer*. An answer marked as *false* is a variant that should (and was) not accepted by the system. In contrast, a *correct alternative* is a learner answer that is not identical to one of the target answers associated with a task field after string normalization. We counted this as a correct answer, even though the system would not have recognized it. These data can be used in future work to improve the system’s paraphrase handling. Examples for correct alternatives are answers with casing differences or answers with minor spelling differences, such as “woul be” instead of “would be”. The last category, *nonsense answer*, was assigned to answers that were off topic, such as copied web pages, random letter strings, names, punctuation, whitespace, or swearwords. Examples for *nonsense answers* are “Keine Ahnung”[sic], “Birne” or “?? i don’t know”[sic]. Examples of false answers are “herself” (correct answer: “yourselves”) or “wellest” (correct answer: “best”). Table 11.1 lists the number of answer types (not tokens) for each of the three categories. The numbers only reflect the types from part (a) and (b) of the pretest (cf. section 7.1.2), since in part (c) with sentence-length input the variability was very high, leading to an exclusion of the respective task in the subsequent analyses (cf. Meurers et al. (2019a)).

answer type	number of answer types in this category
nonsense answer	766
correct alternative	1.182
false	6.076

Table 11.1: Answer types by category

11.1.3 Participants

For the experiments reported here, we use the data of the exactly same sample of participants reported by Meurers et al. (2019a). For this reason, the information about the participants is taken from this article and will be reported in the following. This specific sample constitutes a sub sample of the entire set of participants described in section 7.1.1.

We report analyses for ten 7th grade school classes with 205 students. 104 participants were in experimental group A (intervention group for themes 2 and 4), and 101 in group B (intervention group for themes 1 and 3). The algorithm randomly assigned half of the students in each school class to group A and the other half to group B. The sample consists of 116 male students, 84 female students, and five students who did not report their gender. The group assignment was conducted randomly before the beginning of the first interaction with the system, with the assignment balanced within each school class. The average age of the learners during the experiment was 13 years. From the original, complete setup, two teachers dropped out due to work overload, in one school class the parents opted to stop using the system, and in one class problems with the internet connection during the test prevented valid test results since the learners needed to change the room and partly worked together on the solutions. Also, we excluded 33 students from the analyses who did not consent to a usage of their data. The schools included a mix of public and private schools, as well as several school classes with Content and Language Integrated classes. In two school classes, the students used tablet computers for the entire year.

11.1.4 Procedure

In section 11.2, we describe four experiments that test how features extracted from the interaction of learners with the FeedBook can contribute to predicting how learners develop in terms of explicitly tested grammatical knowledge. Since before the start of the experiments it was not clear which features and which feature operationalizations play a significant role, we conducted an exploratory analysis. Each of the four experiments tests the effect of different variants of related features by systematically following a shared procedure.

The regression models reported here extend the linear regression models proposed by Meurers et al. (2019a). To this end, we controlled for and included the two significant features reported by them. The feature *pretest_correct* expresses the number of correctly answered items in the pretest, thus acting as a proxy for previous knowledge in the specific grammar topics tested in this chapter. The other feature included from Meurers et al. (2019a) is *group*. This factor has two levels: *intervention* and *control*.

As the first step in any experiment, we tested for potential *main effects* by computing one model for each feature that includes the pretest score and group plus the feature to test. As the second step, we tested for *moderation effects* by testing whether any of the features interacts with the group, pretest scores, or the number of attempted tasks. For the group interaction, we tested whether there are interaction patterns and effects that differ between the intervention and control group. Consequently, this allows to see if there is a difference resulting from the difference in treatment of learners. If a feature interacts with the pretest score, this means that learners with higher pretest score interact differently, and that this difference is reflected in the change score. Since the conditions between participants were not equal in that every student could in-

dependently work on tasks of their choice, we also tested whether there was an interaction with the number of tasks. As the third step, we tested for *mediation effects* by computing models that test models that assume the pretest score or group as independent variables and use the interaction features as (potential) mediators. We report models for all significant paths of the mediation analyses. We did not fit models that compute different slopes per school class/teacher, given that Meurers et al. (2019a, p. 21) showed that the teacher has only a very small effect on the change score and that the effects are stable across school classes.

We summarize and interpret the results in a combined discussion section in section 11.3. In section 11.4, we provide a combined discussion of limitations of the approach.

11.1.5 Feature Operationalization

11.2 Results

11.2.1 Feedback

Feature Operationalization

Table 11.2 lists the feedback features we used in the experiments. The features are organized in five groups, separated by horizontal lines. Note that the feature groups are not exclusive - some features are aggregates of and correlated with other features. For example, *manualFeedback* is equal to the sum of all other features that start with the prefix *manual*. Provided that feedback was computed and displayed in different ways, we operationalized feedback in these different ways in order to test if any of the 14 features help predicting the change in explicit grammatical knowledge from pretest to post test. The feature set represents frequencies of feedback computations and includes both displayed (*seen*) and blocked (*blocked*) feedback instances, as well as feedback triggered only for a single item (*manual*) or for all items in a task (*auto*), i.e. when re-entering, before submitting a task, or when clicking the check-all button. In order to scale the features, we z-standardized and log-transformed the individual features.

We tested for potential main effects of the features by extending the significant independent variables *group* and *pretest_correct* from the regression models by Meurers et al. (2019a) with the individual feedback features.

Results

Testing for Main Effects Table 11.3 shows a significant main effect of the feature *manualCorrectSeen* ($p = 0.0332$). The model on the left uses the control group as reference category, the model on the right uses the intervention group as reference. There is a difference in terms of intercept between the control (5.210) and intervention (7.706) group. The feature used in the reported equations

feature name	description
manualFeedback	number of feedback messages computed (displayed and blocked) for a single item
automaticFeedback	number of feedback messages computed (displayed and blocked) for all items in a task
correctFeedback	number of check marks displayed
falseFeedback	number of times in which an incorrect answer was submitted to the feedback algorithm
seenFeedback	number of times feedback passed the blacklist filter
blockedfeedback	number of times feedback was blocked by the blacklist filter
manualCorrectSeen	number of times feedback requested for a single item for a correct answer and displayed
manualCorrectBlocked	number of times feedback requested for a single item for a correct answer, but blocked via blacklist
manualFalseSeen	number of times feedback requested for a single item for an incorrect answer and displayed
manualFalseBlocked	number of times feedback requested for a single item for an incorrect answer, but blocked via blacklist
autoCorrectSeen	number of check marks computed and displayed for all items in a task
autoCorrectBlocked	number of check marks computed for all items in a task, but blocked via blacklist
autoFalseSeen	number of feedback messages computed and displayed for all items in a task
autoFalseBlocked	number of feedback messages computed for all items in a task, but blocked via blacklist

Table 11.2: Feedback features considered in regression analyses

expresses the number of times positive feedback in the form of a check mark was computed and displayed (i.e. not blocked) for a single learner answer (not all learner answers jointly). In other words, this feature expresses the number of times a learner was able to reach a correct solution. For the thirteen other features, we did not find a significant main effect.

	<i>Dependent variable:</i>	
	changeScore	
	(1)	(2)
manualCorrectSeen	0.737** (0.344)	0.737** (0.344)
pretest_correct	-3.352*** (0.295)	-3.352*** (0.295)
groupintervention	2.496*** (0.584)	
groupcontrol		-2.496*** (0.584)
Constant	5.210*** (0.421)	7.706*** (0.414)
Observations	205	205
R ²	0.437	0.437
Adjusted R ²	0.429	0.429
Residual Std. Error (df = 201)	4.171	4.171
F Statistic (df = 3; 201)	52.050***	52.050***
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01	

Table 11.3: Regression model: significant main effect of correct feedback not blocked and displayed for one item

Testing for Moderation Effects Next, we tested for potential interactions with the feature *group*, given that we know from a top-down perspective the values of the features are partly a result of the group assignment that leads to a display or blocking of certain feedback messages. We found a marginally significant interaction between *group* and *autoCorrectSeen*. By calculating the simple effects, we see that there is only a significant effect of 1.3007 ($p = 0.03762$) for the control group (the reference group in the left model in Table 11.4), but not for the intervention group (-0.385, $p=0.652$, right model of Table 11.4), obtained

via re-leveling). This effect is, however not strong, given that it disappears when we include the pretest score into the equation. This can potentially be explained in the following way: a bug was present in the system until November 11, 2018 that allowed the control group to receive specific feedback by requesting feedback for all items and later clicking on the question marks for individual items to see the feedback popup. As can be seen in Table 7.4, there are three school classes for which the pretest of theme 2 was conducted shortly before this date (November 9 and 10, 2018). We furthermore found no interaction between the number of tasks attempted and any of the feedback features.

	<i>Dependent variable:</i>	
	changeScore	
	(1)	(2)
autoCorrectSeen	1.301** (0.622)	-0.385 (0.652)
groupintervention	2.307*** (0.813)	
autoCorrectSeen:groupintervention	-1.686* (0.901)	
groupcontrol		-2.307*** (0.813)
autoCorrectSeen:groupcontrol		1.686* (0.901)
Constant	5.395*** (0.595)	7.702*** (0.554)
Observations	205	205
R ²	0.096	0.096
Adjusted R ²	0.082	0.082
Residual Std. Error (df = 201)	5.287	5.287
F Statistic (df = 3; 201)	7.095***	7.095***
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01	

Table 11.4: Regression model: estimating simple effects of group and correct feedback computed for all items in a task

We tested for potential interactions between the pretest score and the feedback features. We found an interaction between *manualFalseBlocked* (-1.2346,

$p=0.0282$, feedback requested and blocked for a single item at a time) and *autoCorrectSeen* (-0.8087 , $p=0.0187$, feedback requested and not blocked for all items at a time). Table 11.5 shows the regression models.

	<i>Dependent variable:</i>	
	changeScore	
	(1)	(2)
manualFalseBlocked	0.468 (0.517)	
autoCorrectSeen		0.533 (0.354)
pretest_correct	-3.625*** (0.332)	-3.586*** (0.317)
groupintervention	2.631*** (0.596)	2.585*** (0.587)
manualFalseBlocked_log:pretest_correct	-1.235** (0.558)	
autoCorrectSeen_log:pretest_correct		-0.809** (0.341)
Constant	5.151*** (0.444)	5.238*** (0.444)
Observations	205	205
R ²	0.439	0.448
Adjusted R ²	0.428	0.437
Residual Std. Error (df = 200)	4.176	4.141
F Statistic (df = 4; 200)	39.089***	40.604***
<i>Note:</i>	* $p<0.1$; ** $p<0.05$; *** $p<0.01$	

Table 11.5: Regression models: interaction of pretest score with feedback requesting features

Testing for Mediation Effects We furthermore tested potential mediation effects of features with *group* and *pretest_score* as dependent variables. We found effects of group on the features that were blocked. However, given that in mediation analyses it is necessary that the mediator can be computed equally

for both groups, we can not draw conclusions from these analyses since the mediator can not be computed equally for the two groups, but only for the control group, for which the feedback was blocked.

11.2.2 From Learning Product to Learning Process

Having established a significant main effect of attempts that yielded a correct answer, we decided to investigate this effect in more detail. Given that the learning product in the form of a correct answer can be reached with different learning processes, or different learning paths, we decided to zoom in and look at different learning process measures that lead to correct answers. Fundamentally, there are three ways in which a correct answer could be reached in the FeedBook. In case (a), learners submit a correct answer that is **correct at first try**. In this case, learners demonstrate previous knowledge. While it may also be the case that learners are guessing correct at the first attempt, we assume that the majority of answer submitted as correct at first try can be attributed to previous knowledge of the learners. In case (b), learners first submit one or more incorrect answers, but, after having seen feedback, show **uptake with specific feedback**. In this case, the specific, corrective feedback helped learners reach a correct solution. There is another case, (c), that arose due to the experimental design. For the control group in each chapter, the specific feedback was blocked. Learners did, however, in tasks which not exclusively focused on the constructs covered in the pretest and post test, still receive the green check marks indicating a correct answer. The specific feedback was, however, not displayed for learners in the control group. Due to the absence of feedback and a green check mark, learners were in a position to understand that their answer was not correct. This could push them to correct their answer, leading to **uptake without specific feedback**. In this case, and in contrast to the case of uptake with feedback, learners can only focus on the form and did not receive an explanation of why or how their answer was incorrect. Splitting up the learning product of a correct answer into these three learning processes, we investigated which of them were responsible for the learning product. In section 11.2.3, we investigate the role of uptake, and in section 11.2.5, we continue with an in-depth investigation of the phenomenon of answer submitted correctly at first try. Note that these learning process features do not exclude each other, but rather are compatible with each other.

11.2.3 Uptake

Feature Operationalization

We operationalized the concept of uptake discussed in section 2.3 as two features listed in Table 11.6. The raw features were computed as frequency counts of the exercise interaction log (cf. section 7.2).

The first feature, *uptakeSeenFeedback*, expresses an interaction sequence in which learners submitted an answer, received feedback by the system, and man-

feature	description
uptakeSeenFeedback	uptake (correction of learner answer) after feedback displayed
uptakeBlockedFeedback	uptake after feedback not displayed due to blacklist filter

Table 11.6: Uptake feature operationalization

aged to correct their answer in the next step. It includes both instances of specific and default feedback, but due to the study design (section 7.1), specific feedback for all constructs introduced in this chapter and covered in the pretest and post test is only displayed to the intervention group. This feature thus implicitly operationalizes aspects of the study design. The second feature, *uptakeBlockedFeedback*, describes an interaction sequence in which a learner submitted an answer, the system computed feedback, and the feedback was blocked since the construct was on the blacklist of the learner. However, due to the absence of a green check mark, the learner was still able to recognize that the answer was incorrect and able to correct it.

This minimal difference in the operationalization allows to see whether it is (a) the attention to the specific feedback or (b) only the attention to the form without the specific feedback that leads to corrections and in the end an influence on the change score.

In order to normalize the feature values, we z-standardized both predictor values. By inspecting the distributions of the features (top plot in Figure 11.1), we recognized that the distribution is left skewed. A skewness test (from the library *e1071*) formally confirmed that the predictor variable is highly skewed (+1.19). For this reason, we log-transformed the predictor values for these experiments (bottom plot in Figure 11.1).

Results

Testing for Main Effects Table 11.7 shows a regression model testing a potential main effect of the feature *uptakeBlockedFeedback*, in addition to the predictors *group* and *pretest_correct* established by Meurers et al. (2019a). It shows that there is no main effect for the uptake feature with blocked feedback, with the p-value 0.228 far from significance. For uptake based on seen feedback, the model in Table 11.7 shows that there is a significant main effect ($p = 0.0412$).

Testing for Moderation Effects As a first step, we tested whether there is an interaction effect of one of the uptake features with the independent variable *group*. We found no significant interaction effect with *group* for either of the uptake features. Table 11.8 lists the regression models that show that for neither uptake with displayed nor for uptake with blocked feedback there is a significant interaction with *group*.

We also tested whether there is an interaction between the uptake features

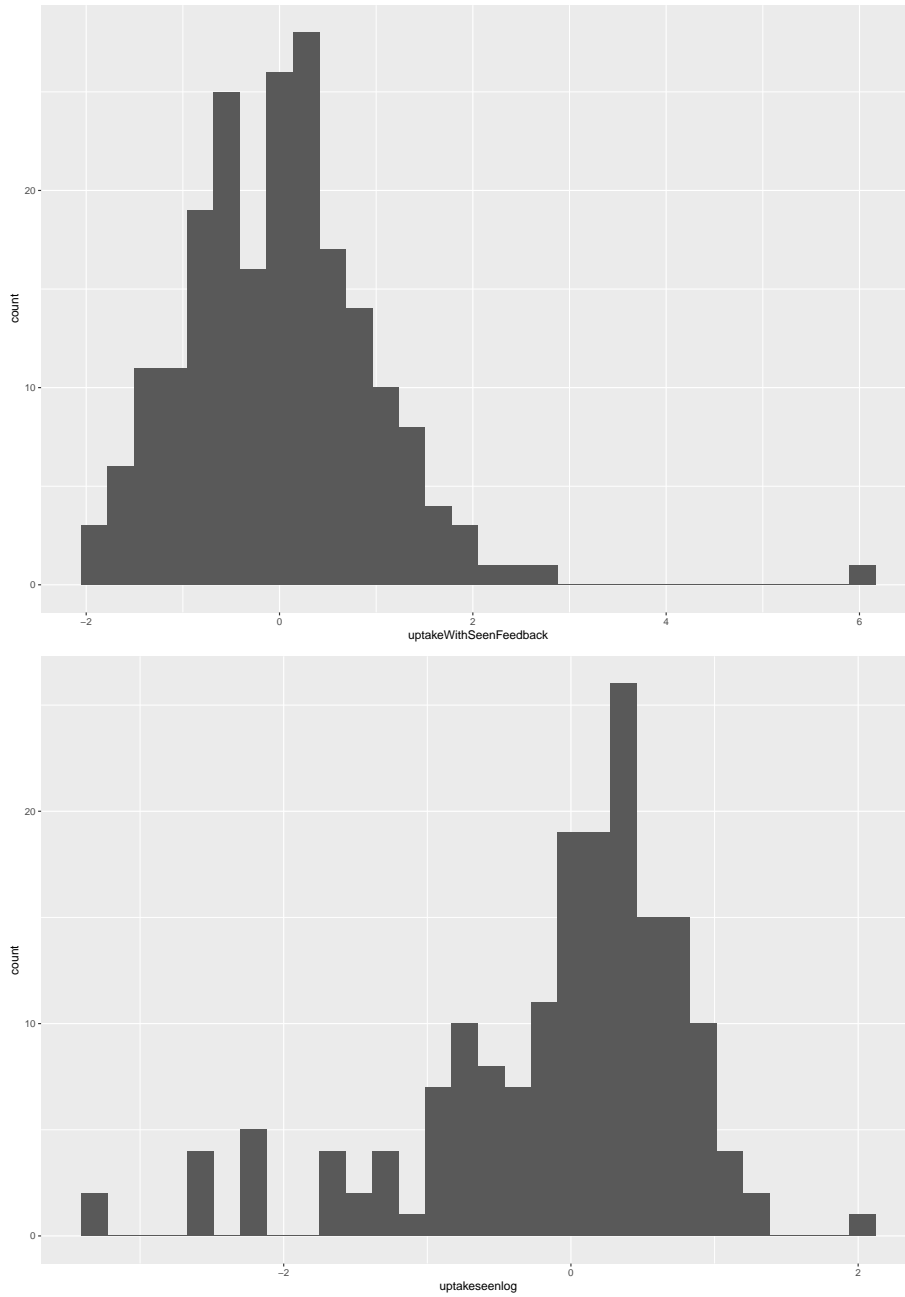


Figure 11.1: Independent variable uptake after displayed feedback, normal z-standardized values (top) and log-transformed, z-standardized values (bottom)

