Automated Feedback Generation for Learner Modeling in Intelligent Tutoring Systems

by

Chang Lu

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Measurement, Evaluation, and Data Science

Department of Educational Psychology
University of Alberta

Abstract

Feedback is essential for knowledge acquisition, but there is a paucity of automated feedback generation frameworks in intelligent tutoring systems (ITSs) that facilitate and scaffold students' learning across domains. This study introduces a novel framework for generating templated-based feedback to tackle the issue of automated feedback generation for different learner models. Specifically, it (1) implements several learner modeling algorithms including IRT, BKT, DKT-RNN, and DKT-RNN-LSTM; (2) devises and implements DKT-CI (i.e., DKT-RNN-LSTM with Contextualized Information) to estimate learners' skill mastery states using both product data and process data; (3) compares these algorithms' prediction accuracy, interpretability, and applicability on three datasets with various sizes extracted from different ITSs; and (4) introduces a framework to automatically generate template-based feedback on learners' performance for the output of these learner models. Results revealed that (1) BKT and IRT outperformed DKT on smaller datasets, whereas DKT-CI outperformed other models on larger datasets; (2) for BKT, the proposed template-based feedback generation could produce KC-dependent feedback based on learner performance and expert-derived thresholds; (3) for DKT, the feedback-generation method could produce adaptive feedback for all KCs at every time step and plot individuals' knowledge transfer, thus being more suitable for individualized formative tutoring. Implications regarding context-specific automated feedback provision for interactive digital learning systems are discussed. Findings from the present research facilitate the understanding of students' learning behaviour and the dynamic knowledge acquisition process on different knowledge components in the ITSs and inform decision making on when and how to provide feedback in these systems.

Acknowledgments

First, I would like to express my gratitude to my supervisors, Dr. Maria Cutumisu and Dr. Mark Gierl, for their constant motivation and patience to guide me through my Ph.D. study. They mentored me to be an independent researcher, and their support has always been an encouragement for me to push myself to the next level. I cannot express my appreciation enough for all they have provided for me. Thank you, Maria, for bringing me in so many amazing projects, helping me outline the path to an academic career, teaching me what is expected from a researcher, and investing countless hours throughout these years to painstakingly foster my abilities on research, critical thinking, writing, and beyond. Thank you for inspiring me and raising me up when I felt lost during this journey. I will never have come to this point without your guidance and support. Thank you, Mark, for always being supportive and pointing out the direction for me.

I would also like to thank my examining committee members Dr. Lili Mou, Dr. Luiza Antonie, Dr. Ying Cui, and Dr. Veronica Smith for providing insightful comments for my dissertation and Dr. George Buck for hosting my defense.

Moreover, I want to thank my friends. Dr. Qi Guo and Dr. Fu Chen studied with me and were always willing to offer help. I also want to thank other CRAMERs for being inspiring friends. Special thanks to my friends Shuran Meng and Qin Wang for being my side along the way.

Finally, I would like to thank my parents for their unconditional love, especially my mom for trusting, comforting, and encouraging me on this journey.

Table of Contents

List of Figures

FIGURE PAGE

List of Tables

TABLE PAGE

List of Abbreviations and Acronyms

ANN	Artificial Neural Network
AUC	Area Under the Curve
BKT	Bayesian Knowledge Tracing
CGMH	Constrained Sentence Generation by Metropolis-Hastings Sampling
CNN	Convolutional Neural Network
DKT	Deep Knowledge Tracing
EDM	Educational Data Mining
FIT	Feedback Intervention Theory
GRU	Gated Recurrent Units
HCI	Human Computer Interaction
HMM	Hidden Markov Model
ICT	Information and Communication Technologies
IRT	Item Response Theory
ITS	Intelligent Tutoring System
KC	Knowledge Components
LMS	Learning Management System
LSTM	Long Short-Term Memory
MCMC	Markov Chain Monte Carlo
MOOC	Massive Open Online Course
NLL	Negative Likelihood
NLP	Natural Language Processing
OULA	Open University Learning Analytics
PFA	Performance Factor Analysis
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic

SVM	Support Vector Machine
TAF-ClaF	Technologies for Automated Feedback – Classification
TSF	Time Series Forest
VLE	Virtual Learning Environments

Chapter 1 Introduction

Online education was brought to the forefront of higher education by the outbreak of the coronavirus disease (COVID-19) in late 2019. China initiated a nationwide massive migration from traditional face-to-face teaching to fully online courses in 1,291 universities, enforced by the government's policy of "non-stop teaching and learning" in early 2020 (Bao, 2020; Sun et al., 2020). Followed by the rapid spread of the pandemic across the globe, 123 countries in Asia, North America, South America, Europe, Africa, and the Middle East have announced school closures and immediate transformations to online education as of June 21, 2020; meanwhile, more than one billion learners have been affected by the pandemic, which is equivalent with 62.3% of the total enrolled learners (UNESCO, 2020). Online education is becoming the main theme for most learners around the world.

Online education has strong roots in both informal online learning platforms and formal higher education. The proliferation of information and communication technologies (ICT) has supported the advent of open learning platforms, including Massive Open Online Courses (MOOCs), Coursera, and other virtual learning environments (VLE). Great efforts have also been made by post-secondary institutions to adopt Learning Management Systems (LMSs) and promote formal online education over the last decades (Allen & Seaman, 2008; Bradford et al., 2007; O'Neill et al., 2004; Patel & Patel, 2005). Macfadyen and Dawson (2012) reported that more than 90% of higher-education institutions in the USA have made significant investments in the implementation of LMSs, such as Moodle, Canvas, and Blackboard, since the late 1990s. Similarly, in Great Britain, 85% of higher education institutions have adopted VLEs since 2003 (Ferguson, 2012). Unlike open online learning platforms, LMSs adopted by higher education are often used to deliver course materials as a supplement to traditional formal instruction (Zhou et al., 2020), whereas most interactions between students and instructors occur in class.