	<i>Dependent variable:</i>	
	changeScore	
	(1)	(2)
uptakeSeenFeedback	0.678** (0.330)	
uptakeBlockedFeedback		0.355 (0.294)
groupintervention	2.641*** (0.589)	2.631*** (0.598)
pretest_correct	-3.376*** (0.297)	-3.321*** (0.298)
Constant	5.148*** (0.418)	5.149*** (0.425)
Observations	205	205
R ²	0.436	0.428
Adjusted R ²	0.428	0.420
Residual Std. Error (df = 201)	4.175	4.203
F Statistic (df = 3; 201)	51.831***	50.235***
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01	

Table 11.7: Regression model: uptake based on unseen feedback not significant, uptake based on feedback significant

	<i>Dependent variable:</i>	
	changeScore	
	(1)	(2)
uptakeSeenFeedback	0.682 (0.478)	
uptakeBlockedFeedback		0.161 (0.401)
groupintervention	2.639*** (0.607)	2.806*** (0.647)
pretest_correct	-3.376*** (0.298)	-3.330*** (0.298)
uptakeSeenFeedback:groupintervention	-0.008 (0.654)	
uptakeBlockedFeedback:groupintervention		0.415 (0.584)
Constant	5.148*** (0.421)	5.105*** (0.430)
Observations	205	205
R ²	0.436	0.430
Adjusted R ²	0.425	0.419
Residual Std. Error (df = 200)	4.185	4.208
F Statistic (df = 4; 200)	38.680***	37.710***
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01	

Table 11.8: Regression model: no interaction between group and uptake features

and the number of attempted tasks. Table 11.9 shows a regression model for both uptake features and their interaction with the number of tasks. The models show that there is no direct interaction of either of the uptake features with the number of attempted tasks. Furthermore, we tested whether there is an interaction between the pretest score and any of the uptake features. We did not find an interaction between the pretest score and the uptake features.

	<i>Dependent variable:</i>	
	changeScore	
	(1)	(2)
numTasksAttempted	-0.642* (0.328)	-0.562* (0.336)
uptakeSeenFeedback	0.991*** (0.368)	
uptakeBlockedFeedback		0.538 (0.338)
groupintervention	2.754*** (0.587)	2.707*** (0.604)
pretest_correct	-3.354*** (0.295)	-3.280*** (0.298)
numTasksAttempted:uptakeSeenFeedback	-0.256 (0.324)	
numTasksAttempted:uptakeBlockedFeedback		-0.214 (0.317)
Constant	5.264*** (0.434)	5.285*** (0.448)
Observations	205	205
R ²	0.449	0.437
Adjusted R ²	0.435	0.423
Residual Std. Error (df = 199)	4.147	4.193
F Statistic (df = 5; 199)	32.462***	30.883***
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01	

Table 11.9: Regression model: no direct interaction between uptake features and number of tasks attempted

Testing for Mediation Effects We tested whether *group* or *pretest_score* are mediated through one of the uptake features. For *uptakeBlockedFeedback*, we observed that when fitting a model to estimate the effect on the mediator, there is a significant effect of *group* and a marginally significant effect ($p = 0.0614$) for the pretest score (Table 11.10). However, as Table 11.7 shows, the effect disappears when estimating the indirect effect. This is explainable also with regard to the fact that one of the pre-conditions of mediation analyses is that the mediator needs to be computable in the same way for different experimental group (such as time-on-task), which is not the case for uptake based on blocked feedback. In contrast, uptake based on feedback can be computed for both groups since some feedback was displayed for both groups.

	<i>Dependent variable:</i>
	uptakeBlockedFeedback
groupintervention	-0.345** (0.141)
pretest_correct	0.133 . (0.071)
Constant	-0.240** (0.100)
Observations	205
R ²	0.048
Adjusted R ²	0.039
Residual Std. Error	1.007 (df = 202)
F Statistic	5.144*** (df = 2; 202)
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01

Table 11.10: Regression model: significant effect of group, marginally significant effect of pretest score on uptake based on blocked feedback

For *uptakeSeenFeedback*, we see that there is no effect of *group* on the (potential) mediator, but a significant effect of pretest score (see Table 11.11).

When computing the indirect effect by including the mediator as a main effect along the independent variables, we see that the significant effect of uptake based on seen feedback persists.

Having already established a main effect of uptake based on displayed feedback (see Table 11.7), we computed a causal mediation analysis model based on 1,000 iterations over the data, with *pretest_score* as the independent variable and uptake based on seen feedback as the mediator. The model displayed in Table 11.13 formally asserts that uptake based on seen feedback is a partial mediator of *pretest_score*. The model shows a significant average causal mediation

	<i>Dependent variable:</i>
	uptakeSeenFeedback
groupintervention	-0.196 (0.125)
pretest_correct	0.150** (0.062)
Constant	-0.124 (0.089)
Observations	205
R ²	0.042
Adjusted R ²	0.033
Residual Std. Error	0.890 (df = 202)
F Statistic	4.434** (df = 2; 202)

Note: *p<0.1; **p<0.05; ***p<0.01

Table 11.11: Regression model: significant effect of pretest score on uptake based on displayed feedback

effect of 0.1018 ($p=0.03$), an average direct effect of -3.3757 ($p < 0.0001$) and a total effect of -3.2739 ($p < 0.0001$). While the effect of pretest score is very strong, the mediation analysis shows that uptake based on seen feedback is a factor to consider that can help explain the variance in the dependent variable in a more detailed manner. We visualize the causal mediation model in Figure 11.2. Note that the validity of this analysis depends on whether *pretest_score* is accepted as the independent variable in this case. It is possible to argue that the independent variable in a mediation analysis needs to be an experimental condition assigned before the mediators were measured and that additionally needs to have been assigned randomly, such as the *group* variable.

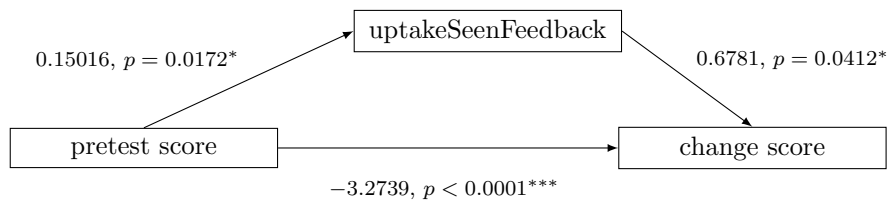


Figure 11.2: Mediation analysis of uptake based on seen feedback and pretest score on the change score

In order to make the observed relation between the features more tangible,

<i>Dependent variable:</i>	
changeScore	
uptakeSeenFeedback	0.678** (0.330)
groupintervention	2.641*** (0.589)
pretest_correct	-3.376*** (0.297)
Constant	5.148*** (0.418)
Observations	205
R ²	0.436
Adjusted R ²	0.428
Residual Std. Error	4.175 (df = 201)
F Statistic	51.831*** (df = 3; 201)

Note: *p<0.1; **p<0.05; ***p<0.01

Table 11.12: Regression model: significant effect of uptake based on seen feedback

Causal Mediation Analysis

Nonparametric Bootstrap Confidence Intervals with the Percentile Method

	Estimate	95% CI Lower	95% CI Upper	p-value
ACME	0.1018	0.0048	0.23	0.03 *
ADE	-3.3757	-4.0156	-2.75	<0.0000000000000002 ***
Total Effect	-3.2739	-3.8917	-2.67	<0.0000000000000002 ***
Prop. Mediated	-0.0311	-0.0764	0.00	0.03 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Sample Size Used: 205

Simulations: 1000

Table 11.13: Causal mediation analysis results of uptake based on seen feedback (mediator) and pretest score (independent variable)

we plotted the three dimension (uptake based on seen feedback, pretest score, and change score) in three-dimensional plots (Figure 11.3). The plots suggest non-linearities in the relationship that are not captured by individual regression models, but can be approximated with an ensemble of models, as it is the case in mediation analyses. We return to this in section 11.2.4.

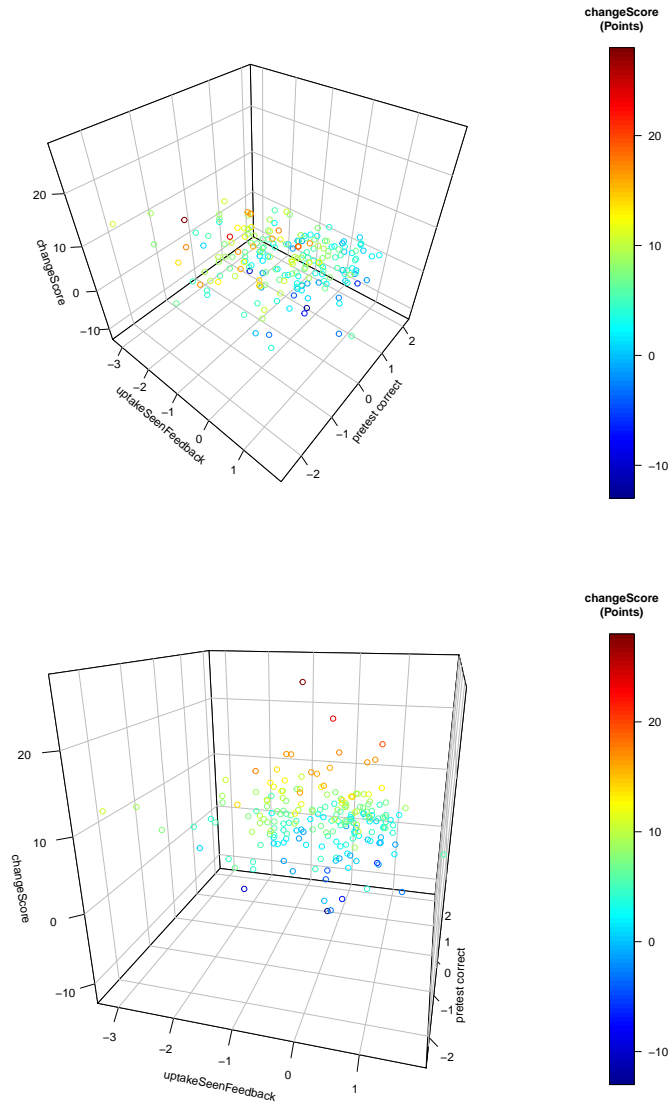


Figure 11.3: Visualization of uptake after displayed feedback (x-axis), pretest score (y-axis), and change score (z-axis)

Keeping the methodological limitation of the choice of independent variable in the mediation analysis in mind, we conducted further analyses. As the second step towards better understanding the significant effect of the uptake feature, we conducted a causal mediation analysis of the uptake feature in relation to the number of tasks. Due to the fact that in the FeedBook study every student could independently work on tasks in that we did not restrict the tasks they could work on (cf. section 7.1), the experimental conditions were not equal across students, leading us to hypothesize that there could be a mediation through the number of tasks. We computed supplementary models in addition to the main effect model (Table 11.7), which we will report in the following. Table 11.14 tests the effect of the independent variable *uptake with seen feedback* on the mediator number of tasks this learner attempted. There is a highly significant effect ($p < 0.0001$) of uptake on the number of attempted tasks.

	<i>Dependent variable:</i>
	numTasksAttempted
uptakeSeenFeedback	0.505*** (0.071)
groupintervention	0.169 (0.126)
pretest_correct	0.028 (0.064)
Constant	0.027 (0.090)
Observations	205
R ²	0.211
Adjusted R ²	0.200
Residual Std. Error	0.895 (df = 201)
F Statistic	17.967*** (df = 3; 201)
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01

Table 11.14: Regression model: effect of independent variable on the mediator

Table 11.15 shows a regression model testing the effect of the mediator on dependent variable *changeScore*. Since both the independent variable *uptake-SeenFeedback* and the mediator *numTasksAttempted* are included, this regression model displays the total effect. The fact that there is a significant effect ($p = 0.0412$) of the number of tasks attempted in conjunction with the *uptake-SeenFeedback* is the last piece of the puzzle regarding the mediation analysis and confirms that there is indeed a mediation through the number of tasks.

	<i>Dependent variable:</i>
	changeScore
numTasksAttempted	-0.661** (0.327)
uptakeSeenFeedback	1.012*** (0.367)
groupintervention	2.753*** (0.587)
pretest_correct	-3.357*** (0.295)
Constant	5.166*** (0.415)
Observations	205
R ²	0.447
Adjusted R ²	0.436
Residual Std. Error	4.143 (df = 200)
F Statistic	40.497*** (df = 4; 200)
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01

Table 11.15: Regression model: indirect effect of mediator on dependent variable

In order to get a formal assertion of the mediation effect, we combined the previous models into a causal mediation summary. Table 11.16 shows the output of a causal mediation analysis summary provided by the *mediation* package in R (cf. Tingley et al. (2014)). In a process of 1,000 resamples of the data, the package estimated the confidence intervals for the individual effect. The results show a significant ($p = 0.048$) average causal mediation effect (ACME) of -0.3339 , i.e. the complete indirect effect. Furthermore, the model shows a significant ($p = 0.004$) average direct effect (ADE) of 1.0120 . The total effect with the value of 0.6781 is significant as well ($p = 0.028$). The model shows a bootstrapped unstandardized indirect effect of -0.4924 , which is marginally significant ($p = 0.076$) given the 95% confidence intervals from -3.6588 to 0.08 . We visualize the results of the mediation analysis in Figure 11.4. The ACME of -0.3339 is represented as its components in the form of the effect of the independent variable on the mediator (0.505) and the indirect mediated effect of the independent variable on the dependent variable (-0.062). The significant direct effect of 0.678 , as calculated in Table 11.7 and confirmed in the causal mediation analysis in table 11.16, is visualized at the bottom of the visualization.

Causal Mediation Analysis

Nonparametric Bootstrap Confidence Intervals with the Percentile Method

	Estimate	95% CI Lower	95% CI Upper	p-value	
ACME	-0.3339	-0.7668	0.00	0.048	*
ADE	1.0120	0.2616	1.74	0.004	**
Total Effect	0.6781	0.0544	1.31	0.028	*
Prop. Mediated	-0.4924	-3.6588	0.08	0.076	.

 Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Sample Size Used: 205

Simulations: 1000

Table 11.16: Causal mediation analysis results of uptake based on 1.000 iterations

11.2.4 Exploring the Relationship between Predictors Further

Visualization In order to gain insights into the, at first glance counter-intuitive, negative weight of the number of tasks in the regression model (Table 11.15) visualized in the mediation structure in Figure 11.4, we visualized the involved data in a 3-dimensional scatter plot in order to see the relevant dimensions together. We generated the plots displayed in Figure 11.5 with the

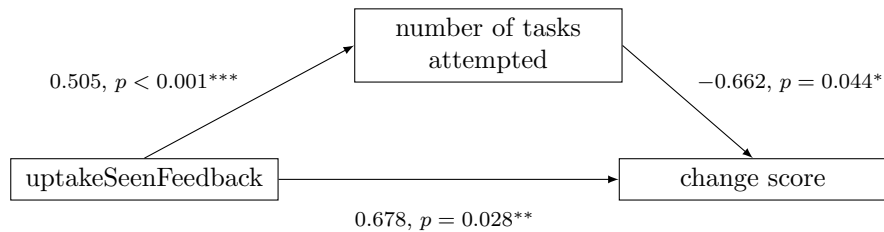


Figure 11.4: Mediation analysis of uptake based on seen feedback and number of attempted tasks on the change score

plot3D package (Soetaert (2014), version 1.3) in R. The graphic shows the relation between uptake after displayed feedback (x-axis), number of attempted tasks (y-axis), and the change score (z-axis). The change score is visualized as values on the z-axis as well as on a color spectrum from blue (low) over green (medium) to red (high). The two plots visualize the same data, just with the perspective rotated.

The plots show the deepest blue points for those learners who attempted the least number of tasks. With respect to uptake, the lower plot shows that learners with much less than the average number of uptake observations (left side), show a lower change score (green/light blue) than average learners (middle, orange and red values). The learners with the highest number of uptake do not perform better than learners with less uptake. Quite the contrary, the plots show that there is a “happy medium” in that the highest chance scores are achieved with average number of uptake observations on a larger number of tasks. We hypothesized that the fact that the number of tasks coefficient in the mediation analysis is negative could thus be explained by the fact that there is a non-linear relationship of uptake and change score, and the coefficient of number of tasks attempted needs to correct for this non-linearity. For this reason, in the next section, we report exploratory non-linear models

Fitting Non-Linear Models In order to test whether the data can be fitted more accurately with non-linear model, and whether we can gain insights about the nature of the data from this, we computed non-linear alternatives to the linear models reported. As a first step, we fitted a polynomial regression model (Table 11.17). An Anova revealed that this model is not significantly better than its simpler model alternative ($p = 0.4699$).

As a next step, we fitted generalized additive models on the data. Table 11.18 shows the model summary of one example model. The generalized additive model shows that for uptake based on seen feedback, the optimal degree of the polynomials is actually one - i.e. a linear function. For pretest, it however estimates an optimal fit for a 6th-order polynomial. We visualize the fitted parameter in Figure 11.6. The plot reveals a non-linear structure with a local maximum in the center. While such a complex model can fit the data more

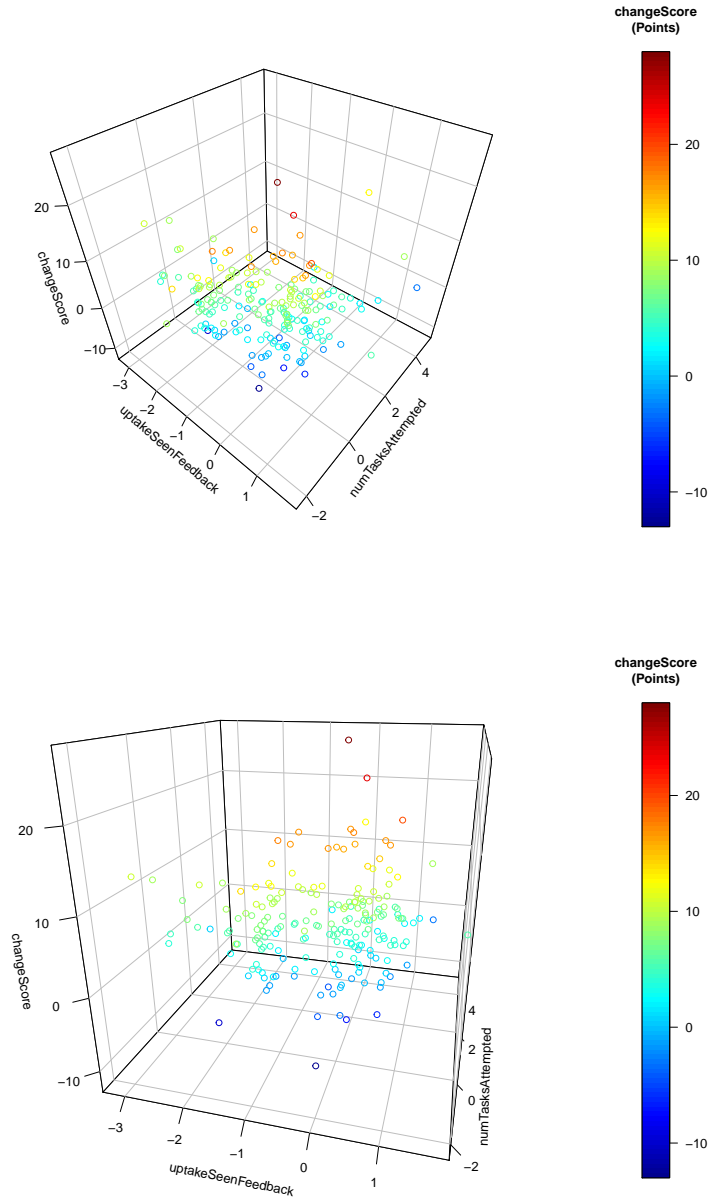


Figure 11.5: Visualization of uptake after displayed feedback (x-axis), number of attempted tasks (y-axis), and change score (z-axis)

	<i>Dependent variable:</i>
	changeScore
uptakeSeenFeedback	0.469 (0.439)
I(uptakeSeenFeedback ²)	-0.178 (0.246)
groupintervention	2.623*** (0.590)
pretest_correct	-3.381*** (0.298)
Constant	5.264*** (0.448)
Observations	205
R ²	0.438
Adjusted R ²	0.426
Residual Std. Error	4.180 (df = 200)
F Statistic	38.912*** (df = 4; 200)
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01

Table 11.17: Polynomial regression model with uptake based on seen feedback

	Model 1
(Intercept)	5.00*** (0.41)
groupintervention	2.63*** (0.58)
EDF: s(uptakeSeenFeedback)	1.00** (1.00)
EDF: s(pretest_correct)	6.14*** (7.24)
EDF: s(numTasksAttempted)	1.00* (1.00)
AIC	1166.12
BIC	1203.14
Log Likelihood	-571.92
Deviance	3180.64
Deviance explained	0.49
Dispersion	16.32
R ²	0.46
GCV score	17.17
Num. obs.	205
Num. smooth terms	3

*** $p < 0.001$; ** $p < 0.01$; * $p < 0.05$

Table 11.18: Generalized additive model with pretest, number of attempted tasks, and uptake based on seen feedback

closely, it at the same time makes more assumptions and is less generalizable. Given that a higher-order, non-linear model is only automatically selected for the pretest score, but not for the learning process predictors under consideration, we go with the simpler model specification following Ockham's principle.

11.2.5 Correct at First Try

In this section, we continue to investigate the role of answers submitted correctly at some point by testing if answers submitted correctly at first try are partly responsible for the significant effect of the former.

Feature Operationalization

We extracted a feature from the interaction log that expresses the number of task fields for which learners submitted a correct answer at first try. In order to compute the feature, we extracted all interaction steps and sorted them per learner by task field and ordered them by their time stamp. Then we counted for how many cases the first logging token for each task field was associated with a correct answer. Note that this approach does not exclude cases in which

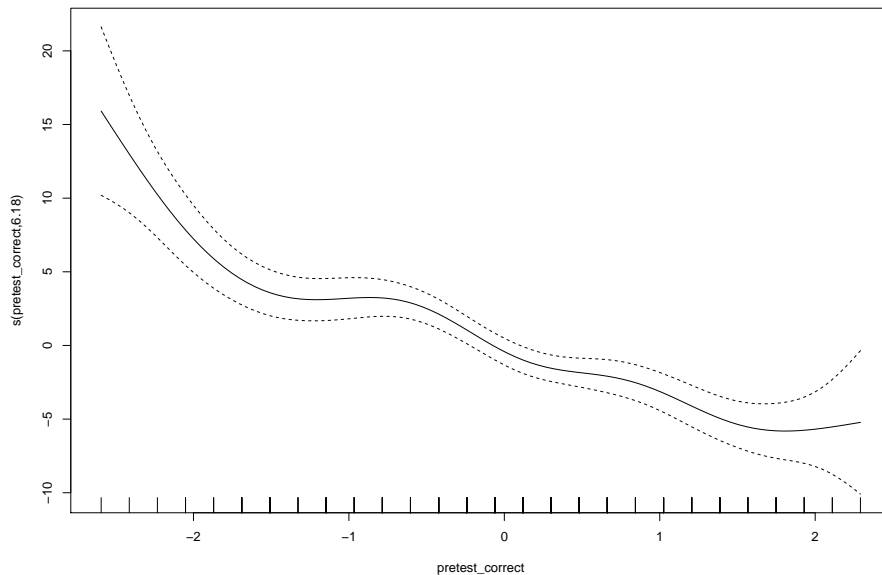


Figure 11.6: Polynomial fit of pretest score in relation to change score

learners later re-evaluated their answer again as correct, e.g. in a check-all process triggered by a button or when re-entering a task. The extracted feature values were z-standardized (centered and scaled).

Results

Testing for Main Effects We tested for a potential main effect of answers submitted as correct at first try. Table 11.19 shows that there is a highly significant effect of answers submitted correct at first try.

Testing for Moderation Effects As a next step, we tested for potential interaction effects with the assignment to experimental group, the achieved number of points in the pretest and number of attempted tasks. Table 11.20 shows that the feature *correctAtFirstTry* does not interact with *group*, but that there is a significant interaction with the pretest score.

Testing for Mediation Effects In order to find out more about the relation between pretest score and correct at first try, we conducted a causal mediation analysis to test a potential mediation of the pretest score through the number of correct tries at first attempt. Table 11.21 shows the significant total effect of the pretest score. In Table 11.20, we see that there is a significant effect of the independent variable pretest score on the mediator correct at first try.

<i>Dependent variable:</i>	
	changeScore
correctAtFirstTry	0.899*** (0.318)
groupintervention	2.592*** (0.580)
pretest_correct	-3.488*** (0.300)
Constant	4.959*** (0.414)
Observations	205
R ²	0.446
Adjusted R ²	0.438
Residual Std. Error	4.137 (df = 201)
F Statistic	54.010*** (df = 3; 201)
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01

Table 11.19: Regression model: highly significant main effect of answers correct at first try

<i>Dependent variable:</i>			
	(1)	(2)	(3)
	changeScore		
correctAtFirstTry	0.615 (0.412)	0.960*** (0.318)	1.614*** (0.383)
numTasksAttempted			-1.009*** (0.361)
groupintervention	2.549*** (0.582)	2.631*** (0.577)	2.700*** (0.569)
correctAtFirstTry:pretest_correct		-0.575* (0.313)	
correctAtFirstTry:numTasksAttempted			-0.310 (0.236)
pretest_correct	-3.511*** (0.301)	-3.420*** (0.301)	-3.529*** (0.294)
correctAtFirstTry:groupintervention	0.678 (0.624)		
Constant	4.999*** (0.415)	5.074*** (0.416)	5.026*** (0.431)
Observations	205	205	205
R ²	0.450	0.456	0.478
Adjusted R ²	0.439	0.445	0.465
Residual Std. Error	4.135 (df = 200)	4.113 (df = 200)	4.038 (df = 199)
F Statistic	40.840*** (df = 4; 200)	41.830*** (df = 4; 200)	36.399*** (df = 5; 199)

Note: * p<0.1; ** p<0.05; *** p<0.01

Table 11.20: Regression model: no significant interaction effects of answers correct at first try with group assignment and number of attempted tasks, but with pretest score

<i>Dependent variable:</i>	
changeScore	
groupintervention	2.508*** (0.590)
pretest_correct	-3.274*** (0.295)
Constant	5.064*** (0.419)
Observations	205
R ²	0.424
Adjusted R ²	0.419
Residual Std. Error	4.208 (df = 202)
F Statistic	74.449*** (df = 2; 202)
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01

Table 11.21: Causal mediation analysis: significant total effect of pretest score

<i>Dependent variable:</i>	
correctAtFirstTry	
groupintervention	-0.093 (0.128)
pretest_correct	0.238*** (0.064)
Constant	0.116 (0.091)
Observations	205
R ²	0.068
Adjusted R ²	0.059
Residual Std. Error	0.915 (df = 202)
F Statistic	7.395*** (df = 2; 202)
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01

Table 11.22: Causal mediation analysis: significant effect of pretest score on the mediator correct at first try

Table 11.23 shows the combined effect of the mediator and independent variable on the dependent variable *changeScore*. As displayed in the table, there is a highly significant effect of both the mediator and the independent variable on the outcome variable.

	<i>Dependent variable:</i>
	changeScore
correctAtFirstTry	0.899*** (0.318)
groupintervention	2.592*** (0.580)
pretest_correct	-3.488*** (0.300)
Constant	4.959*** (0.414)
Observations	205
R ²	0.446
Adjusted R ²	0.438
Residual Std. Error	4.137 (df = 201)
F Statistic	54.010*** (df = 3; 201)
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01

Table 11.23: Causal mediation analysis: significant effect of pretest score and mediator correct at first try on gain score

The summary of a causal mediation analysis combining the models from tables 11.21, 11.22 and 11.23 based on 1,000 iterations over the data set is depicted in Table 11.24. The table, generated as output of the *mediation* package in R (cf. Tingley et al. (2014)), shows a highly significant ($p = 0.01$) average causal mediation effect of $0,23823 * 0,8994 = 0.2143$, a highly significant average direct effect ($p < 0.001$), a highly significant total effect ($p < 0.001$), and a highly significant bootstrapped unstandardized effect ($p = 0.01$). This analysis shows that the *correct at first try* feature is a causal mediator of the pretest score. We visualize the results of the causal mediation analysis in Figure 11.24.

We furthermore tested if there is a mediation effect with group as the independent variable and correct at first try as mediator. In order to test this, we computed models predicting *correctAtFirstTry* with group alone and with pretest and group together. As reported above, we found a mediation with respect to pretest, but we found no mediation with respect to group. Taken together, the results indicate that previous knowledge plays a significant role with

Causal Mediation Analysis

Nonparametric Bootstrap Confidence Intervals with the Percentile Method

	Estimate	95% CI Lower	95% CI Upper	p-value
ACME	0.2143	0.0543	0.43	0.01 **
ADE	-3.4881	-4.1434	-2.81	<0.0000000000000002 ***
Total Effect	-3.2739	-3.9037	-2.68	<0.0000000000000002 ***
Prop. Mediated	-0.0654	-0.1324	-0.02	0.01 **

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Sample Size Used: 205

Simulations: 1000

Table 11.24: Causal mediation analysis results of correct at first try based on 1.000 iterations

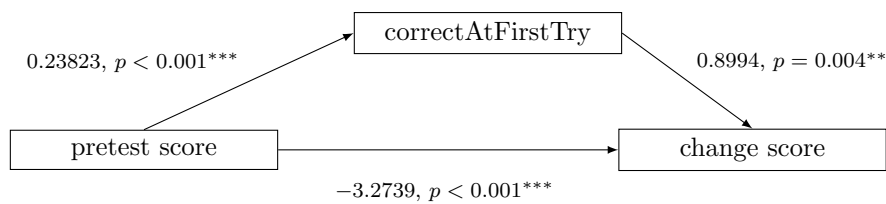


Figure 11.7: Mediation analysis of pretest score and number of items correctly answered at first try with respect to the change score

respect to learning gains. Answers submitted correct at first try are a proxy for and express previous knowledge.

11.2.6 Time-on-Task

Feature Operationalization

We operationalize time-on-task in two features, both of which are directly logged by the system and can be estimated from parsing the server logs given that they contain time stamp information. For the experiments described here, we parsed the server log file. As described in Table 11.25, we distinguish raw time-on-task from time-on-task working. The former is the time between entering and submitting an exercise. In cases in which a learner saved, quit, and later continued an exercise, the feature expresses the sum of all sessions on this exercise. In contrast, time-on-task working only starts to be counted as soon as a learner interacts with the system’s feedback mechanism. This excludes for example the time in which learners read the instructions or consume a task’s associated media like a reading text, audio recording, or video clip. Additionally, for both features, we provide an abstracted version in which the exercise sessions (i.e. contexts in which time-on-task was logged) are counted. This abstracts away from concrete numbers and is thus more robust to outliers. Since these features are quite sensitive to outliers, for example when learners are interrupted during homework and leave the application open, we need to deal with outlier normalization for the feature values. Since we know the maximum duration (60 minutes) that the server keeps a session open, we exclude (i.e. do not count) all sessions which are longer than the maximum threshold, since these are learners who left the application open without logging out. In order to normalize the feature values and make them comparable to other feature values, we z-standardized (centered and scaled) and log-transformed the raw feature values before the analyses, following the procedure of the previous sections. In order to obtain the feature values, we parsed the server log files and added up the time-on-task obtained from server log files. As a preparation for this step, we first needed to reconstruct missing time-on-task statements (described below) caused by a bug in the system. This shows the importance of having multiple representations of the data: with only the exercise log, the reconstruction would not have been noticed and impossible to conduct.

feature	description
workingTime	logged working time in seconds (starting from first feedback interaction)
workingSessions	number of exercise sessions with working time
timeOnTask	logged time in seconds (starting when learners enter exercises)
timeOnTaskSessions	number of exercise sessions with log time statements

Table 11.25: Time-on-task feature operationalization

Reconstructing Time-on-task Log Statements not Logged Over the duration of the study, the tutoring system’s application logic contained a bug that prevented time-on-task statements to be logged immediately before an exercise was submitted to the teacher. This led to the fact that for the subtasks that students worked on before they submitted to the teacher (typically the last subtask of a task), the time-on-task was not logged or added to the data base counter. In this section we describe how the missing time-on-task was reconstructed.

The fundamental challenge underlying the reconstruction of time-on-task statements is that the granularity of the task identifiers in the time-on-task log statements and the practice interface opening statements on the one hand, and the granularity of the task identifiers in the upload statements of exercises on the other hand are not on the same level. The time-on-task measures and their log statement expressions are based on and expressed in subtasks, whereas the opening and upload statements are represented on the task level.

This leads to the necessity to (a) reconstruct the missing time-on-task and its log statement expression before upload statements and (2) attribute the time-on-task to a specific subtask. In order to find out which subtask the learner worked on before uploading an exercise, different cases need to be distinguished and treated differently.

As Algorithm 1 shows, we chose a lookback approach with several nested lookback loops and multiple nested conditional statements. A lookback approach was chosen since in a lookback, the missing statements can be inserted at the position of the reading head and then continued to avoid concurrent modification exceptions and recurring cycles.

The algorithm scans the server log files until it encounters an upload statement (line 3). Then it starts a lookback loop (line 4) to search if there are time-on-task log statements for the task under consideration (line 5).

A necessary distinction that needs to be made is the difference between tasks that only contain one single subtask (line 8) and tasks that have multiple subtasks (line 16). In the first case, the time-on-task can always be attributed to the first subtask from the time point on where the user re-entered the practice interface with this task (line 10). It needs to be added to the previously logged time-on-task (lines 7, 13) in order to compute the total time-on-task for this task and subtask.

In case the uploaded task has multiple subtasks, the algorithm looks at which statement is temporally closer to the upload statement, i.e. is less far away from the upload statement. This can be either a time-on-task log statement, or a practice interface opening statement. In case the exercise opening statement is closer, then the user can not have changed between subtasks and the missing time-on-task is attributed to the subtask with which the user entered the practice interface (lines 18, 20). In case there is a time-on-task statement closer to the upload statement, then the time-on-task needs to be attributed not to this, but to another subtask (line 22).

A completely different case needs to be handled if no previous time-on-task log statements exist (line 27). The algorithm here simply searches for the

statement indicating that the learner opened this exercise and then subtracts this time point from the upload time point (line 31).