On the other hand, open learning platforms such as Intelligent Tutoring Systems (ITSs) make asynchronous learning materials and instructional videos accessible to learners without time or space limitations (Washington, 2019; Zacharis, 2015). ITSs are designed to provide personalized hints and to automate the adaptive learning process of knowledge components (KCs), a generic term for skills, concepts, procedures, and strategies (Pardo et al., 2019). With the availability of large volumes of log data generated from learners' interactions with ITSs, many studies attempted to develop models to estimate individuals' skill mastery state, track their learning processes, and diagnose their strengths and weaknesses on KCs (Macfadyen et al., 2014). Those models were later defined as *learner models* (or *student models*), as they constitute a structured representation of a learner's knowledge, misconceptions, or difficulties (Bull, 2004). The tutoring and assessment within ITSs could help students practice skills outside the school, where feedback from human experts might not be readily available. However, a recent review of automated feedback systems revealed that the weakest links of ITSs are pedagogical policy and feedback provision (Deeva et al., 2021; Gervet et al., 2021). Although providing feedback is crucial for scaffolding students to improve performance (Hattie & Timperley, 2007), research on how learning technologies effectively estimate learners' skills and automatically generate individualized feedback is lagging (Deeva et al., 2021; Maniktala et al., 2020; Sedrakyan et al., 2020; Weimer, 2002). Specifically, discussions of automated feedback systems are still ongoing regarding when to provide feedback (immediate vs. delayed), how to provide feedback (summative vs. formative, adaptive vs. non-adaptive, expert-derived vs. data-driven), and how to assemble the feedback content (generic vs. specific, KC based vs. item based).

Some of the earlier attempts of implementing feedback within ITSs focused on providing real-time online tutoring by humans in those environments (Merrill et al., 1992;

Heffernan & Koedinger, 2002). Findings show that human tutoring is effective in improving students' performance, but it is time- and labor-consuming, and not applicable to large-scale practice and open-ended platforms.

In the last decade, many studies focused on implementing automated feedback systems that can generate item-based hints or holistic summative feedback based on student product data (i.e., student direct responses to the items) in structured domains (e.g., math and algebra) or generating corrective feedback using natural language processing (NLP) techniques in unstructured domains (e.g., essay writing and programming). However, few studies focused on providing timely and specific adaptive feedback at the skill level (i.e., KC level) that would prompt student reflection based on both product data and process data (i.e., the duration of the tasks, action took, number of attempts, number of hints requested) collected from the ITSs (Aleven et al., 2006; Barnes & Stamper, 2010; Shatnawi et al., 2014). With the advancement of various learner models, researchers can trace and update learners' KC profiles using different prediction models based on both product data and process data. Multiple output representations of learner KC mastery are available to inform instructors and students on their status of knowledge acquisition and transfer. Meanwhile, the advances in natural language processing techniques motivate the implementation of human-machine communications that bridge learner modeling and automated feedback generation.

Research Purpose

We propose a framework of automated feedback generation for different learner models at the skill level for structured domains. Specifically, we implement an Item Response Model (IRT), Bayesian Knowledge Tracing (BKT), Deep Knowledge Tracing (DKT), and its variants with Long Short-Term Memory (LSTM; Hochreiter & Schmidhuber, 1997) and with contextualized information. We compare them on input representation, output representation, model prediction accuracy, interpretability, and applicability in various

educational assessment contexts. Further, we describe an automated feedback-generation approach and demonstrate the template-based feedback generated for each learner model. This dissertation is guided by the following research questions:

1. To what extent do the learner models perform accurate and interpretable estimations of students' performance? *What are the predictive accuracies, input representations, output representations, and characteristics of the different learner models?*
2. To what extent does the feedback generation method produce fluent and semantically related feedback? *Can the feedback generation method create a variety of feedback templates that are grammatically correct and semantically related?*
3. To what extent is the proposed framework of Automated Feedback Generation for Learner Models feasible for structured knowledge domains? *How does the proposed feedback generation method fit into different learner models? What format and information does the generated feedback provide for learners?*

Research Contributions

This dissertation makes the following contributions: it (1) implemented several learner modeling algorithms including IRT, BKT, DKT-RNN, and DKT-RNN-LSTM; (2) devised and implemented DKT-CI (i.e., DKT-RNN-LSTM with Contextualized Information) to estimate learners' skill mastery states using both product data and process data; (3) compared these algorithms' prediction accuracy, interpretability, and applicability on three datasets with various sizes extracted from different ITSs; and (4) introduced a framework to automatically generate template-based feedback on learners' performance for the output of these learner models.

Organization of the Dissertation

The rest of the document is organized as follows. First, the theoretical framework of the dissertation is presented. Then, studies on educational data mining (EDM) and different

learner models are reviewed. Further, related work on automated feedback generation is examined. Fourth, the potential research gaps from previous studies are identified. Fifth, a novel learner-modeling and feedback-generation approach is introduced. Lastly, discussions are made on the importance of understanding the test contexts and interpreting the outputs of learner modeling and feedback generation. Suggestions are provided regarding learner model and feedback generation selections that cater to different educational settings.

Chapter Summary

Chapter 1 introduced the context of the problem tackled in this research: technology-enhanced learning, tutoring, and assessments make big educational data in the digital learning environments available (i.e., students' interactions with the systems and artefacts recorded). Then, the state-of-art studies on learner modeling and automated feedback generation in the digital learning environments were reviewed, the gaps in current approaches were identified, and a novel solution to this problem was proposed. Lastly, this chapter outlined the main research questions and the organization of the remaining document.