The generated time-on-task log statements are added to the log file directly after the uploads statement (line 34), with the exactly same format as the regular time-on-task log statements to make them not distinguishable from the original one. This approach was chosen since all subsequent processing tools that analyze the log then do not need to be adapted. We validated the algorithm both qualitatively by doing a plausibility analysis of the generated statements (with markers to distinguish the newly generated log statements), and quantitatively by comparing it with an alternative approach of computing time-on-task based on auto-save statements.

Results

Testing for Main Effects We found no main effect for any of the time-on-task features.

Testing for Moderation Effects Next, we tested for potential interaction effects between the time-on-task features and the experimental group assignment. For one feature, the time-on-task since entering an exercise, we found a significant interaction with the experimental group (cf. Table 11.26). As the models shows, there is a significant positive effect of time-on-task for the control group (0.988, $p=0.012261$, left model), but not for the intervention group (right model).

We furthermore tested for interactions with pretest scores and the number of tasks learner attempted. As displayed in Table 11.27, we found a significant interaction between time-on-task and the number of attempted tasks. The model shows a negative coefficient of -0.623 of the interaction term.

We furthermore tested for potential interactions with pretest scores. We found a significant interaction between the pretest score and time-on-task (see Table 11.28) with a negative coefficient of -0.751 .

Testing for Mediation Effects We tested for potential mediation effects with *group* and *pretest_score* as independent variables, and the time-on-task features as (potential) mediators. The analyses showed no mediation effects. However, one model showed an interesting effect of *pretest_score* on time-on-task (Table 11.29, left model). This effect is, however, not strong enough to persist when time-on-task is included as a main effect to predict the change score (Table 11.29, right model).

11.3 Discussion

We showed that learners who perform well when working in the system, i.e. submit many correct answers, also perform well in the associated post test. The analyses showed that this effect can be attributed to two learning paths leading

```

1 for int i = 0; i < lines.size(); i++ do
2   print lines.get(i);
3   if msg.contains("uploads the submission") then
4     // lookback function
5     for int j = i - 1; j > -1; j- do
6       if prevLine.contains("on the exercise") then
7         // if the previous statement refers to the same task
8         if taskIDsMatch(tid, prevTaskID) then
9           // rember already logged time-on-task
10          prevTotalTime = prevLine[2]; // in case of one single
11          subtask, the missing time-on-task has to refer
12          to this one
13          if numSubTasks of current task == 1 then
14            for int k = i; k > -1; k- do
15              if kMsg.contains("opens the practice") then
16                if taskIDsMatch(kTaskID, tid) then
17                  // local time-on-task since opening
18                  an exercise
19                  compTime = timepoint - kTime;
20                  compTotalTime = prevTotalTime +
21                  compTime;
22                  foundTime = true;
23                  // end lookback
24                  break jloop;
25
26          else
27            for int k = i; k > -1; k- do
28              // what comes first: entering the exercise
29              or a time-spent log what happened closer
30              to the upload is the reference point
31              if kLine.contains("opens the practice") or
32              kLine.contains("on the exercise") then
33                if taskIDsMatch(kTaskID, tid) then
34                  if kLine.contains("opens the practice")
35                  then
36                    oTime = kTime;
37                  else
38                    oTime = iTime;
39                  compTime = timepoint - oTime;
40                  foundTime = true;
41                  break jloop;
42
43          // if no previously recored time has been found, compute the
44          time between opening the practice interface and the
45          upload, since then the students can not have left it
46          if !foundTime then
47            for int j = i - 1; j > -1; j- do
48              if prevLine.contains("opens the practice") then
49                if taskIDsMatch(tid, prevTaskID) then
50                  compTime = timepoint - prevTime;
51                  compTotalTime = compTime;
52                  break;
53
54          print compTime, compTotalTime

```

Algorithm 1: Time-on-task reconstruction pseudo-code algorithm

	<i>Dependent variable:</i>	
	changeScore	
	(1)	(2)
timeOnTask	0.988** (0.391)	-0.101 (0.316)
groupintervention	2.167*** (0.608)	
groupcontrol		-2.167*** (0.608)
pretest_correct	-3.292*** (0.292)	-3.292*** (0.292)
timeOnTask:groupintervention	-1.088** (0.503)	
timeOnTask:groupcontrol		1.088** (0.503)
Constant	5.365*** (0.432)	7.532*** (0.427)
Observations	205	205
R ²	0.442	0.442
Adjusted R ²	0.431	0.431
Residual Std. Error (df = 200)	4.162	4.162
F Statistic (df = 4; 200)	39.674***	39.674***
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01	

Table 11.26: Regression model: significant interaction between experimental group and time-on-task

	<i>Dependent variable:</i>
	changeScore
timeOnTask	0.582 (0.372)
numTasksAttempted	-0.383 (0.349)
groupintervention	2.512*** (0.584)
pretest_correct	-3.311*** (0.296)
timeOnTask:numTasksAttempted	-0.623** (0.312)
Constant	5.490*** (0.445)
Observations	205
R ²	0.446
Adjusted R ²	0.433
Residual Std. Error	4.158 (df = 199)
F Statistic	32.097*** (df = 5; 199)
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01

Table 11.27: Regression model: significant interaction between time-on-task and number of attempted tasks

	<i>Dependent variable:</i>
	changeScore
timeOnTask	0.127 (0.339)
pretest_correct	-3.459*** (0.303)
groupintervention	2.498*** (0.585)
timeOnTask:pretest_correct	-0.751** (0.355)
Constant	5.206*** (0.426)
Observations	205
R ²	0.440
Adjusted R ²	0.429
Residual Std. Error	4.170 (df = 200)
F Statistic	39.338*** (df = 4; 200)
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01

Table 11.28: Regression model: significant interaction between time-on-task and pretest score

	<i>Dependent variable:</i>	
	timeOnTask (1)	changeScore (2)
timeOnTask		0.356 (0.324)
groupintervention	0.096 (0.128)	2.474*** (0.590)
pretest_correct	0.145** (0.064)	-3.326*** (0.299)
Constant	-0.287*** (0.091)	5.167*** (0.429)
Observations	205	205
R ²	0.026	0.428
Adjusted R ²	0.017	0.419
Residual Std. Error	0.913 (df = 202)	4.206 (df = 201)
F Statistic	2.732* (df = 2; 202)	50.087*** (df = 3; 201)

Note: *p<0.1; **p<0.05; ***p<0.01

Table 11.29: Regression model: significant effect of pretest on time-on-task, but no significant effect when predicting the change score

to these correct answers: answers submitted correct at first try, and successful uptake after specific feedback. Via a causal mediation analysis, we showed that the answers submitted correct at first try are actually a (partial) mediator for the pretest score and thus previous knowledge. This finding underlines the importance of taking previous domain knowledge into account when predicting learning gains. In systems implemented in ecologically valid settings outside of study contexts, this information is relevant when designing Learning Analytics tools: since in this case often no pretest results are available, systems still have access to system logs and can extract the number of answers submitted correct at first try.

Furthermore, we showed that not only previous knowledge resulting in correct answers at first try, but also successful uptake after specific feedback is a learning process measure resulting in the learning product of correct answer and a significant predictor of learning gains. In order to test whether it is the attention to feedback or only the attention to the form of the learner answer, we computed two features that form a minimal pair: uptake based on displayed feedback and uptake based on blocked feedback. The difference is that in the second case, the learners only knew that their answer was not accepted due to the absence of a green check mark, whereas in the former case, learners received an explanation as to why and how their learner answer was not accepted by the system. The results show that uptake based on actually seen feedback is a significant predictor for the change in explicit grammatical knowledge from pretest to post test. In contrast, uptake for cases in which the specific feedback was withheld and learners did not receive an explanation of why their answer was incorrect before being corrected, was found to be not a significant predictor. The only difference between the two features is thus the presence of feedback and explanation of errors. Since we showed that uptake with feedback is a significant predictor, we showed the effectiveness of feedback when learners attend to it. Furthermore, by conducting a causal mediation analysis, we showed that uptake is mediated through the number of attempted exercises. It thus makes a difference whether learners can implement feedback in only few or across multiple tasks. With different tasks offering different prerequisites, contexts, formats, requirements, language forms, and thus feedback to different aspects of language, the mediation highlights the need to consistently pay attention to feedback in different environments and practice forms across tasks. The results demonstrate that not (only) attention to forms, but feedback and especially attention to feedback is relevant for learning. Our results fully confirm and are compatible with previous research on the role of uptake (cf. section 2.3 for a detailed discussion of the role of uptake). Regarding the implications of these findings, it is thus not sufficient for systems to only indicate to learners that their answer is not correct. What makes the difference is feedback explaining the error, and whether learners pay attention to it across different tasks.

Providing feedback is, however, not enough for fostering learning. Learners need to pay attention to and process the feedback. This finding is reflected in the fact that raw feedback counts are not indicative for predicting learning gains, but rather only instances of uptake in which learners were able to make

use of the feedback.

The finding that time-on-task is a significant predictor interacting with the experimental group, with a significant positive effect for the control group, and a negative (though insignificant) effect for the intervention group, supports this finding: when learners do not have access to feedback (control group), they need to spend more time to solve the exercises, but investing time eventually pays off. A potential speculative interpretation of the negative effect of time-on-task for the intervention group could be that those learners in the intervention group who need to spend more time in comparison to their peers, are not efficiently using the feedback offered by the system to them.

In essence, these findings show that *(a) previous knowledge, (b) specific feedback and (c) attention of learners to specific feedback and not only forms and (d) strategies for efficient processing of feedback offered to learners* are significant predictors of learning gains.

The results link up with previous research in SLA regarding the role of practice, feedback and uptake (cf. chapter 2). From the results, we draw the following implications and conclusions: First of all, previous knowledge is an essential parameter in determining how well learners will perform in assessments. This can be manifested in pretest scores (if available), but also in learning process data in non-laboratory settings in the form of answers submitted correct at first try. Systems should track the number of correct attempts at first try to estimate the previous knowledge of learners. Secondly, feedback per se is not helpful. What matters is specific feedback explaining the source of the error, and not just pointing out errors. It is insufficient for systems to point towards an error, for learning systems have to explain errors. Thirdly, even specific feedback per se is not helpful by default – simply counting the number of provided specific feedback messages is not indicative of learning gains. What matters is whether learners pay attention to the feedback and are consequently able to use it successfully, i.e. whether learners show successful uptake. Systems need to record not necessarily the provided feedback, but rather instances of successful uptake. Feedback can reduce the time learners need to spend to complete exercises. Thinking further along these lines, systems should make the feedback process as motivating and interesting as possible to keep learners motivated and cognitively focused on it. Finally, we demonstrated that it is not sufficient to look at learning outcomes only. We demonstrated that there are fundamentally different learning processes that result in the same learning outcomes, and that these learning processes are significant predictors for learning outcomes in their own right and need to be considered.

11.4 Limitations and Avenues for Future Research

In the context of this study, we worked on deriving learning process measures from system interaction patterns and using them as predictors for learning gains. This approach yielded promising and significant results, but we need to acknowledge that this is a rather reductionist approach. We attempt to approximate a highly complex and not entirely understood mechanism, the human capacity to learn languages, or at least grammatical structures, with a small set of interaction measures. While we showed that these measures contribute to the predictions, we assume that there are more and far more complex processes not captured by the current analyses.

In the present study, only a very limited number of cognitive measures are available. For example, there is no ambulatory assessment of e.g. attention. Given that we showed that attention to feedback is a significant predictor, having information available on the current mood/state of the user could help interpret the results. Apart from the cognitive aspects, more features could be imagined like modeling the time when learners worked (e.g. morning vs. midnight), location (e.g. at school/home/in the bus), or device (desktop computer vs. smart phone) learners interacted with.

Another limitation is that no qualitative data is available, e.g. verbal reports, that could be triangulated with the existing quantitative data. However, given that these data are often not available when systems are used outside of laboratory contexts, this is a consequence of the experimental design. While ecologically valid settings provide a highly valuable opportunity to conduct end-to-end testing, and allows to put technology and the concept behind it to the test where it matters, a potential limitation is that many factors are not controlled, constituting potentially confounding variables. We did not record or consider social factors, for example learners comparing themselves with classmates or group dynamics arising when learners are working in the class room in the presence of their peers.

Another limitation is the availability and quality of specific feedback. Going through the log files, we recognized that many systematic error patterns are not yet handled by the system. Also, in some case, the feedback was not very intuitive or appropriate given the error. We acknowledge that there is room for improvement in the feedback mechanism that, given more time, could be tackled. With better and a broader range of feedback messages, the effects could potentially become even clearer.

Furthermore, we are only testing explicit grammatical knowledge, but there are other measures for competence beyond this. For example, we did not conduct tests targeting receptive skills or whether learners are able to transfer the knowledge outside of grammar-focused exercise contexts. This was the case in the delayed post test, but analyzing the delayed post test is beyond the scope of this thesis.

The interpretation of the mediation analyses depends on whether *pretest_score*

is accepted as the independent variable in this case (cf. also VanderWeele and Vansteelandt (2009)). It is possible to argue that the independent variable in a mediation analysis needs to be an experimental condition assigned before the mediators were measured and that additionally needs to have been assigned randomly, such as the *group* variable. Furthermore, having a negative and a positive path in a mediation, and values that are close to zero (relatively speaking), could point at statistical artifacts. Nevertheless, the regression models by themselves offer valuable insights into dependencies between factors. Even if for certain models no causal link could be established depending on the methodological tradition, the individual models are methodologically valid and can be interpreted in their own right. An avenue for future research could be to estimate Complier Average Causal Effects (CACE) models (Nagengast et al., 2018) that take into account both the group assignment (experimental versus control) as well as the compliance with the conditions of the experiment.

Another methodological limitation is the focus on linear models (cf. also McNeish et al. (2017)). While we found and reported results by following the standard procedure of fitting linear models and explored some non-linear models to complement this. Future work needs to approach the data with alternative, non-linear models. If the focus is on model accuracy, future work could also integrate random effects that model the clustering of learners into school classes (cf. also McNeish and Stapleton (2016)) and implement hierarchical modeling (Raudenbush and Bryk, 2002). However, especially in exploratory data analyses, there is a danger of over-fitting the data at hand if very complex and powerful (higher polynomial) models are employed from the beginning (Baayen et al., 2017). In our analyses, we were interested in whether there are effects at all, and in which direction(s) – having simpler models that make fewer assumptions in this case supports the interpretability of the results and paves the way for more complex models to be estimated later.

From a methodological point of view regarding the study setup, follow-up studies should conduct a power analysis in advance to estimate the reliability of the statistical effects – conducting it post-hoc would be highly problematic (Zhang et al., 2019).

Another limitation is that we focused on one specific work book theme, as justified in section 11.1.1. Future work needs to test whether the results found in the context of this study generalize to other, comparable grammar topics.

In follow-up studies that employ the FeedBook or systems derived from it, special attention should be paid to the factors discussed in the following. We suggest to directly and explicitly log these features in suitable data structures in order to facilitate the hypothesis testing approach. First of all, the system should log the number of answers submitted correctly at first try – or rather continue logging this or keep logging this in subsequent learner models, since the original FeedBook learner model already provides functions to log this (cf. chapter 8). Secondly, uptake based on feedback in relation to the time learners spend on exercises should be represented prominently in any teacher dashboard implemented in the system. Finally, the number of correctly submitted answers should be represented not only per exercise, but rather across exercises since we

showed that this is a highly predictive feature. In an ideal scenario, a teacher dashboard in the system would make these metrics available to provide teachers with valuable diagnostic information to help them identify on the one hand struggling learners so that they can provide special support to them, and on the other hand see which learners are performing already very well and can be pushed further.

Part IV

Conclusion

Chapter 12

Summary and Outlook

12.1 Discussion and Conclusions

The purpose of this thesis was to gain a better understanding of the intersection between Second Language Acquisition, Intelligent Computer-Assisted Language Learning, Computational Linguistics, Learning Analytics, and Empirical Educational Science. This is only possible with both a theoretical background (Part I) and a practical application that supports learner interaction and data collection (Part II).

The thesis takes an exploratory approach towards the potentials of interdisciplinary work in this context (Part III). We pursued two main strands of research, which use different methodologies, but serve the same purpose: contributing to an understanding and enabling an improvement of learning and the conditions in which it occurs - the central goal of Learning Analytics research. In order to enable interesting analyses, rich data is necessary. In our case, these data come from an Intelligent Computer-Assisted Language Learning system with detailed interaction logs.

The first goal was to develop Learning Analytics for end users, i.e. tools for humans to make decisions to understand and improve learning. The key finding here is that different user groups need different tools tailored towards their specific needs. While the data underlying the tools is the same for all tools, our work highlights that it needs to be used and visualized very differently for different stakeholders in the educational context.

Learners are most interested in their proficiency and how well they are prepared for the next exam. Therefore, we propose an open learner model that informs them about their proficiency on different levels and suggests exercises to maximize it. Teachers have only very limited time and need to know quickly about the progress and problems of their students, primarily in the preparation of the next classroom session by getting insights into the learning processes of their students and diagnostic information about their learners' skills. To this end, we propose Learning Analytics tools that both aggregate the interaction

data of a teacher's school class on individual items in assignments, as well as tools to show the interaction by individual students. These tools help teachers diagnose problems in the understanding of grammatical structures by their students and allow them to extract valuable information that can be used to adjust the content of the next classroom lecture. Material designers and educational administrators again need different tools that allow to see how the entire learner population interacts with exercises. We present Learning Analytics tools that enable them to see how well exercises are working, identify problems in the exercise contents and the system's feedback behavior, and remedy these problems. The tools for the different user groups help educational stakeholders to learn and work more efficiently and provide opportunities unavailable in traditional paper-based setups.