Chapter 2 Theoretical Framework

The Definition of Feedback

In education, feedback is regarded as a key concept for learning and performance (Shute, 2008). Feedback is defined as information provided by an agent regarding one's performance or understanding to improve knowledge and skills (Hattie & Timperley, 2007), or learning processes where the gaps between actual performance and intended performance are identified and provided for learners to improve on (Carless et al., 2011; Molley & Boud, 2014).

Sadler (1989) stated that feedback should deliver messages related to the process of learning that specifies the discrepancies between what is understood and the pre-defined goal. Winnie and Butler (1994, p. 5740) noted that "feedback is information with which a learner can confirm, add to, overwrite, tune, and restructure information in memory, whether that information is domain knowledge, meta-cognitive knowledge, beliefs about self and tasks, or cognitive tactics and strategies". Kluger and DeNisi (1996, 1998) defined feedback as the information regarding one's performance and one of widely used psychological interventions. Hattie and Timperley (2007) conceptualized feedback as a "consequence" of performance and positioned feedback within a continuum of instruction and feedback. To fulfil the instructional purpose of feedback, it can be achieved through cognitive processes (e.g., increased effect, motivation, and engagement) and affective processes (e.g., restructuring understanding and confirming information). Shute (2008, p. 154) defined feedback "as information communicated to the learner that is intended to modify his or her thinking or behaviour for the purpose of improving learning". Molley and Boud (2014) described feedback as a key process of learning, where the gap between actual performance and the goal of performance is identified and provided for learners. To summarize, feedback can be either regarded as information that is related to specific learning tasks and goals, or a learning

process that communicates the gaps between one's actual performance and the intended performance.

Feedback and Learning Theories

Feedback is regarded a key concept for learning and teaching, which has gained growing interests by researchers and practitioners alike on exploring the development, administration, and effectiveness of feedback under different educational contexts (e.g., Anderson, Conrad, & Corbett, 1989; Black & Wiliam, 1998; Carless, 2006; Corbett, Conrad, & Anderson, 1994; Hattie & Timperley, 2007; Kluger & DeNisi, 1996; Moreno, 2004; Narciss & Huth, 2004; Shute, 2008). Previous research on feedback was mainly grounded on several learning theories—behaviourism, cognitivism, and social constructivism—to represent different processes of how feedback impacts learning and performance. However, social constructivism is generally embedded in traditional classroom-based contexts and not directly related to the present work; thus, it is excluded from the discussion.

Behaviourism regards providing feedback as a linear process, where teachers guide students step by step to achieve the goals of the predefined curriculum (Thurlings et al., 2013). It primarily focuses on how teachers monitor and manipulate students' behaviour through positive feedback to reinforce the behaviour and negative feedback to punish and correct the behaviour (Skinner, 1968). In the behaviourist perspective, teachers are seen as the action takers and students as recipients of instructions. Therefore, the teacher's role takes the lead and functions as the determinant of the effectiveness of feedback, whereas students follow what teachers request and produce an outcome of feedback (Butler & Winne, 1995; Duchaine, Jolivette, & Fredrick, 2011). However, the behaviourist view of feedback was later contested by other researchers (Boud & Molloy, 2013; Carless et al., 2011; Price et al., 2010; Shute, 2008). They criticized that the psychological behaviourism oversimplifies the process of providing feedback by interpreting learning as a unilateral and unidirectional process led

by educators and the curriculum (Biggs, 1993), without acknowledging "the active role of the learner in co-producing knowledge" (Molloy & Boud, 2014, p. 3) and the learners' capabilities on self-evaluating and self-regulating (Carless et al., 2011; Hattie & Timperley, 2007; Price et al., 2010).

Cognitivism also considers feedback as a linear process and interprets learning as human information processing rather than observable behaviour (Newell & Simon, 1972; Shuell, 1986). As opposed to behaviourists, cognitivists shift the focus from teachers to students. Teachers use feedback to guide students to receive, use, and reflect on knowledge so that students can process and decode information internally and finalize it in the learning outcomes. Therefore, cognitivists focus more on the internal mental processes in learning, the interconnections among notions of knowledge, and conscious representations of the real world rather than on the behaviour-based exhibition of learning. Most technology-enhanced learning is based on cognitivist learning theory. The cognitive approach identifies the cognitive domain in a hierarchy of learning objectives, including remembering, understanding, applying, analyzing, evaluating, and creating (Anderson & Bloom, 2001; Bloom, 1956). It operationalizes the cognitive domain into external factors, and it embeds them into technology-based learning, such as intelligent tutoring systems and adaptive learning environments (Roll et al., 2011). Among the digital learning systems, knowledge acquisition is broken down into manageable steps. Then, the machine provides instructions or more practices, and it leads learners to the most appropriate step to achieve the goal of pre-defined learning outcomes. To conclude, cognitivists hold a concept of mind (Fontana, 1981), and view feedback as information that is processed and internalized by learners.

Models of Feedback

Several models of feedback underlying different learning theories were proposed to model different types of feedback within various educational contexts. Specifically, we

discussed the *Instructive Feedback* by Werts, Wolery, Holcombe, and Gast (1995), the *Feedback Intervention Theory (FIT)* by Kluger and DeNisi's (1996), the *Four Levels of Feedback Framework* by Hattie and Timperley (2007), and the *Framework of Formative Feedback* by Shute (2008).