The second goal was to extend the understanding of learning process factors contributing to learning products. We exploratorily tested whether and which learning process measures, identified in related work as contributing to learning, have a predictive power regarding learning products. The results strongly imply that correctly submitted answers are predictive of learning gains, i.e. instances in which learners manage to submit a correct answer. We split this learning product up into different learning processes that can result in obtaining correct answers: answers submitted correctly at first try and answers submitted correctly after feedback, with and without specific explanations. We showed that the number of answers submitted correct at first try is both a highly significant predictor and a proxy for previous knowledge. We furthermore obtained evidence that instances of successful uptake predict learning gains – but only after specific feedback was actually displayed. Successful uptake after blocked feedback, where learners only know that their answer was not correct but not why, is not a significant predictor. Time-on-task was found to be a significant predictor for the control group (i.e. an interaction effect), and negatively correlated with the test outcomes of the intervention group. One interpretation of this finding is that when learners in the intervention group need more time than their peers, they lack a deep and efficient processing of the displayed feedback. On the contrary, when learners in the control group invest more time to overcome the lack of specific feedback, this effort pays off. We showed that raw feedback counts are not indicative for learning gains. What makes the difference between high and low learning gains is the attention to specific feedback and efficient strategies for processing it.

Taken together, our findings underline the need to consider learning process variables in addition to learning products in a very fine-grained manner. While our results are compatible with previous research on the effect of uptake, in this thesis we have shown that uptake per se is not a predictor of learning gains, but rather uptake as a reaction to specific feedback with learners paying attention to the feedback with efficient strategies for processing it efficiently. Whereas time-on-task has been demonstrated to be a positive factor for learning in previous research, we show that investing more time is beneficial, but only if there is no other help in the form of feedback available (i.e. for the control group) – if feedback is available, it is essential to engage with and make use of the

feedback. Especially notable is that the findings constitute robust evidence based on ecologically valid data from diverse settings that were only loosely controlled (in contrast to the majority of research with lab studies and adult learners).

There are a range of limitations concerning the results of the thesis (cf. section 11.4 for a detailed discussion). A first limitation concerns the reductionist approach chosen for the analysis: we approximate a highly complex process (the human capacity to learn) with a small number of features. However, given that in this specific domain there is so little research, we view this as a first step towards understanding a more complex process. A second limitation related to the first one is that we do not consider external factors like time (morning, midnight, . . .), location (mobile, at home, in school), or social context of the interaction data used in the analyses. Furthermore, we do not make use of cognitive measures and do not consider motivation or qualitative data that could serve as valuable sources of information that complement the existing measures. Certain limitations of this study could be addressed in future research. For example, we know – from the Learning Analytics tools presented in this thesis – that the system and its feedback are not perfect. Variance can be attributed to errors systematically not covered by the feedback algorithm or feedback triggered for ambiguous cases. In terms of outcome measures, we used only one test measure in the form of the change in the number of points between the pretest and post test but did not include more implicit measures and also did not check for potential transfer of the knowledge to different types of tests. This can be addressed in future work by analyzing the free writing tasks collected at the end of the school year. Although the present methodological approach clearly supports our findings, the employed linear models could be extended with more non-linear models if future work focuses more on modeling accuracy. We focused on linear models as a standard practice for exploratory data analyses and chose simpler over more complex (higher order polynomial) models to foster interpretability and to avoid making too many assumptions or over-fitting the data (cf. also Baayen et al. (2017)).

Despite these limitations, these results suggest several theoretical and practical implications. First of all, follow-up systems to FeedBook need to enable a better logging of relevant features. The learning process features identified as predictors of learning gains need to be logged explicitly so that no reconstruction via external log files is necessary. Furthermore, key features need to be visualized prominently in Learning Analytics interfaces for all user groups (students, teachers, material designers, educational administrators). With respect to methods, apart from an extension to non-linear methods, we acknowledge that we considered only one theme. Future work needs to test the generalizability of the findings to comparable data sets and different language constructs.

This research can be seen as a first step towards integrating Computational Linguistics, Second Language Acquisition, Learning Analytics, Empirical Educational Sciences, and Tutoring System Development research. To our knowledge, these lines of research have not been linked in this manner before. With the current system presenting a broad range of Learning Analytics tools next

to each other, future work should consider providing a combination of the tools in the form of dashboards instead of individual interfaces; this should connect the information distributed across different interfaces instead of having them side-by-side. From an analytical perspective, follow-up studies need to take the findings into consideration, test them, and see if they can be confirmed and potentially even refined with more detailed log data and measures. In any case, it is essential that future work respects the unique interdisciplinary context this work is situated in and continues to link the different lines of research.

12.2 Summary and Contributions

Part I: Background

In the first part of the thesis, we provide a comprehensive overview about the interdisciplinary context in which this work is situated. We establish the theoretical background for the system design in the second part, and Learning Analytics tools and the data analyses in the third part.

In the background chapter about Second Language Acquisition, we review the role of practice. We zoom in on learning process observations, specifically the reaction to corrective feedback in the form of uptake and its effects.

We explain the definition, historical development, and domains of application of Learning Analytics and Educational Data Mining in Chapter 3. We argue that the two (proposed) fields are actually different strands of the same field, and that Learning Analytics tools are essential for tutoring systems to provide teachers with insights into the learning process. In the context of tutoring systems that help learners reach a correct solution, without Learning Analytics teachers only see similar learning products even though the learning processes and thus diagnostic information differs drastically.

In the third background chapter, we review tutoring systems, including their architecture, components and implementation. We put a special focus on the generation and provision of feedback to be able to relate back to this aspect when presenting the FeedBook design. This chapter forms the theoretical basis for the chapters about the system design and feedback algorithm design. In this context, we discuss important related systems and describe which aspects of these systems provide solutions to similar challenges/tasks as the FeedBook, and how these solutions are implemented in the respective systems.

Contributions Our contributions in this part are three unpublished, comprehensive literature reviews. We lay the theoretical foundation for part II and III, allowing us to situate our work in this interdisciplinary context.

Part II: Towards an Empirical Basis

We start the second part by describing the FeedBook system, its development, and diverse user interfaces for learners, teachers, and material designers in the

system. Chapter 5 describes the work flows of different user groups in the system, and the user interfaces enabling these actions.

We then turn to the feedback algorithm by describing how feedback for orthography, grammar, and semantics are generated, computed and displayed in FeedBook. We describe both the individual feedback strategies and how they relate to and are combined with each other.

The focus here is a detailed description of how the task context is used in all three feedback modalities to generate feedback, with special emphasis on the grammar feedback algorithm. We explain how pre-generating feedback parts in an offline feedback generation approach is combined with a flexible online matching part. We describe how grammar feedback is extended with feedback on conceptually different types of errors on the level of spelling and meaning.

Chapter 7 describes the FeedBook study in detail. We utilize this chapter to describe the procedure, participants and materials of the year-long study. That is to say, we describe the randomized controlled field trial study and explain the context from which the data comes that is later used in the third part.

Contributions Our contributions in this part are substantial contributions to the implementation and design of the FeedBook system and the feedback algorithm. We extend the description of the feedback algorithms in higher detail than previously described. Moreover, we helped in the study and supported the system, which allowed us to collect the data.

Part III: Learning Analytics in Practice

Part III builds on and combines the previous parts. Based on the theoretical background (part I) and given the system (part II), we show how the two parts can come together.

We start by presenting three chapters on Learning Analytics tools targeting the needs of different user groups: learners, teachers and material designers.

To begin with, we present Learning Analytics tools for individual students, most notably an open learner model integrated in FeedBook. This model offers visualizations about the learner progress on different levels of granularity, the development of accuracy in specific constructs over time, and an automatic, proficiency-dependent and construct-specific sequencing of exercises.

Secondly, we introduce Learning Analytics tools for teachers that show aggregated learning process data of their students. These tools provide insight into the learning process and products, allowing teachers to see the learning paths of their students and allowing them to diagnose errors on the way to help for a better preparation for classroom teaching.

Finally, we showcase Learning Analytics tools for material designers and educational administrators. In this chapter, we provide interfaces that allow them to detect factors related to the quality of the design of exercises and the strengths and weaknesses in the feedback algorithm for specific tasks. Most notably, this includes user interfaces that put effort and learning process measures

in relation to ultimate learning success and visualize process metrics for a range of interaction patterns.

Taking a more analytical perspective, in the last chapter we exploratorily test a range of theoretically motivated learning process measures with respect to their predictive power regarding learning products. The theoretical background on SLA and Learning Analytics guides the selection and operationalization of the features. Via regression models and causal mediation analyses, we linked learning process features to learning products. First of all, we found out that answers submitted correctly are a highly significant predictor of learning outcomes in a grammar test scenario. We then dived deeper by splitting this learning product up into different learning processes that can result in obtaining correct answers: answers submitted correctly at first try and answers submitted correct after feedback, with and without specific explanations. We showed that the number of answers submitted correct at first try is a highly significant predictor and a proxy for previous knowledge. Moreover, we demonstrated that instances of successful uptake predict learning products – but only after specific feedback was actually displayed. Successful uptake after blocked feedback, where learners only know that their answer was not correct but not why, is not a significant predictor. Time-on-task was found to be only a significant predictor for the control group (i.e. an interaction effect), and negatively correlated with the test outcomes of the intervention group. When learners in the intervention group need more time than their peers, they do not process the feedback efficiently. On the contrary, when learners in the control group invest more time to overcome the lack of specific feedback, this effort pays off. Taken together, the findings underline the need to not only consider learning products, but also learning processes. We showed that raw feedback counts are not indicative for learning gains. What makes the difference between high and low learning gains is the attention to specific feedback and efficient strategies for processing it.

Contributions Our contributions in this part constitute the development, implementation and description of novel Learning Analytics tools that make learning processes available. We present a very broad range of tools tailored towards the needs of different user group and providing insight into the learning process. Likewise, we formally establish – with statistical methods – learning process measures whose design is driven by theory and demonstrate that they have a predictive power with respect to learning products in data collected in authentic, ecologically valid environments on a large scale. With the FeedBook study the first large-scale test of intelligent tutoring systems in Germany, the results fill an important research gap by identifying learning process features that predict learning products in ecologically valid school contexts.

12.3 Outlook

In this thesis, we described the FeedBook system and the results obtained based on the work in the corresponding project. This project officially ended in late

2019. However, the system continues to be used and also extended in various follow-up projects.

One project that is taking shape and that the author of this thesis is developing is the Didi system. This tutoring system builds on and extends the FeedBook in various ways. First of all, this system provides exercise materials that are independent of specific workbooks. Secondly, these exercise materials are organized via language topics and sub topics. This is reflected in the student perspective and in the teacher lobby. Figure 12.1 shows the student start page. It shows a tile for every grammar topic, as well as a link to the pretests and to the “Murkse-App”. This is an ambulatory assessment test for working memory that is triggered at login every time the last interaction with this app is more than four hours earlier.

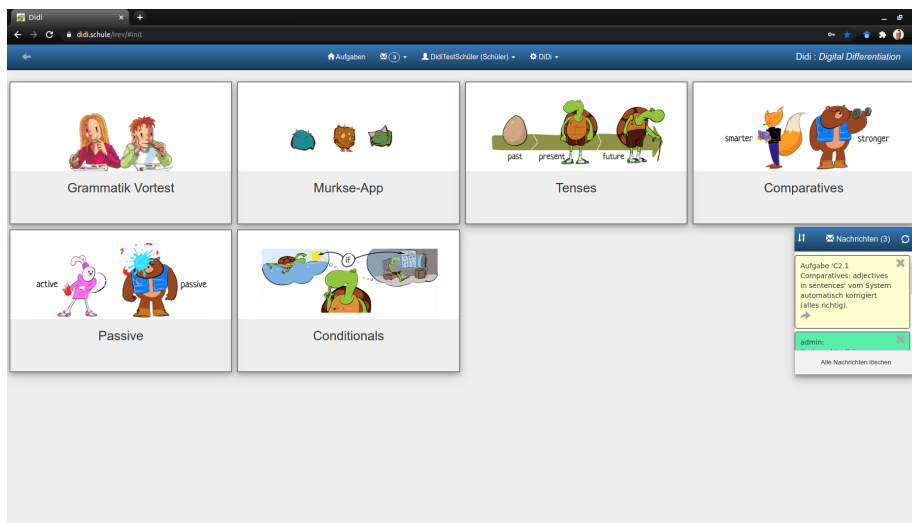


Figure 12.1: Student lobby in Didi

In terms of the teacher interface, Figure 12.2 displays the start page teachers see when logging in. It provides a hierarchical menu with three levels. The filtering of submission has been changed so that initially, all submissions are listed. Selecting a grammar topic or sub topic from the menu filters the current view to display less information. This merged the two previous, distinct submission views into one perspective. While the submission table (on the right hand side) was ported over from FeedBook, there are two new buttons on top of it that allow teachers to lock a grammar topic or set it as active (i.e. as a current assignment). The settings the teacher sets here are reflected in the student perspective with locked tiles and tiles in a different color.

The Didi system furthermore offers a range of new task types, with an emphasis on easier accessible exercises. Figure 12.3 shows a categorization task. Learners need to drag-and-drop elements into categories. In this example, the task is to sort comparative forms into base, comparative, and superlative cate-

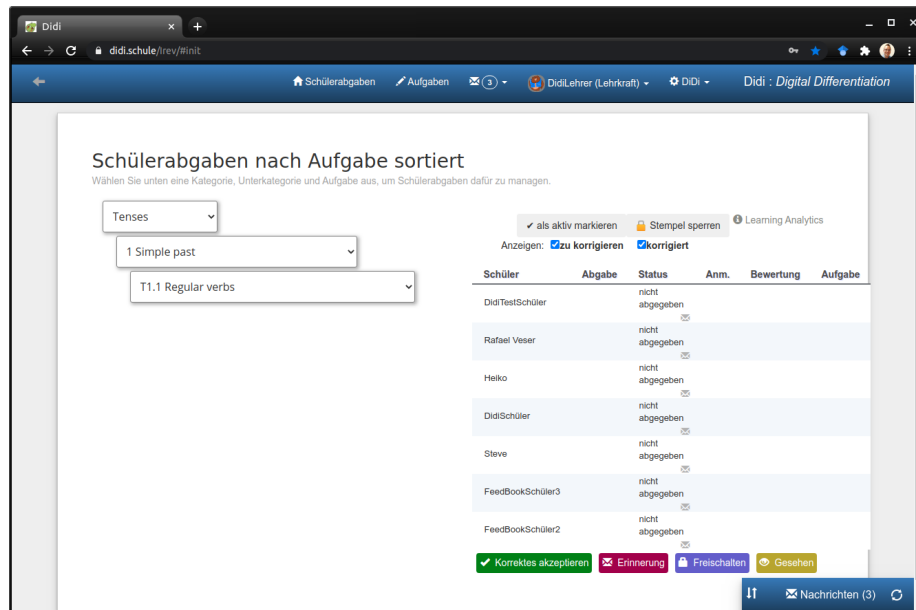


Figure 12.2: Teacher lobby page in Didi

gories. In Figure 12.4, learners are presented with jumbled sentence parts and need to drag-and-drop them to build correct sentences. Another new task type is a memory game, for example the exercise displayed in Figure 12.5. Learners need to find matching cards, with the remaining (i.e. non-matched pairs) cards shuffled every time learners (re-)enter the exercise. In this example, learners are asked to find pairs of infinitives and simple past forms. The last new task type (Figure 12.6) requires learners to underline specific parts of a text. In this example, learners need to mark all nouns in the given text. Common to all of these new task types is that the system provides binary feedback by highlighting the elements in either green (correct) or red (false).

While still under development, the Didi system offers a range of notable improvements over the FeedBook. For example, teachers have the option to sign up themselves and create an account via the system's login page. In terms of NLP, the system implements a new, simplified learner model that uses linguistic, but pedagogically selected categories. Each construct is recorded as either a productive, receptive, or interactive constructs depending on whether learners can edit the form (productive), the form appears in the context (receptive), or learners can work with the form but not change it (interactive, e.g. memory task). Instead of inside the system, the Didi system outsources the annotation of linguistic constructs to a UIMA Rule-based Text Annotation component that allows non-developers to write rules to detect linguistic constructs. The long-term plan for this FeedBook successor system is to provide completely automatic, adaptive sequencing of learning materials. Studies with this system

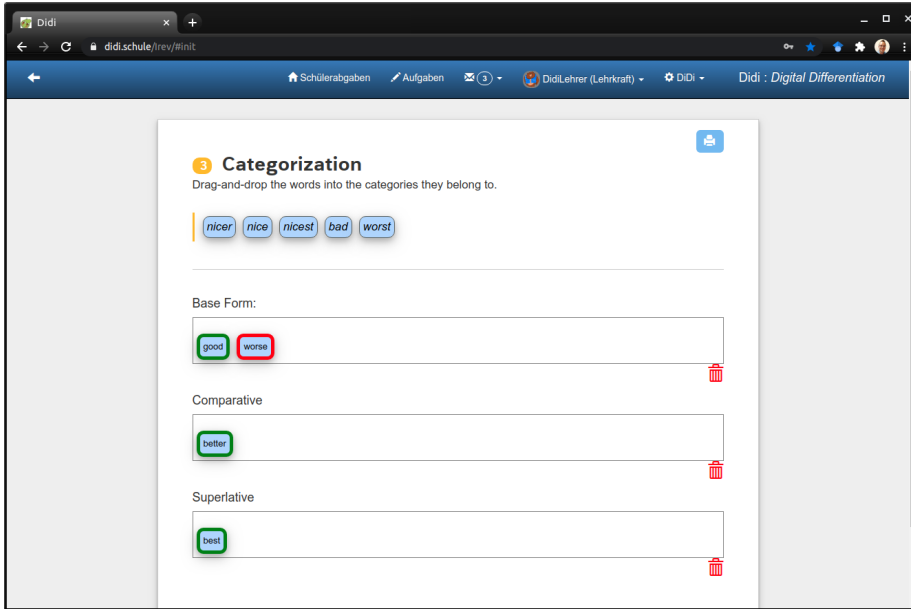


Figure 12.3: Categorization task in Didi

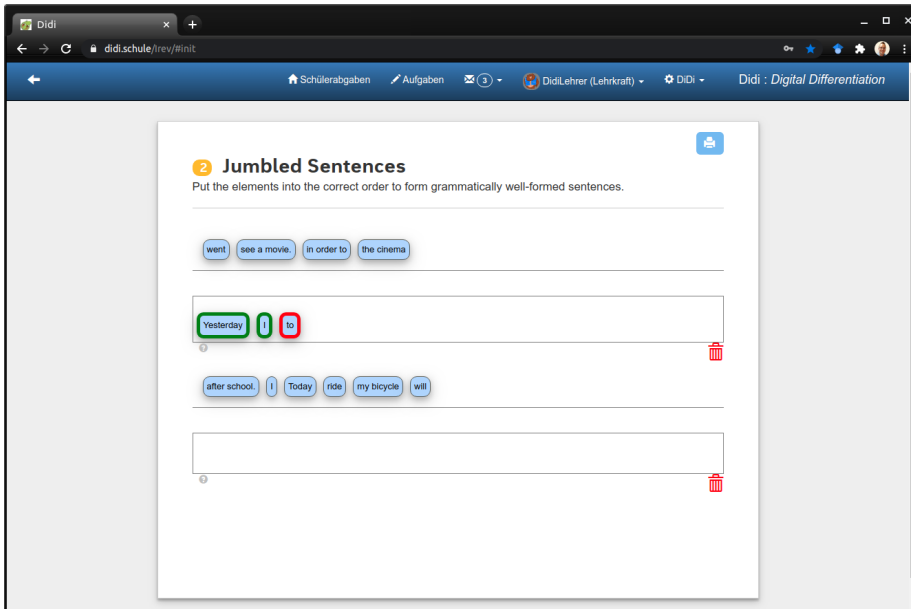


Figure 12.4: Jumbled sentence task in Didi

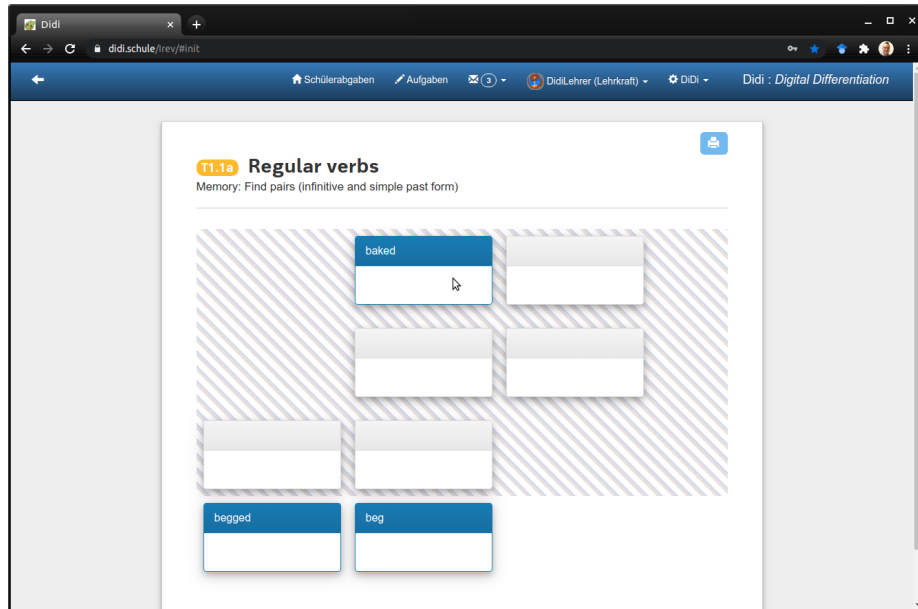


Figure 12.5: Memory task in Didi

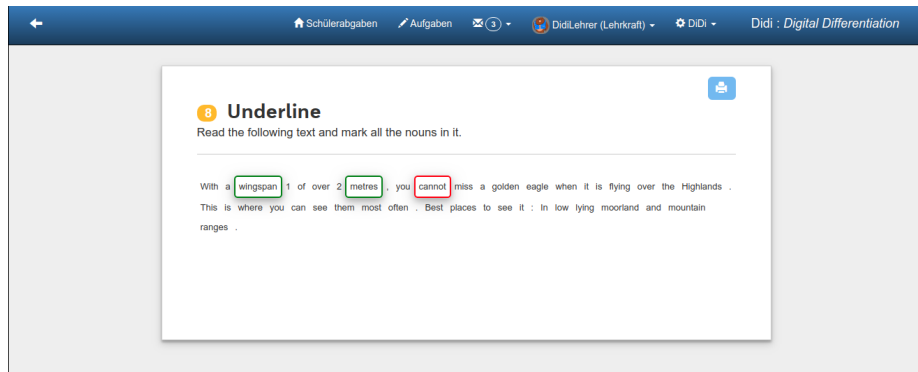


Figure 12.6: Underline task in Didi

and potential follow-up studies will be able to confirm and extend the findings of this thesis (cf. section 11.4).

Bibliography

- Julie D Allen, Deborah Anderson, Joe Becker, Richard Cook, Mark Davis, Peter Edberg, Michael Everson, Asmus Freytag, Laurentiu Iancu, Richard Ishida, and others (the Unicode consortium). *The Unicode Standard*, volume 8. Cite-seer, 2012.
- Luiz Amaral and Detmar Meurers. From recording linguistic competence to supporting inferences about language acquisition in context: Extending the conceptualization of student models for intelligent computer-assisted language learning. *Computer-Assisted Language Learning*, 21(4):323–338, 2008. URL <http://purl.org/dm/papers/amaral-meurers-call108.html>.
- Luiz Amaral and Detmar Meurers. On using intelligent computer-assisted language learning in real-life foreign language teaching and learning. *ReCALL*, 23(1):4–24, 2011.
- Erling B Andersen. A goodness of fit test for the rasch model. *Psychometrika*, 38(1):123–140, 1973.
- John Anderson. The expert module. In Martha Polson and Jeffrey Richardson, editors, *Foundations of Intelligent Tutoring Systems*, pages 21–54. Lawrence Erlbaum Associates, Hillsdale, NJ, 1988.
- Kimberly E Arnold and Matthew D Pistilli. Course signals at purdue: Using learning analytics to increase student success. In *Proceedings of the 2nd international conference on learning analytics and knowledge*, pages 267–270, 2012.
- Ivon Arroyo, Beverly Park Woolf, Winslow Burelson, Kasia Muldner, Dovan Rai, and Minghui Tai. A multimedia adaptive tutoring system for mathematics that addresses cognition, metacognition and affect. *International Journal of Artificial Intelligence in Education*, 24(4):387–426, 2014.
- Peter D Ashworth and Judy Saxton. On ‘competence’. *Journal of further and higher education*, 14(2):3–25, 1990.
- Harald Baayen, Shravan Vasishth, Reinhold Kliegl, and Douglas Bates. The cave of shadows: Addressing the human factor with generalized additive mixed models. *Journal of Memory and Language*, 94:206–234, 2017.

- Stacey Bailey and Detmar Meurers. Diagnosing meaning errors in short answers to reading comprehension questions. In Joel Tetreault, Jill Burstein, and Rachele De Felice, editors, *Proceedings of the 3rd Workshop on Innovative Use of NLP for Building Educational Applications (BEA)*, pages 107–115, Columbus, Ohio, 2008. URL <http://aclweb.org/anthology/W08-0913>.
- Ryan SJD Baker and Kalina Yacef. The state of educational data mining in 2009: A review and future visions. *JEDM—Journal of Educational Data Mining*, 1(1):3–17, 2009.
- Jason Baldridge. The OpenNLP Project. URL: <http://opennlp.apache.org/index.html>, (accessed 25 August 2015), 2005.
- Nicolas Bissantz and Jürgen Hagedorn. Data mining (datenumustererkennung). *Wirtschaftsinformatik*, 51(1):139–144, 2009.
- Trevor Bond, Zi Yan, and Moritz Heene. *Applying the Rasch model: Fundamental measurement in the human sciences*. Routledge, 2020.
- Indranil Bose and Radha K Mahapatra. Business data mining—a machine learning perspective. *Information & management*, 39(3):211–225, 2001.
- Eric Brill and Robert C. Moore. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 286–293, Hong Kong, October 2000. ACL. doi: 10.3115/1075218.1075255. URL <https://www.aclweb.org/anthology/P00-1037>.
- Peter Brusilovsky. Adaptive hypermedia for education and training. *Adaptive technologies for training and education*, 46:46–68, 2012.
- Peter Brusilovsky, Sibel Somyürek, Julio Guerra, Roya Hosseini, and Vladimir Zadorozhny. The value of social: Comparing open student modeling and open social student modeling. In *International conference on user modeling, adaptation, and personalization*, pages 44–55. Springer, 2015.
- Susan Bull. Supporting learning with open learner models. *Planning*, 29(14):1, 2004.
- Susan Bull and Judy Kay. *Student models that invite the learner in: The SMILI open learner modelling framework*. Citeseer, 2006.
- Susan Bull and Judy Kay. Open learner models. In *Advances in intelligent tutoring systems*, pages 301–322. Springer, 2010.
- Susan Bull and Theson Nghiem. Helping learners to understand themselves with a learner model open to students, peers and instructors. In *Proceedings of workshop on individual and group modelling methods that help learners understand themselves, International Conference on Intelligent Tutoring Systems*, volume 2002, pages 5–13. Citeseer, 2002.

- Susan Bull, Manveer Mangat, Andrew Mabbott, AS Abu Issa, and Josie Marsh. Reactions to inspectable learner models: seven year olds to university students. In *Proceedings of workshop on learner modelling for reflection, international conference on artificial intelligence in education*, pages 1–10, 2005.
- Susan Bull, Matthew D Johnson, Drew Masci, and Carmen Biel. Integrating and visualising diagnostic information for the benefit of learning. *Measuring and visualizing learning in the information-rich classroom*, page 167, 2015.
- Steven Burrows, Iryna Gurevych, and Benno Stein. The eras and trends of automatic short answer grading. *International Journal of Artificial Intelligence in Education*, 25(1):60–117, 2015.
- John B. Carroll. A model of school learning. *Teachers' College Record*, 64: 723–733, 1963.
- Luis Cerezo. Type and amount of input-based practice in cali: The revelations of a triangulated research design. *Language Learning & Technology*, 20(1): 100–123, 2016.
- Mohamed Amine Chatti, Anna Lea Dyckhoff, Ulrik Schroeder, and Hendrik Thüs. A reference model for learning analytics. *International Journal of Technology Enhanced Learning*, 4(5-6):318–331, 2013.
- Maria Chinkina, Madeeswaran Kannan, and Detmar Meurers. Online information retrieval for language learning. In *Proceedings of ACL-2016 System Demonstrations*, pages 7–12, Berlin, Germany, August 2016. Association for Computational Linguistics. <http://anthology.aclweb.org/P16-4002>.
- Inn-Chull Choi. Efficacy of an ICALL tutoring system and process-oriented corrective feedback. *Computer Assisted Language Learning*, 29(2):334–364, 2016.
- Jinho D Choi and Martha Palmer. Fast and robust part-of-speech tagging using dynamic model selection. In *Proceedings of the 50th Annual Meeting of the ACL*, pages 363–367, 2012.
- Konstantina Chrysafiadi and Maria Virvou. Student modeling approaches: A literature review for the last decade. *Expert Systems with Applications*, 40(11):4715–4729, 2013.
- Kenneth Ward Church and William A. Gale. Probability scoring for spelling correction. *Statistics and Computing*, 1(2):93–103, 1991. URL <http://www.springerlink.com/content/vqqt0656502721430/fulltext.pdf>.
- E. H. Cooper and A. J. Pantle. The total-time hypothesis in verbal learning. *Psychological Bulletin*, 68(4), 1967.
- Stephen S Corbett and W Flint Smith. Identifying student learning styles: Proceed with caution! *Modern Language Journal*, 1984.

- Frederik Cornillie, Wim Van Den Noortgate, Kris Van den Branden, and Piet Desmet. Examining focused l2 practice: From in vitro to in vivo. *Language Learning & Technology*, 21(1):121–145, 2017.
- Lee J. Cronbach and Richard E. Snow. *Aptitudes and instructional methods: A handbook for research on interactions*. Irvington, England, 1977.
- Richard Eckart de Castilho and Iryna Gurevych. A broad-coverage collection of portable NLP components for building shareable analysis pipelines. In *Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT (OIAF4HLT)*, pages 1–11, Dublin, Ireland, 2014.
- Ab de Haan and Tinus Oppenhuizen. Speller: A reflexive its to support the learning of second language spelling. *Computers in Human Behavior*, 10(1): 21–31, 1994.
- Edward L Deci, Richard Koestner, and Richard M Ryan. Extrinsic rewards and intrinsic motivation in education: Reconsidered once again. *Review of educational research*, 71(1):1–27, 2001.
- Robert DeKeyser. Implicit and explicit learning. In Catherine Doughty and Michael Long, editors, *The Handbook of Second Language Acquisition*, pages 313–348. Blackwell, 2003.
- Robert DeKeyser. Skill acquisition theory. In Bill VanPatten and Jessica Williams, editors, *Theories in Second Language Acquisition: An introduction*, volume 97113. Routledge, 2007.
- Robert M. DeKeyser. Interactions between individual differences, treatments, and structures in SLA. *Language Learning*, 62:189–200, 2012. ISSN 1467-9922. doi: 10.1111/j.1467-9922.2012.00712.x.
- Ana Díaz Negrillo, Detmar Meurers, Salvador Valera, and Holger Wunsch. Towards interlanguage POS annotation for effective learner corpora in SLA and FLT. *Language Forum*, 36(1–2):139–154, 2010. ISSN 0253-9071. URL <http://purl.org/dm/papers/diaz-negrillo-et-al-09.html>.
- Huizhong Duan and Bo-June Hsu. Online spelling correction for query completion. In *Proceedings of the 20th international conference on World wide web*, pages 117–126, 2011.
- Jean duPlessis, Doreen Solin, Lisa Travis, and Lydia White. Ug or not ug, that is the question: a reply to clahsen and muysken. *Second Language Research*, 3(1):56–75, 1987. doi: 10.1177/026765838700300105.
- Erik Duval. Attention please! learning analytics for visualization and recommendation. In *Proceedings of the 1st international conference on learning analytics and knowledge*, pages 9–17, 2011.

- Thomas Eckes and Rüdiger Grotjahn. A closer look at the construct validity of c-tests. *Language Testing*, 23(3):290–325, 2006.
- Rod Ellis, Helen Basturkmen, and Shawn Loewen. Learner uptake in communicative esl lessons. *Language learning*, 51(2):281–318, 2001.
- Mohammad Ali Elmi and Martha Evens. Spelling correction using context. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 360–364. Association for Computational Linguistics, 1998.
- John A Erickson, Anthony F Botelho, Steven McAteer, Ashvini Varatharaj, and Neil T Heffernan. The automated grading of student open responses in mathematics. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, pages 615–624, 2020.
- Katrin Erk. Vector space models of word meaning and phrase meaning: A survey. *Language and Linguistics Compass*, 6(10):635–653, 2012.
- Wei Fan and Albert Bifet. Mining big data: current status, and forecast to the future. *ACM sIGKDD Explorations Newsletter*, 14(2):1–5, 2013.
- Rebecca Ferguson. Learning analytics: drivers, developments and challenges. *International Journal of Technology Enhanced Learning*, 4(5/6):304–317, 2012.
- David Ferrucci and Adam Lally. UIMA: An architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*, 10(3–4):327–348, 2004. URL <http://journals.cambridge.org/action/displayAbstract?fromPage=online&aid=252253&fulltextType=RA&fileId=S1351324904003523>.
- Michael Flor and Yoko Futagi. On using context for automatic correction of non-word misspellings in student essays. In *Proceedings of the seventh workshop on building educational applications Using NLP*, pages 105–115. Association for Computational Linguistics, 2012.
- Eibe Frank, Mark Hall, Len Trigg, Geoffrey Holmes, and Ian H Witten. Data mining in bioinformatics using weka. *Bioinformatics*, 20(15):2479–2481, 2004.
- Emily R Fyfe and Bethany Rittle-Johnson. Feedback both helps and hinders learning: The causal role of prior knowledge. *Journal of Educational Psychology*, 108(1):82, 2016.
- David Gale and Lloyd S. Shapley. College admissions and the stability of marriage. *American Mathematical Monthly*, 69:9–15, 1962. URL <http://www.econ.ucsb.edu/~tedb/Courses/Ec100C/galeshapley.pdf>.
- Dragan Gašević, Shane Dawson, and George Siemens. Let’s not forget: Learning analytics are about learning. *TechTrends*, 59(1):64–71, Jan 2015. ISSN 1559-7075. doi: 10.1007/s11528-014-0822-x. URL <https://doi.org/10.1007/s11528-014-0822-x>.