Instructive Feedback. Werts et al. (1995, p. 55) claimed that feedback should be instructional and that "instructive feedback is a method of presenting extra, non-target stimuli in the consequent events of instructional trials (e.g., during praise statements)". This framework is rooted in a behaviourist approach. More specifically, they categorized instructive feedback into three categories: parallel, expansion, and novel. Parallel instructive feedback stimuli refer to the repetition of previous instruction. For example, teachers use numbers to teach numbers. Expansive instructional feedback extends the previous instruction from a specific content to new target stimuli. For example, Gast et al. (1994) used spelling of words as instructive feedback to teach the target stimuli sight words. Novel instructional feedback uses conceptually unrelated instruction and targets to a new study domain. For example, Werts et al. (1993) used social studies (i.e., instructive feedback) to teach students math equations (i.e., target stimuli). The common goal of all three categories is to stimulate students to learn target stimuli through a series of instructions and trials. However, if additional instructions are provided, students are not required to respond to the stimuli or reinforced by feedback. Werts et al. (1995) also examined key factors of instructive feedback from previous literature including (1) student demographic variables (e.g., age, gender); (2) diagnosis of disabilities (e.g., mental retardation, autism, seizure disorders, developmental delays, learning disabilities); (3) procedural parameters (e.g., the type of target behaviour, location of the instruction, the type of instructor, instructional grouping, and the type of instructional strategy used); and (4) presentation variables (e.g., the feedback is presented verbally or visually). They concluded that students acquire and maintain performance on the

response-prompting instructive feedback behaviours among parallel, expensive, and novel categories; thus, teachers are encouraged to incorporate more direct instructional feedback in their pedagogical practices.

Feedback Intervention Theory (FIT). FIT is one of the most widely recognized feedback theories grounded in cognitivism. Kluger and DeNisi (1996) conducted a meta-analysis that included 607 cases (effect sizes) and 23,663 observations. They found that FIs improved performance on average (d = .41), but more than a third of studies reported that FI reduced performance, which cannot be explained by sampling errors or existing theories. Following on from this finding, they proposed the Feedback Intervention Theory (FIT). FIT assumes that feedback affects performance by changing the locus of a learner's attention among three hierarchical levels of control including task learning, task motivation, and meta-task processes. The lower in the hierarchy the FI-induced locus of attention is, the stronger is the benefit of an FI for performance. More specifically, FIT consists of five basic arguments: (1) behaviour is regulated by comparisons of feedback to goals or standard, (2) goals or standards are organized hierarchically, (3) attention is limited and therefore only feedback standard gaps (i.e., discrepancies between actual and desired performance) that receive attention actively participate in behaviour regulation, (4) attention is normally directed to a moderate level of the hierarchy, and (5) FIs change the locus of attention and therefore affect behaviour. These arguments are interdependent, and each consecutive argument is built on the preceding argument. Kluger and DeNisi (1998) later further discussed how FIs might have both positive and negative effects on performance. They drew on three theoretical constructs of control theory including the regulation of feedback-standard discrepancies, the locus of attention, and the task complexity (Carver & Scheier, 1981). In addition, they related them to the FIT assumptions as follows: (1) behaviour is regulated by comparisons of feedback with goals or standards (and identification of gaps between the two); (2) attention is

limited, and only those feedback-standard gaps that receive attention actively participate in behaviour regulation; and (3) FIs change the locus of attention and therefore affect behaviour. By connecting between control theory and FIT, they made the following arguments (1) "behaviour is regulated through the control of discrepancies or errors in the system"; (2) changing locus of attention and "knowing where attention is directed provides a better position to predict FIs' effects on performance"; and (3) task properties determine how FI affect performance (Kluger & DeNisi, 1998, p. 69). To conclude, the FI functions as a double-edge sword that may support and hinder performance, depending on conditions.

Four Levels of Feedback Framework. Hattie and Timperley (2007) conceptualize feedback as a tool that reduces discrepancies between current and desired performance. They stated that effective feedback must answer three major questions asked by a teacher and/or by a student: Where am I going? (What are the goals?), How am I going? (What progress is being made toward the goal?), and Where to next? (What activities need to be undertaken to make better progress?). These questions correspond to notions of feed-up, feed-back, and feed-forward. The level at which the feedback operates partly determines whether a teacher and/or student can effectively answer the three questions above. Therefore, they proposed a model that addresses four levels of feedback, namely, *task*, *process*, *self-regulation*, and *self*. Feedback has differing effects across these levels. The *task* level and the *process* level refer to how well the task is being performed; the *self-regulation* level describes how students monitor, direct, and regulate actions toward the learning goal; and the *self* level describes the affective evaluations about the student, which are usually not related to the task itself (Brophy, 1981). Hattie and Timperley (2007) argued that effective feedback should be focused on the *task* level and the *process* level, rather than the *self* level.

Framework of Formative Feedback. Shute (2008) proposed a framework for formative feedback, in which several key components of effective formative feedback were

addressed and elaborated including the purpose (i.e., directive or facilitative feedback), cognitive mechanisms (i.e., how the formative feedback motivates learning efforts), feedback specificity (i.e., levels of information provided), features (i.e., verification and elaboration), complexity and length, and timing (i.e., immediate vs. delayed feedback). Furthermore, Shute (2008) reviewed existing frameworks of feedback and provided guidelines to generate effective formative feedback. Shute (2008) concluded that formative feedback should be non-evaluative, supportive, timely, and specific to promote learning. First, in contrast to behaviorists and social constructivists who suggest feedback be only positive and constructive, Shute (2008) suggested feedback should be neutral, unbiased, and be either positive or negative. Second, timing is sensitive to task difficulties. For difficult tasks, immediate feedback is more effective. For easy tasks, delayed feedback yields better learning outcomes. Immediate feedback helps retain procedural or conceptual knowledge, whereas delayed feedback promotes transfer of learning. Third, feedback should be task-directed and goal-oriented to help learners to identify and close the gaps between the intended and actual performance. Thus, the feedback should be specific, detailed, and clear at task levels without being too long. Fourth, learner characteristics such as ability levels and motivation should also be considered when providing formative feedback. For example, for high-achieving students, it is suggested to use delayed facilitative feedback with verification. For low-achieving students, it is suggested to use immediate corrective feedback with elaboration. Table 1 summarizes the feedback guidelines categorized by Shute based on three dimensions including learning, timing, and student characteristics. To sum up, Shute (2008, p. 175) highlighted the importance of formative feedback for learning in technology-assisted instructions and called for more investigation on three aspects of feedback. The first aspect is 'motive', which discusses whether the students need the feedback. The second aspect is

'opportunity', which evaluates the timing of the feedback. The last aspect is 'means', which explores whether the students are able and willing to use the feedback.

Table 1

Guidelines of Formative Feedback Adapted from Shute (2008)

Dimension	**Prescriptions**
Learning	Focus feedback on the task, not on the learner.
	Provide elaborated feedback to enhance learning.
	Present elaborated feedback in manageable units.
	Be specific and clear with the feedback message.
	Keep feedback as simple as possible but no simpler (based on learner needs and instructional constraints).
	Reduce uncertainty between performance and goals.
	Give unbiased, objective feedback, written or via a computer.
	Promote a "learning" goal orientation via feedback.
	Provide feedback after learners have attempted a solution.
	Do not give normative comparisons.
	Be cautious about providing overall grades
	Do not present feedback that discourages the learner or threatens the learner's self-esteem.
	Use "praise" sparingly, if at all.
	Try to avoid delivering feedback orally.
	Do not interrupt the learner with feedback if the learner is actively engaged.

	Avoid using progressive hints that always terminate with the correct answer.
	Do not limit the mode of feedback presentation to text.
	Minimize the use of extensive error analyses and diagnosis.
Timing	Design timing of feedback to align with the desired outcome.
	For difficult tasks, use immediate feedback.
	For relatively simple tasks, use delayed feedback.
	For retention of procedural or conceptual knowledge, use immediate feedback.
	To promote transfer of learning, consider using delayed feedback.
Learner characteristics	For high-achieving learners, consider using delayed feedback.
	For low-achieving learners, use immediate feedback.
	For low-achieving learners, use directive (or corrective) feedback.
	For high-achieving learners, use facilitative feedback.
	For low-achieving learners, use scaffolding.
	For high-achieving learners, verification feedback may be sufficient.
	For low-achieving learners, use correct response and some kind of elaboration feedback.
	For learners with low learning orientation (or high performance orientation), give specific feedback.

Chapter Summary

Chapter 2 provided an overview of the learning theories underlying feedback and several feedback frameworks. Behaviourist and cognitivist approaches to feedback were discussed. In addition, four frameworks of feedback based on different learning theories were

presented detailing what information is delivered by feedback, how it is provided, the cognitive mechanism, and the related instructional activities. The next chapter describes a comprehensive review of empirical studies in feedback and learning within technology-enhanced learning environments.

Chapter 3 Literature Review

This chapter starts with an introduction to technology-enhanced education and digital learning environments, followed by a comprehensive review of educational data mining techniques and learner models that were commonly used for identifying student learning status and providing automated feedback. Next, a survey of current trends of automated feedback generation within digital learning environments is discussed, with a special focus on the data-driven approach to feedback generation based on student data. Lastly, the gaps and limitations of previous studies are summarized.

Technology-Enhanced Education

The current learning technology climate prompted the transitions of blended courses to exclusively online delivered courses in education (Hannafin & Land, 1997). In addition, many computer-based tutoring and assessment systems are available for students to practice knowledge and skills without time and space limitation. Thus, it is especially important to understand students' behaviours while they are learning in the digital learning systems and to provide individualized feedback. Also, the availability of large volumes of log events generated by the learning systems as well as the advancement of educational data mining (EDM) techniques prompted many studies designed to observe and cluster students' online learning behaviours (Becker et al., 2006; Cantabella et al., 2019; Geigle & Zhai, 2017; Khalil & Ebner, 2017; Papamitsiou & Economides, 2014; Peña-Ayala, 2014; Shi et al., 2015), predict students' academic achievement or dropout rates (Baker & Inventado, 2014; Baker, Lindrum, Lindrum, & Perkowski, 2015; Baker & Yacef, 2009; Conijn et al., 2016; Gardner & Brooks, 2018; Gašević et al., 2016; Jayaprakash, Moody, Lauría, Regan, & Baron, 2014; Kim et al., 2018; Poornima & Pushpalatha, 2019; Smith, Lange, & Huston, 2012; Xing et al., 2016; Zacharis, 2015), conduct sequential mining using features extracted from log files (Hung & Crooks, 2009; Juhaňák, Zounek, & Rohlíková, 2019), evaluate students' real-time

online work and activities (Liu et al., 2018), and detect at-risk students at early stages to inform decision making (Chung & Lee, 2019; He et al., 2015; Lu et al., 2017; Mao et al., 2018; Xing & Du, 2019). In short, a great number of studies have been conducted to understand learners' online learning behaviours and their relationship with performance on digital learning systems using a variety of data-driven approaches.

Predictions of Learner Performance: Static versus Sequential Models and Features. Many previous studies on log-file analysis have been conducted to examine the relationships between online learning activities and academic achievement (Bousbia & Belamri, 2014; Dutt, Ismail, & Herawan, 2017; Jovanovic, Gasevic, Dawson, Pardo, & Mirriahi, 2017; Juhaňák, Zounek, & Rohlíková, 2019). Research on prediction of academic performance using log data generated by digital learning systems heavily relies on feature engineering, data representation, and analytical methods. Previous studies mainly adopted two types of behaviour-based features (static and sequential) and two types of predictive models (static and sequential).