- Hanna Gaspard, Isabelle Häfner, Cora Parrisius, Ulrich Trautwein, and Benjamin Nagengast. Assessing task values in five subjects during secondary school: Measurement structure and mean level differences across grade level, gender, and academic subject. *Contemporary Educational Psychology*, 48: 67–84, January 2017. ISSN 0361476X. doi: 10.1016/j.cedpsych.2016.09.003.
- Jeroen Geertzen, Theodora Alexopoulou, and Anna Korhonen. Automatic linguistic annotation of large scale L2 databases: The EF-Cambridge open language database (EFCAMDAT). In *Proceedings of the 31st Second Language Research Forum (SLRF)*. Cascadilla Press, 2013. URL <http://purl.org/ical1/efcamdat>.
- Andrew Gelman. Multilevel (hierarchical) modeling: what it can and cannot do. *Technometrics*, 48(3):432–435, 2006.
- Stefanie Golke, Tobias Dörfler, and Cordula Artelt. The impact of elaborated feedback on text comprehension within a computer-based assessment. *Learning and instruction*, 39:123–136, 2015.
- Thilo Götz and Walt Detmar Meurers. Compiling HPSG type constraints into definite clause programs. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL 95)*, pages 85–91, Cambridge, MA, 1995. MIT. URL <http://purl.org/dm/papers/ac195.html>.
- Thilo Götz and Oliver Suhre. Design and implementation of the uima common analysis system. *IBM Systems Journal*, 43(3):476–489, 2004. URL <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.90.5824&rep=rep1&type=pdf>.
- Michael C Grace, Wu Zhou, Xuxian Jiang, and Ahmad-Reza Sadeghi. Unsafe exposure analysis of mobile in-app advertisements. In *Proceedings of the fifth ACM conference on Security and Privacy in Wireless and Mobile Networks*, pages 101–112, 2012.
- Ken Hale. Endangered languages: On endangered languages and the safeguarding of diversity. *language*, 68(1):1–42, 1992.
- David J Hand. Principles of data mining. *Drug safety*, 30(7):621–622, 2007.
- Elliott Rusty Harold and W. Scott Means. *XML in a Nutshell*. O’Reilly, Sebastopol, CA, 3rd edition, 2004.
- Johannes Hartig and Jana Höhler. *Modellierung von Kompetenzen mit mehrdimensionalen IRT-Modellen*. *Projekt MIRT*, volume 56. Zeitschrift für Pädagogik, 2010.
- John Hattie and Helen Timperley. The power of feedback. *Review of Educational Research*, 77(1):81–112, 2007. doi: 10.3102/003465430298487.

- Neil T Heffernan and Cristina Lindquist Heffernan. The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497, 2014.
- Trude Heift. An interactive intelligent language tutor over the internet. In *Proceedings of ED-MEDIA, ED-TELECOM 98, World Conference on Educational Multimedia and Educational Telecommunications*, volume 2, pages 508–512. Citeseer, 1998.
- Trude Heift. Error-specific and individualized feedback in a web-based language tutoring system: Do they read it? *ReCALL*, 13(2):129–142, 2001.
- Trude Heift. Learner control and error correction in icall: Browsers, peekers and adamantans. *CALICO Journal*, 19(2):295–313, January 2002. URL <https://calico.org/a-428-Learner%20Control%20and%20Error%20Correction%20in%20ICALL%20Browsers%20Peekers%20and%20Adamants.html>.
- Trude Heift. Corrective feedback and learner uptake in CALL. *ReCALL*, 16(2):416–431, 2004.
- Trude Heift. Prompting in CALL: A longitudinal study of learner uptake. *Modern Language Journal*, 94(2):198–216, 2010a. doi: 10.1111/j.1540-4781.2010.01017.x. URL <http://onlinelibrary.wiley.com/doi/10.1111/j.1540-4781.2010.01017.x/abstract>.
- Trude Heift. Developing an intelligent language tutor. *CALICO Journal*, 27(3):443–459, 2010b.
- Trude Heift and Anne Rimrott. Learner responses to corrective feedback for spelling errors in call. *System*, 36(2):196–213, 2008.
- Trude Heift and Mathias Schulze. *Errors and Intelligence in Computer-Assisted Language Learning: Parsers and Pedagogues*. Routledge, New York, 2007.
- Michael Heilman and Nitin Madnani. Ets: Domain adaptation and stacking for short answer scoring. In *Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 275–279, Atlanta, Georgia, USA, June 2013. Association for Computational Linguistics. URL <http://aclweb.org/anthology/S13-2046>.
- Kenny L. Hicks, Jeffrey L. Foster, and Randall W. Engle. Measuring working memory capacity on the web with the Online Working Memory Lab (the OWL). *Journal of Applied Research in Memory and Cognition*, 5:478–489, 2016. doi: 10.1016/j.jarmac.2016.07.010.
- Graeme Hirst and David St. Onge. Wordnet: An electronic lexical database. In Christiane Fellbaum, editor, *Lexical Chains as Representations of Context for the Detection and Correction of Malapropisms*, pages 305–332. MIT Press, 1998.

- Charles D Hockett and Charles F Hockett. *The origin of speech*, volume 203. Scientific American, 1960.
- Andreas Holzinger, Michael D Kickmeier-Rust, and Dietrich Albert. Dynamic media in computer science education; content complexity and learning performance: is less more? *Educational Technology & Society*, 11(1):279–290, 2008.
- Andrea Horbach, Alexis Palmer, and Manfred Pinkal. Using the text to evaluate short answers for reading comprehension exercises. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 286–295, Atlanta, Georgia, USA, June 2013. Association for Computational Linguistics. URL <http://aclweb.org/anthology/S13-1041>.
- Brian W Junker, C Romero, S Ventura, S Viola, M Pechenizkiy, and R Baker. Modeling hierarchy and dependence among task responses in educational data mining. *Handbook of educational data mining*, pages 143–155, 2010.
- Verena Jurik, Alexander Gröschner, and Tina Seidel. Predicting students’ cognitive learning activity and intrinsic learning motivation: How powerful are teacher statements, student profiles, and gender? *Learning and individual differences*, 32:132–139, 2014.
- Daniel Karp, Yves Schabes, Martin Zaidel, and Dania Egedi. A freely available wide coverage morphological analyzer for English. In *Proceedings of the 14th Conference on Computational Linguistics, COLING ’92*, pages 950–955, Stroudsburg, PA, 1992. doi: 10.3115/993079.993105. URL <http://dx.doi.org/10.3115/993079.993105>.
- Kim Kelly, Neil Heffernan, Cristina Heffernan, Susan Goldman, James Pellegrino, and Deena Soffer Goldstein. Estimating the effect of web-based homework. In *International Conference on Artificial Intelligence in Education*, pages 824–827. Springer, 2013.
- Mark D Kernighan, Kenneth W Church, and William A Gale. A spelling correction program based on a noisy channel model. In *Proceedings of the 13th conference on Computational linguistics-Volume 2*, pages 205–210. Association for Computational Linguistics, 1990.
- Bohyun Kim. Responsive web design, discoverability, and mobile challenge. *Library technology reports*, 49(6):29–39, 2013.
- Philipp Koehn. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *AMTA*, pages 115–124, 2004.
- Sotiris Kotsiantis and Dimitris Kanellopoulos. Association rules mining: A recent overview. *GESTS International Transactions on Computer Science and Engineering*, 32(1):71–82, 2006.

- Stephen D Krashen. *The input hypothesis: Issues and implications*. Longman, New York, 1985.
- Justin Kruger and David Dunning. Unskilled and unaware of it: How difficulties in recognizing one's own incompetence lead to inflated self-assessments. *Journal of Personality and Social Psychology*, 7(6):1121–1134, 1999. URL <http://www.apa.org/journals/features/psp7761121.pdf>.
- Kultusministerium. Englisch als erste Fremdsprache [English as a first foreign language]. Bildungsplan des Gymnasiums 2016 [State curriculum for academic track schools 2016], 2016. URL <http://www.bildungsplaene-bw.de/Lde/LS/BP2016BW/ALLG/GYM/E1>. Ministerium für Kultus, Jugend und Sport, Baden Württemberg.
- Barbara Kump, Christin Seifert, Guenter Beham, Stefanie N Lindstaedt, and Tobias Ley. Seeing what the system thinks you know: visualizing evidence in an open learner model. In *Proceedings of the 2nd international conference on learning analytics and knowledge*, pages 153–157, 2012.
- Ronja Laarmann-Quante. Towards a tool for automatic spelling error analysis and feedback generation for freely written german texts produced by primary school children. In *SLaTE*, pages 36–41, 2017.
- James P. Lantoff and Gabriela Appel, editors. *Vygotskian Approaches to Second Language Research, Second Language Learning*. Ablex Publishing Corporation, Norwood, NJ, 1994.
- Diane E Larsen-Freeman. An explanation for the morpheme acquisition order of second language learners. *Language learning*, 26(1):125–134, 1976.
- Claudia Leacock and Martin Chodorow. Combining local context and wordnet sense similarity for word sense identification. In C. Fellbaum, editor, *WordNet, An Electronic Lexical Database*, pages 265–283. MIT Press, 1998.
- Claudia Leacock and Martin Chodorow. C-rater: Automated scoring of short-answer questions. *Computers and the Humanities*, 37:389–405, 2003.
- Stella JH Lee and Susan Bull. An open learner model to help parents help their children. *Technology Instruction Cognition and Learning*, 6(1):29, 2008.
- Geoffrey Leech. Corpus annotation schemes. *Literary and Linguistic Computing*, 8 (4):275–281, 1993.
- Ronald P Leow. *The Routledge Handbook of Second Language Research in Classroom Learning: Processing and Processes*. Routledge, 2019.
- Mu Li, Yang Zhang, Muhua Zhu, and Ming Zhou. Exploring distributional similarity based models for query spelling correction. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 1025–1032. Association for Computational Linguistics, 2006.

- Laura Calvet Liñán and Ángel Alejandro Juan Pérez. Educational data mining and learning analytics: differences, similarities, and time evolution. *International Journal of Educational Technology in Higher Education*, 12(3):98–112, 2015.
- Pilar Prieto Linillos, Sergio Gutierrez, Abelardo Pardo, and Carlos Delgado Kloos. Sequencing parametric exercises for an operating system course. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*, pages 450–458. Springer, 2006.
- Anke Lüdeling, Maik Walter, Emil Kroymann, and Peter Adolphs. Multi-level error annotation in learner corpora. In *Proceedings of Corpus Linguistics*, Birmingham, 2005. URL <http://www.corpus.bham.ac.uk/PCLC/Falko-CL2006.doc>.
- Collin Lynch, Kevin Ashley, Vincent Alevan, and Niels Pinkwart. Defining ill-defined domains; a literature survey. In *Proceedings of the workshop on intelligent tutoring systems for ill-defined domains, held during the 8th international conference on intelligent tutoring systems*, pages 1–10, Jhongli, Taiwan, 2006. URL <http://people.cs.pitt.edu/~collin1/Papers/Ill-DefinedProceedings.pdf#page=7>.
- Roy Lyster and Leila Ranta. Corrective feedback and learner uptake: Negotiation of form in communicative classrooms. *Studies in Second Language Acquisition*, 19 n1:37–66, 1997.
- Andrew Mabbott and Susan Bull. Student preferences for editing, persuading, and negotiating the open learner model. In *International conference on intelligent tutoring systems*, pages 481–490. Springer, 2006.
- Katerina Mangaroska and Michail N Giannakos. Learning analytics for learning design: A systematic literature review of analytics-driven design to enhance learning. *IEEE Transactions on Learning Technologies*, 2018.
- Geoff N Masters. A rasch model for partial credit scoring. *Psychometrika*, 47 (2):149–174, 1982.
- Gerald Matthews, Ian J Deary, and Martha C Whiteman. *Personality traits*. Cambridge University Press, 2003.
- Patrick McGuire, Shihfen Tu, Mary Ellin Logue, Craig A Mason, and Korinn Ostrow. Counterintuitive effects of online feedback in middle school math: results from a randomized controlled trial in assistments. *Educational Media International*, 54(3):231–244, 2017.
- Daniel McNeish, Laura M Stapleton, and Rebecca D Silverman. On the unnecessary ubiquity of hierarchical linear modeling. *Psychological Methods*, 22(1): 114, 2017.

- Daniel M McNeish and Laura M Stapleton. The effect of small sample size on two-level model estimates: A review and illustration. *Educational Psychology Review*, 28(2):295–314, 2016.
- Detmar Meurers. Natural language processing and language learning. In Carol A. Chapelle, editor, *Encyclopedia of Applied Linguistics*, pages 4193–4205. Wiley, Oxford, 2012. URL <http://purl.org/dm/papers/meurers-12.html>.
- Detmar Meurers. Learner corpora and natural language processing. In Sylviane Granger, Gaëtanelle Gilquin, and Fanny Meunier, editors, *The Cambridge Handbook of Learner Corpus Research*, pages 537–566. Cambridge University Press, 2015. <http://purl.org/dm/papers/meurers-15.html>.
- Detmar Meurers, Niels Ott, and Ramon Ziai. On the creation and analysis of a reading comprehension exercise corpus: Evaluating meaning in context.
- Detmar Meurers, Ramon Ziai, Luiz Amaral, Adriane Boyd, Aleksandar Dimitrov, Vanessa Metcalf, and Niels Ott. Enhancing authentic web pages for language learners. In *Proceedings of the 5th Workshop on Innovative Use of NLP for Building Educational Applications (BEA)*, pages 10–18, Los Angeles, 2010. ACL. URL <http://aclweb.org/anthology/W10-1002.pdf>.
- Detmar Meurers, Ramon Ziai, Niels Ott, and Stacey Bailey. Integrating parallel analysis modules to evaluate the meaning of answers to reading comprehension questions. *IJCELL. Special Issue on Automatic Free-text Evaluation*, 21(4):355–369, 2011. URL <http://purl.org/dm/papers/Meurers.Ziai.ea-11.pdf>.
- Detmar Meurers, Kordula De Kuthy, Verena Möller, Florian Nuxoll, Björn Rudzewitz, and Ramon Ziai. Digitale Differenzierung benötigt Informationen zu Sprache, Aufgabe und Lerner. Zur Generierung von individuellem Feedback in einem interaktiven Arbeitsheft. *FLuL – Fremdsprachen Lehren und Lernen*, 47(2):64–82, 2018. URL <http://purl.org/dm/papers/Meurers.DeKuthy.ea-18.pdf>.
- Detmar Meurers, Kordula De Kuthy, Florian Nuxoll, Björn Rudzewitz, and Ramon Ziai. Scaling up intervention studies to investigate real-life foreign language learning in school. *Annual Review of Applied Linguistics*, 39:161–188, 2019a. URL <https://doi.org/10.1017/S0267190519000126>.
- Detmar Meurers, Kordula De Kuthy, Florian Nuxoll, Björn Rudzewitz, and Ramon Ziai. Ki zur lösung realer schulherausforderungen: Interaktive und adaptive materialien im fach englisch. *Schulmanagement-Handbuch*, pages 65–84, 2019b.
- Lisa Michaud and Kathleen McCoy. Capturing the evolution of grammatical knowledge in a CALL system for deaf learners of English. *International Journal of Artificial Intelligence in Education*, 16(1):65–97, 2006.

- Lisa N. Michaud and Kathleen F. McCoy. Supporting intelligent tutoring in call by modeling the user's grammar. In *Proceedings of the 13th Annual International Florida Artificial Intelligence Research Symposium (FLAIRS-00)*, pages 50–54, Orlando, Florida, 2000. URL <http://citeseer.ist.psu.edu/michaud00supporting.html>.
- Guido Minnen, John Carroll, and Darren Pearce. Applied morphological processing of English. *Natural Language Engineering*, 7(3):207–233, 2001.
- Antonija Mitrovic and Brent Martin. Evaluating the effects of open student models on learning. In *International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, pages 296–305. Springer, 2002.
- Michael Mohler, Razvan Bunescu, and Rada Mihalcea. Learning to grade short answer questions using semantic similarity measures and dependency graph alignments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 752–762, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL <http://aclweb.org/anthology/P11-1076>.
- Rolf Molich and Jakob Nielsen. Improving a human-computer dialogue. *Communications of the ACM*, 33(3):338–348, 1990.
- Maureen Murphy and Michael McTear. Learner modelling for Intelligent CALL. In *Proceedings of The 6th International Conference on User Modeling*, pages 301–312, Sardinia, Italy, 1997.
- Noriko Nagata. Robo-Sensei's NLP-based error detection and feedback generation. *CALICO Journal*, 26(3):562–579, 2009. URL <http://purl.org/calico/nagata09.pdf>.
- Benjamin Nagengast, Brigitte M Brisson, Chris S Hulleman, Hanna Gaspard, Isabelle Häfner, and Ulrich Trautwein. Learning more from educational intervention studies: Estimating complier average causal effects in a relevance intervention. *The Journal of Experimental Education*, 86(1):105–123, 2018.
- Melissa M Nelson and Christian D Schunn. The nature of feedback: How different types of peer feedback affect writing performance. *Instructional Science*, 37(4):375–401, 2009.
- Rodney D. Nielsen, Wayne Ward, and James H. Martin. Recognizing entailment in intelligent tutoring systems. *Natural Language Engineering*, 15(4):479–501, 2009. doi: 10.1017/S135132490999012X.
- Joakim Nivre, Jens Nilsson, Johan Hall, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(1):1–41, 2007. URL <http://w3.msi.vxu.se/~nivre/papers/nle07.pdf>.

- Roger Nkambou. Modeling the domain: An introduction to the expert module. In *Advances in intelligent tutoring systems*, pages 15–32. Springer, 2010.
- Terence Odlin. Crosslinguistic influence and conceptual transfer: What are the concepts? *Annual review of applied linguistics*, 25:3–25, 2005.
- Lourdes Ortega. *Understanding Second Language Acquisition*. Hodder Education, London, 2009.
- Niels Ott, Ramon Ziai, and Detmar Meurers. Creation and analysis of a reading comprehension exercise corpus: Towards evaluating meaning in context. In Thomas Schmidt and Kai Wörner, editors, *Multilingual Corpora and Multilingual Corpus Analysis*, Hamburg Studies in Multilingualism (HSM), pages 47–69. Benjamins, Amsterdam, 2012. URL <https://benjamins.com/#catalog/books/hsm.14.05ott>.
- Niels Ott, Ramon Ziai, Michael Hahn, and Detmar Meurers. CoMeT: Integrating different levels of linguistic modeling for meaning assessment. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval)*, pages 608–616, Atlanta, GA, 2013. ACL. URL <http://aclweb.org/anthology/S13-2102.pdf>.
- Benjamin Paaßen, Barbara Hammer, Thomas William Price, Tiffany Barnes, Sebastian Gross, and Niels Pinkwart. The continuous hint factory-providing hints in vast and sparsely populated edit distance spaces. *arXiv preprint arXiv:1708.06564*, 2017.
- Indira Padayachee. Intelligent tutoring systems: Architecture and characteristics. In *Proceedings of the 32nd Annual SACLA Conference*, pages 1–8. Citeseer, 2002.
- Iliana Panova and Roy Lyster. Patterns of corrective feedback and uptake in an adult esl classroom. *TESOL Quarterly*, 36(4):573–595, Winter 2002. URL http://people.mcgill.ca/files/roy.lyster/Panova_Lyster2002_TESOLQ.pdf.
- Zacharoula Papamitsiou and Anastasios A Economides. Learning analytics and educational data mining in practice: A systematic literature review of empirical evidence. *Journal of Educational Technology & Society*, 17(4), 2014.
- Melissa M Patchan, Christian D Schunn, and Richard J Correnti. The nature of feedback: How peer feedback features affect students’ implementation rate and quality of revisions. *Journal of Educational Psychology*, 108(8):1098, 2016.
- Thanaporn Patikorn, Neil T Heffernan, and Ryan S Baker. Assistments longitudinal data mining competition 2017: A preface. In *Proceedings of the Workshop on Scientific Findings from the ASSISTments Longitudinal Data Competition, International Conference on Educational Data Mining*, 2018.

- Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.
- Helge Petersohn. *Data Mining: Verfahren, Prozesse, Anwendungsarchitektur*. Oldenbourg Verlag, 2005.
- Jan L Plass, Dorothy M Chun, Richard E Mayer, and Detlev Leutner. Supporting visual and verbal learning preferences in a second-language multimedia learning environment. *Journal of educational psychology*, 90(1):25, 1998.
- Martí Quixal. *Language Learning Tasks and Automatic Analysis of Learner Language. Connecting FLTL and NLP in the design of ICALL materials supporting effective use in real-life instruction*. PhD thesis, Universitat Pompeu Fabra, Barcelona and Eberhard-Karls-Universität Tübingen, 2012.
- NR Srinivasa Raghavan. Data mining in e-commerce: A survey. *Sadhana*, 30(2-3):275–289, 2005.
- Vyshnavi Malathi Ramesh and NJ Rao. Tutoring and expert modules of intelligent tutoring systems. In *2012 IEEE Fourth International Conference on Technology for Education*, pages 251–252. IEEE, 2012.
- Stephen W Raudenbush and Anthony S Bryk. *Hierarchical linear models: Applications and data analysis methods*, volume 1. sage, 2002.
- Alexander Renkl. Why practice recommendations are important in use-inspired basic research and why too much caution is dysfunctional. *Educational Psychology Review*, 25(3):317–324, 2013.
- Robert Reynolds, Eduard Schaf, and Detmar Meurers. A view of Russian: Visual input enhancement and adaptive feedback. In *Proceedings of the third workshop on NLP for computer-assisted language learning*, NEALT Proceedings Series 22 / Linköping Electronic Conference Proceedings 107, pages 98–112, Uppsala, 2014. ACL. URL <http://aclweb.org/anthology/W14-3508.pdf>.
- Marc Reznicek, Anke Lüdeling, and Hagen Hirschmann. Competing target hypotheses in the Falko corpus: A flexible multi-layer corpus architecture. In Ana Díaz-Negrillo, Nicolas Ballier, and Paul Thompson, editors, *Automatic Treatment and Analysis of Learner Corpus Data*, volume 59, pages 101–123. John Benjamins, 2013.
- Brian Richards. Type/token ratios: What do they really tell us? *Journal of child language*, 14(2):201–209, 1987.
- Anne Rimrott and Trude Heift. Evaluating automatic detection of misspellings in german. *Language Learning and Technology*, 12(3):73–92, October 2008. URL <http://llt.msu.edu/vol12num3/rimrottheift.pdf>.

- Håkan Ringbom. *Cross-linguistic similarity in foreign language learning*, volume 21. Multilingual Matters, 2007.
- Peter Robinson. Task complexity and second language narrative discourse. *Language learning*, 45(1):99–140, 1995.
- Peter Robinson. Task complexity, theory of mind, and intentional reasoning: Effects on L2 speech production, interaction, uptake and perceptions of task difficulty. *International Review of Applied Linguistics in Language Teaching*, 45(3):193–213, 2007.
- Cristobal Romero and Sebastian Ventura. Educational data mining: A survey from 1995 to 2005. *Expert systems with applications*, 33(1):135–146, 2007.
- Cristóbal Romero and Sebastián Ventura. Educational data mining: a review of the state of the art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(6):601–618, 2010.
- Cristobal Romero and Sebastian Ventura. Data mining in education. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 3(1):12–27, 2013. ISSN 1942-4795. doi: 10.1002/widm.1075.
- Cristobal Romero, Sebastian Ventura, Mykola Pechenizkiy, and Ryan SJD Baker. *Handbook of educational data mining*. CRC press, 2010.
- Cristóbal Romero, José Raúl Romero, and Sebastián Ventura. A survey on pre-processing educational data. In *Educational data mining*, pages 29–64. Springer, 2014.
- Rod D Roscoe and Danielle S McNamara. Writing Pal: Feasibility of an Intelligent Writing Strategy Tutor in the High School Classroom. *Journal of Educational Psychology*, 105(4):1010, 2013.
- Shourya Roy, Y Narahari, and Om D Deshmukh. A perspective on computer assisted assessment techniques for short free-text answers. In *International Computer Assisted Assessment Conference*, pages 96–109. Springer, 2015.
- Björn Rudzewitz and Ramon Ziai. CoMiC: Adapting a short answer assessment system for answer selection. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 247–251. ACL, 2015. URL <http://aclweb.org/anthology/S15-2044>.
- Björn Rudzewitz, Ramon Ziai, Kordula De Kuthy, and Detmar Meurers. Developing a web-based workbook for English supporting the interaction of students and teachers. In *Proceedings of the Joint 6th Workshop on NLP for Computer Assisted Language Learning and 2nd Workshop on NLP for Research on Language Acquisition*, pages 36–46, 2017. URL <http://aclweb.org/anthology/W17-0305.pdf>.

- Björn Rudzewitz, Ramon Ziai, Kordula De Kuthy, Verena Möller, Florian Nuxoll, and Detmar Meurers. Generating feedback for English foreign language exercises. In *Proceedings of the 13th Workshop on Innovative Use of NLP for Building Educational Applications (BEA)*, pages 127–136. ACL, 2018. URL <http://aclweb.org/anthology/W18-0513.pdf>.
- Björn Rudzewitz, Ramon Ziai, Florian Nuxoll, Kordula De Kuthy, and Detmar Meurers. Enhancing a web-based language tutoring system with learning analytics. In Luc Paquette and Cristobal Romero, editors, *Joint Proceedings of the Workshops of the 12th International Conference on Educational Data Mining co-located with the 12th International Conference on Educational Data Mining (EDM 2019)*, volume 2592, pages 1–7, Montréal, Canada, 2020. CEUR Workshop Proceedings. URL <http://ceur-ws.org/Vol-2592/paper1.pdf>.
- Simón Eduardo Ruiz Hernández. *Individual differences and instructed Second Language Acquisition: Insights from Intelligent Computer Assisted Language Learning*. Doctoral dissertation, Eberhard-Karls-Universität Tübingen, 2018.
- Richard M Ryan. Control and information in the intrapersonal sphere: An extension of cognitive evaluation theory. *Journal of personality and social psychology*, 43(3):450, 1982.
- Carol Sansone and Judith M Harackiewicz. *Intrinsic and extrinsic motivation: The search for optimal motivation and performance*. Elsevier, 2000.
- Helmut Schmid. A programming language for finite state transducers. In *Proceedings of the 5th International Workshop on Finite State Methods in Natural Language Processing*, pages 308–309, 2005.
- R. Schmidt. The role of consciousness in second language learning. *Applied Linguistics*, 11:206–226, 1990. URL <http://applied.oxfordjournals.org/cgi/reprint/11/2/129.pdf>.
- Mark A Schmuckler. What is ecological validity? a dimensional analysis. *Infancy*, 2(4):419–436, 2001.
- Martin Schrepp, Andreas Hinderks, and Jörg Thomaschewski. Construction of a benchmark for the user experience questionnaire (ueq). *IJIMAI*, 4(4):40–44, 2017.
- John H Schumann. Extending the scope of the acculturation/pidginization model to include cognition. *Tesol Quarterly*, 24(4):667–684, 1990.
- Niall Scater, Alice Peasgood, and Joel Mullan. Learning Analytics in Higher Education. A review of UK and international practice Full report. JISC Report, 2016. Retrieved from <https://osf.io/mp47b/>.
- Tina Seidel. The role of student characteristics in studying micro teaching–learning environments. *Learning Environments Research*, 9(3):253–271, 2006.

- Valerie J Shute. Focus on formative feedback. *ETS Research Report Series*, 2007(1):i-47, 2007.
- George Siemens. Learning analytics: The emergence of a discipline. *American Behavioral Scientist*, 57(10):1380-1400, 2013.
- George Siemens and Ryan SJ Baker. Learning analytics and educational data mining: towards communication and collaboration. In *Proceedings of the 2nd international conference on learning analytics and knowledge*, pages 252-254. ACM, 2012.
- Vanja Slavuj, Ana Meštrović, and Božidar Kovačić. Adaptivity in educational systems for language learning: a review. *Computer Assisted Language Learning*, 30(1-2):64-90, 2017.
- Bryan Smith. The relationship between negotiated interaction, learner uptake, and lexical acquisition in task-based computer-mediated communication. *TESOL Quarterly*, 39(1):33-58, 2005. URL <http://docserver.ingentaconnect.com/deliver/connect/tesol/00398322/v39n1/s2.pdf?expires=1249035785&id=51427763&titleid=6562&accname=Eberhard+Karls+Universitaet+Tuebingen&checksum=7B044D22976F5329C66D3D0A624333C9>. Available through library.
- Karline Soetaert. plot3d: Tools for plotting 3-d and 2-d data. *R package version*, pages 10-2, 2014.
- Reza Soltanpoor and Timos Sellis. Prescriptive analytics for big data. In *Australasian Database Conference*, pages 245-256. Springer, 2016.
- Harold Somers. Translation memory systems. *Benjamins Translation Library*, 35:31-48, 2003.
- Robert A Sottolare, Arthur C Graesser, Xiangen Hu, Andrew Olney, Benjamin Nye, and Anna M Sinatra. *Design Recommendations for Intelligent Tutoring Systems: Volume 4-Domain Modeling*, volume 4. US Army Research Laboratory, 2016.
- Renée Stevens and RO Pihl. The identification of the student at-risk for failure. *Journal of Clinical Psychology*, 38(3):540-545, 1982.
- Pramuditha Suraweera and Antonija Mitrovic. Kermit: A constraint-based tutor for database modeling. In *International Conference on Intelligent Tutoring Systems*, pages 377-387. Springer, 2002.
- Yuichi Suzuki, Tatsuya Nakata, and Robert Dekeyser. Optimizing second language practice in the classroom: Perspectives from cognitive psychology. *The Modern Language Journal*, 103(3):551-561, 2019.
- Yuichi Suzuki, Satoko Yokosawa, and David Aline. The role of working memory in blocked and interleaved grammar practice: Proceduralization of l2 syntax. *Language Teaching Research*, page 1362168820913985, 2020.

- M. Swain. The output hypothesis and beyond: Mediating acquisition through collaborative dialogue. In J. P. Lantolf, editor, *Sociocultural theory and second language learning*, pages 97–114. Oxford University Press, Oxford, 2000.
- Merrill Swain. Communicative competence: Some roles of comprehensible input and comprehensible output in its development. In Susan M. Gass and Carolyn G. Madden, editors, *Input in second language acquisition*, pages 235–253. Newbury House, Rowley, MA, 1985.
- John Sweller. Implications of cognitive load theory for multimedia learning. *The Cambridge handbook of multimedia learning*, pages 19–30, 2005.
- Dustin Tingley, Teppei Yamamoto, Kentaro Hirose, Luke Keele, and Kosuke Imai. mediation: R package for causal mediation analysis. *Journal of Statistical Software*, 59(5):1–38, 2014. URL <http://www.jstatsoft.org/v59/i05/>.
- Judith Tonhauser. Diagnosing (not-)at-issue content. In *Proceedings of Semantics of Under-represented Languages of the Americas (SULA)*, volume 6, pages 239–254, UMass, Amherst, 2012. GLSA.
- Kristina Toutanova and Robert C Moore. Pronunciation modeling for improved spelling correction. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 144–151, 2002.
- Ulrich Trautwein. The homework–achievement relation reconsidered: Differentiating homework time, homework frequency, and homework effort. *Learning and Instruction*, 17(3):372–388, 2007.
- Ulrich Trautwein and Olaf Köller. The relationship between homework and achievement—still much of a mystery. *Educational psychology review*, 15(2): 115–145, 2003.
- Endel Tulving and Daniel L Schacter. Priming and human memory systems. *Science*, 247(4940):301–306, 1990.
- Elisabeth van der Linden. Does feedback enhance computer-assisted language learning? *Computers and Education*, 21(1-2):61–65, 1993. ISSN 0360-1315. doi: [http://dx.doi.org/10.1016/0360-1315\(93\)90048-N](http://dx.doi.org/10.1016/0360-1315(93)90048-N).
- Tyler J VanderWeele and Stijn Vansteelandt. Conceptual issues concerning mediation, interventions and composition. *Statistics and its Interface*, 2(4): 457–468, 2009.
- Kurt Vanlehn, Collin Lynch, Kay Schulze, Joel A Shapiro, Robert Shelby, Linwood Taylor, Don Treacy, Anders Weinstein, and Mary Wintersgill. The andes physics tutoring system: Lessons learned. *International Journal of Artificial Intelligence in Education*, 15(3):147–204, 2005.

- Kurt VanLehn, Collin Lynch, Kay Schulze, Joel A Shapiro, Robert Shelby, Linwood Taylor, Don Treacy, Anders Weinstein, and Mary Wintersgill. The andes physics tutoring system: Five years of evaluations. Technical report, Naval Academy Annapolis MD, 2005.
- Katrien Verbert, Erik Duval, Joris Klerkx, Sten Govaerts, and José Luis Santos. Learning analytics dashboard applications. *American Behavioral Scientist*, 57(10):1500–1509, 2013.
- Matthias von Davier and Young-Sun Lee. *Handbook of diagnostic classification models*. Springer, 2019.
- Lev S. Vygotsky. *Mind in society: The development of higher psychological processes*. Harvard University Press, Cambridge, MA, 1978.
- Jingyun Wang, Takahiko Mendori, and Juan Xiong. A language learning support system using course-centered ontology and its evaluation. *Computers & Education*, 78:278–293, 2014.
- Lydia White. Against comprehensible input in the input hypothesis and the development of second language competence. *Applied Linguistics*, 8(2):95–110, 1987.
- Lydia White, Nina Spada, Patsy M. Lightbown, and Leila Ranta. Input enhancement and L2 question formation. *Applied Linguistics*, 12(4):416–432, 1991.
- Mark Wilson, Paul De Boeck, and Claus H Carstensen. Explanatory item response models: A brief introduction. *Assessment of competencies in educational contexts*, pages 91–120, 2008.
- Juan-Diego Zapata-Rivera and Jim E Greer. Interacting with inspectable bayesian student models. *International Journal of Artificial Intelligence in Education*, 14(2):127–163, 2004.
- Mi Zhang, Jie Tang, Xuchen Zhang, and Xiangyang Xue. Addressing cold start in recommender systems: A semi-supervised co-training algorithm. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 73–82. ACM, 2014.
- Yiran Zhang, Rita Hedo, Anna Rivera, Rudolph Rull, Sabrina Richardson, and Xin M Tu. Post hoc power analysis: is it an informative and meaningful analysis? *General Psychiatry*, 32(4), 2019.
- Ramon Ziai. *Short Answer Assessment in Context: The Role of Information Structure*. PhD thesis, Eberhard-Karls Universität Tübingen, 2018. URL <http://hdl.handle.net/10900/81732>.

- Ramon Ziai and Detmar Meurers. Automatic focus annotation: Bringing formal pragmatics alive in analyzing the Information Structure of authentic data. In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 117–128, New Orleans, LA, 2018. ACL. URL <http://aclweb.org/anthology/N18-1011.pdf>.
- Ramon Ziai, Niels Ott, and Detmar Meurers. Short answer assessment: Establishing links between research strands. In *Proceedings of the 7th Workshop on Innovative Use of NLP for Building Educational Applications (BEA-7) at NAACL-HLT 2012*, pages 190–200, Montreal, 2012. URL <http://aclweb.org/anthology/W12-2022.pdf>.
- Ramon Ziai, Björn Rudzewitz, Kordula De Kuthy, Florian Nuxoll, and Detmar Meurers. Feedback strategies for form and meaning in a real-life language tutoring system. In *Proceedings of the 7th Workshop on Natural Language Processing for Computer-Assisted Language Learning (NLP4CALL)*, pages 91–98. ACL, 2018. URL <http://aclweb.org/anthology/W18-7110>.
- Ramon Ziai, Florian Nuxoll, Kordula De Kuthy, Björn Rudzewitz, and Detmar Meurers. The impact of spelling correction and task context on short answer assessment for intelligent tutoring systems. In *Proceedings of the 8th Workshop on NLP for Computer Assisted Language Learning*, pages 93–99, Turku, Finland, September 2019. ACL. URL <https://www.aclweb.org/anthology/W19-6310.pdf>.
- Nicole Ziegler, Detmar Meurers, Patrick Rebuschat, Simón Ruiz, José Luis Moreno Vega, Maria Chinkina, Wenjing Li, and Sarah Grey. Interdisciplinary research at the intersection of CALL, NLP, and SLA: Methodological implications from an input enhancement project. *Language Learning*, 67:209–231, 2017. doi: 10.1111/lang.12227.