Behaviour-based static features commonly refer to the aggregated click frequencies related to logins, files accessed, assignment submissions, practice attempts and forum postings over a certain period or the total time spent on different modules embedded in the systems. *Static predictive models* are classification or regression models that predict academic achievement indicators (e.g., assignment scores, midterm exam scores, final exam scores, project performance, and dropout; Bousbia & Belamri, 2014; Dutt, Ismail, & Herawan, 2017; Jovanovic, Gasevic, Dawson, Pardo, & Mirriahi, 2017; Zacharis, 2015). Some of the prevalent methods employed to predict academic achievement include multiple linear regressions (Agudo-Peregrina et al., 2014; Ashenafi et al., 2015; Zacharis, 2015), decision trees (Hung & Crooks, 2009; Topîrceanu & Grosseck, 2017), random forests (Liu, Wang, Benachour, & Tubman, 2018), support vector machines (Ifenthaler &

Widanapathirana, 2014; Kloft et al., 2014), and artificial neural networks (Olivé et al., 2020; Zhang & Jiang, 2018). Generally, static models rely on statistical assumptions, but they are widely used in the previous studies for their adequate interpretability.

On the other hand, *behaviour-based sequential features* extracted in the related research are typically in the format of time-series sequence on one or more components (e.g., Hassan et al., 2019; Liu et al., 2018; Liu et al., 2018). *Sequential predictive models* are analytical models that can take as input and process high-dimensional sequential features, such as Recurrent Neural Networks (RNN; Mikolov et al., 2010), and Long Short-Term Memory (LSTM; Hassan et al., 2019; Liu et al., 2018), which is a special RNN variant.. Sequential models can capture the temporal variations and heterogeneity of the evolving online behaviours within learners. Thus, they can more accurately model the temporal behavioural information in log events (Li & Zhao, 2020). For example, if two students both reached 100 clicks on the *Quiz* component in the LMS during the term, static models would fail to capture students' potentially different problem-solving patterns (e.g., one student could have practiced more at the beginning of the term, whereas the other student could have practiced the most at the end of the term). Sequential models detect their differences in terms of temporal behavioural patterns. Indeed, sequential models, especially deep sequential models, showed superior performance over static models on predictions in several Human Computer Interaction (HCI) domains (Beutel et al., 2018; Donkers, Loepp, & Ziegler, 2017; Kim et al., 2019).

Based on the behaviour-based features and statistical models prevalent in the literature on early prediction of academic performance, I organized the related research studies into three main categories: *static models with static features, static models with sequential features,* and *sequential models with sequential features.* The category of *sequential models with static features* is not found in the literature, because sequential models

generally require specific data structures of sequence representation that only sequential features could achieve. Thus, no previous studies have been conducted on log analysis using *sequential models with static features.*

Static Models with Static Features. Some of the earliest studies on performance prediction through log analysis calculated the term-total click frequencies on each section of an LMS using static models with strong assumptions of independence of observations, such as multiple linear regression (Seber & Lee, 2012), logistic regression (Kleinbaum et al., 2002), support vector machines (Schlkopf, Smola, & Bach, 2018), decision trees (Kamiński, Jakubczyk, & Szufel, 2018), or random forests (Breiman, 2001). Researchers extracted data from log files in LMSs to develop a multiple-regression model for predicting 134 first-year university Computer Science and Computer Engineering students at risk of performing poorly or of failing in blended courses and to further identify the most significant explanatory variables related to online activities when predicting students' academic achievement in a blended course (Zacharis, 2015). Another study compared the online learning behaviours between peer-moderated and teacher-moderated groups for 98 undergraduate students using log data extracted from an LMS to investigate the indicators that predicted students' learning outcomes (Hung & Crooks, 2009). The authors used cluster analysis to observe the differences in learning patterns across levels of academic achievement. Association rule analysis was used to discover meaningful relationships among logged events, employing support and confidence measures (Hung & Crooks, 2009). Also, a decision-tree model was developed based on several manually extracted features (i.e., independent variables) to predict students' final grades for both groups (Hung & Crooks, 2009). More recently, Xu et al. (2019) revealed the relationship between Internet usage behaviours and undergraduates' academic achievement by predicting students' course final grades from their Internet usage features, including total time spent online, Internet traffic volume, and login frequency. They

used the non-parametric Mann-Whitney *U* test to ascertain the significance of the differences between usage features and academic performance levels among groups of students. Additionally, they calculated the Spearman's correlation coefficient to explore any associations between participants' academic performance and their time-usage behavioural features. Finally, they compared the performance of three popular machine-learning algorithms (decision trees, neural networks, and support vector machines) to validate the predictive power of the extracted features to performance. Findings showed that features related to Internet usage time were able to discriminate and predict students' academic performance. All three algorithms showed substantial accuracy on predicting performance using the time-usage features (Xu et al., 2019).

Static Models with Sequential Features. To address the limitation of simply using static models and features, previous studies harnessed the temporal nature of log data by transforming the static LMS features to sequential features to predict academic performance. However, most studies only used sequential features (i.e., extracting LMS features at different time steps), without adopting sequential models to describe the temporal dependency of the behaviour-based features. Instead, they employed static models, assuming that the temporal LMS features were not related chronologically but rather that they were independent from each other (Juhaňák, Zounek, & Rohlíková, 2019; Keogh & Kasetty, 2003). A study used hierarchical regression to predict 530 South Korean undergraduate students' self-regulated learning outcomes, represented by course grades in an online-learning course, from their weekly online activities, such as total viewing time, late submissions, proof of reading, message created, and total time spent on sessions (You, 2016). Another study collected 4,989 Dutch university students' log data from 17 blended courses on Moodle, extracting weekly click frequencies and time spent on different LMS sections to predict students' final scores (Conijn et al., 2016). In total, LMS features collected across 11

weeks, including total time spent online, number of course pages viewed, number of discussion posts, number of quizzes started, and number of assignments submitted, were fit into a multilevel regression model to observe the stepwise prediction accuracy for students from different grade levels. The findings showed a significant increase in the proportion of the final grade variance explained as more weekly variables were added into the model. However, both previous studies use a static model with sequential features, assuming that the weekly LMS features are independent of each other. Thus, they used multiple regression to make predictions, which neglects the temporal nature of online activities. Waheed et al. (2020) also implemented a three-layer perceptron and compared its classification performance with a support vector machine (SVM) and logistic regression on predicting at-risk students' dropout rate using quarterly aggregated hand-crafted features from the VLE click-stream data. They found that the artificial neural network (ANN) not only outperformed the SVM and logistic regression at the fourth quarter but also yielded early good prediction accuracy and better accuracy at every time step (quarter 1 to quarter 4).

Sequential Models with Sequential Features. The last category is sequential models with sequential features. Geigle and Zhai (2017) proposed a student behaviour-representation method that enables the automatic discovery of behaviour patterns based on students' click log data collected from a MOOC. They used a two-layer Markov model (2L-HMM) to extract interpretable and meaningful behaviour representation of students' interactions with the MOOC platform. However, their study only modeled students' online learning at the behavioural level, without linking learning behaviours to academic performance. One study employed a univariate sequential classification model to predict the dropout rates of students from the Open University (OU; Liu, Wang, Benachour, & Tubman, 2018). More specifically, they counted 170,000 learners' daily click frequencies on all sections of OU and fit the temporal sequence feature to a time-series forest (TSF) model to predict the dropout status of

students registered in OU. The univariate prediction model yields an accuracy of 84% with only 5% of the dataset. Results show a great potential of applying sequential models to process temporal features. Nonetheless, compared with multivariate sequential models, univariate sequential prediction models bring in limited information on students' online learning behaviours. Liu et al. (2018) expanded the univariate sequential model by proposing a prediction model based on Recurrent Neural Network (RNN), namely Long Short-Term Memory (LSTM), to predict learners' early dropout status. The model that they implemented and trained on a dataset extracted from Chinese University MOOCs yielded an accuracy of 90%. The authors transformed the dropout-prediction problem into a sequential prediction problem and utilized LSTMs to make predictions. However, their work has several limitations. First, they apply their approach in the context of the informal MOOCs online learning platform, so it is not clear whether the results would generalize to other online learning environments, such as the LMSs formal learning platform. Second, the scope of the features they employed is limited, including only learners' interactions with the forums (e.g., post count in general discussion, post count in professor answer area, and post count in class exchange area) and ignoring other important aspects of online learning. Third, the study focuses on dropout rate, but it does not explore the relationship between log events and academic performance.

Recent studies implemented sequential deep sequential learning algorithms and compared the prediction performance of students' dropout rate in virtual learning environments using large-scale, publicly available MOOC datasets. Hassan et al. (2019) implemented a deep-learning model with LSTM and compared its performance with a two-layer perceptron and logistic regression on predicting students' dropout rate on the Open University Learning Analytics (OULA) dataset in different weeks (e.g., weeks 5, 10, 15, 20, and 25) across a term. They found that the deep-learning model performed best among the

three models at every time step and achieved high accuracy as early as Week 10, which was comparable to the prediction accuracy of a two-layer perceptron and logistic regression in Week 25. Their study demonstrates the potential of deep-sequential models in predicting students' attainment in online learning platforms.

Learner Modeling in Digital Learning Systems

While EDM is developing at a rapid pace in the last decades, one branch of EDM, namely adaptive tutoring and e-learning, draws special attention from the field for two reasons. First, the log-event data collected from the open online tutoring systems is at a much larger scale compared with data collected from local digital learning systems. Second, the huge datasets are publicly available so that different newly devised analytic approaches and algorithms can be tested on them and compared with benchmarks.

Emerging research on learner models is being conducted based on large volumes of publicly available process data, which facilitates researchers and teachers to better understand learners' behaviours in the ITSs and to implement efficient tools to scaffold learners on adaptive learning paths that meet their individualized needs (Abyaa et al., 2019). Learner models are crucial for individualized learning and tutoring, as they model learners' behaviours and characteristics, including prior knowledge, emotions, and demographic information to predict their future skill performance and to adapt within the learning platforms. Some of the commonly used learner models include Item Response Theory (IRT; Embretson & Reise, 2013), Performance Factor Analysis (PFA; Pavlik et al., 2009), Bayesian Knowledge Tracing (BKT; Corbett & Anderson, 1994), and Deep Knowledge Tracing (DKT; Piech et al., 2015).

Item Response Theory (IRT; Embretson & Reise, 2013) is the methodological framework that applies generalized linear models to predict the probability that the student will answer the next item correctly based on previous item responses. This technique is

widely used in computer-adaptive testing to measure student mastery of latent skills (van der Linden & Glas, 2010). Several variants of IRT have been developed to incorporate more parameters for better calibration among various contexts, including 1-parameter logistic item response theory (1PL IRT; Rasch, 1993), 2-parameter logistic item response theory (2PL IRT; Lord, 1980), 3-parameter logistic item response theory (3PL IRT; Birnbaum, 1968), 4-parameter logistic item response theory (4PL IRT; Barton & Lord, 1981) model, and partial credit item response theory model (PCM; Masters, 1982) for polytomous response data. The 1PL IRT (i.e., Rasch model) is widely used in computer-based educational and psychometric measurement for its simplicity and precision. The Rasch model can be computed as follows:

$$ln\left(\frac{P(X_{ni}=1)}{P(X_{ni}=0)}\right) = \theta_n - \beta_i,$$

where X_{ni} is the actual response to a dichotomously scored item, X_{ni} = 1 refers to the student correctly answering an item, and X_{ni} = 0 refers to the student incorrectly answering an item; θ_n is the ability parameter of student n (a higher θ_n means that the student has a higher ability level); and β_i is the difficulty parameter of item *i*. Thus, the log odds of the probability of success on an item $P(X_{ni}=1)$ over the failure trial $P(X_{ni}=0)$ can be calculated as the $(\theta_n - \beta_i)$.

Bayesian Knowledge Tracing (BKT; Corbett & Anderson, 1994) is a popular learner modeling approach in the EDM community, designed to make inferences about learners' knowledge learning and transfer. A standard BKT is a two-state Hidden Markov Model (HMM; Eddy, 1996) that estimates learners' skill mastery, where learners' response is a binary (correct or incorrect) observable node, and a knowledge state is a hidden binary node with mastery and non-mastery states. Although BKT is one of the most popular knowledge tracing models, it presents some limitations. First, BKT dichotomizes student knowledge mastery states into "learned" and "unlearned", which is unrealistic. Second, BKT maps items to a single concept, and cannot be adapted for questions that cover multiple knowledge

concepts. Some variants of BKT have been developed to improve predictive accuracy such as incorporating time-based contextualized parameters (Baker et al., 2008) or past item responses and correctness (Beck et al., 2008) into BKT estimation. Some variants are also developed based on BKT such as individualized BKT (Yudelson et al., 2013) and BKT plus (Khajah et al., 2016).

BKT can be characterized by five basic elements as plotted in Figure 1:

(1) $P(Initial)$ or $P(L_0)$: the probability that the skill was known *a priori* (i.e., before learning began); (2) $P(Learn)$ or $P(T)$: the probability that the skill will transition into mastered state after a practice attempt; (3) $P(Forget)$ or $P(F)$: the probability that the skill will transition into a non-mastery state after a practice attempt; traditionally, *p-forget* is set to zero and is not counted towards the total number of parameters; (4) $P(Slip)$ or $P(S)$: the probability that a mastered skill is applied incorrectly; and (5) $P(Guess)$ or $P(G)$: the probability that an unmastered skill will be applied correctly.

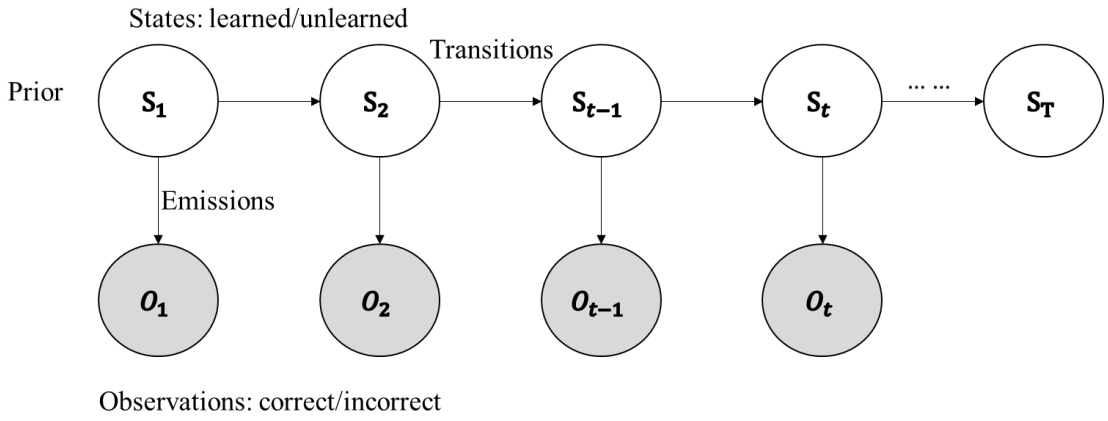

Figure 1. *Graph Representation of the Standard Bayesian Knowledge Tracing (BKT) Model*

In standard BKT, the binary state nodes, 0 and 1, represent the state of student knowledge mastery: 0 refers to the student not mastering the skill or knowledge component, whereas 1 assumes that the student mastered the skill. The conditional probability of a student mastering a skill at time step *t* can be calculated as follows:

$$P(Correct_t) = \frac{P(L_{t-1}) * (1 - P(S))}{P(L_{t-1}) * \big(1 - P(S)\big) + \big(1 - P(L_{t-1})\big) * P(G)}$$

$$P(Incorrect_t) = \frac{P(L_{t-1}) * (1 - P(S))}{P(L_{t-1}) * P(S) + (1 - P(L_{t-1}) * (1 - P(G)))}$$

Thus, the student's actual response $Response_t \in \{0, 1\}$ to an exercise can be used to compute the probability of mastering a skill at time t:

$$P(L_t) = P(L_t|Response_t) + [1 - P(L_t|Response_t)] * P(T)$$

Deep Knowledge Tracing (DKT; Piech et al., 2015) is another widely discussed approach in the EDM community. DKT was first proposed by Piech et al. (2015). It utilizes recurrent neural networks to model learners' sequential product data. Similar to BKT, the DKT approach models students' product data and observes knowledge transfer at both latent skill level and item level. However, DKT does not treat skills independently. Instead, it updates the knowledge states on all skills concurrently as a learner practices a single skill. Moreover, DKT considers the temporal connections between actions within the systems. The RNN is illustrated in Figure 2. At different time steps $\{t = 1, 2, \ldots, T\}$, the input layer x_1, $x_2,\ldots, x_T$ is described as a one-hot encoding of a student trial-performance, the hidden layer $h_1, h_2,\ldots, h_T$ is an RNN layer for predictions, and the output layer $y_1, y_2,\ldots, y_T$ is the correctness prediction of every knowledge component. The output vector can be computed as follows:

$$h_t = tanh(W_{hx}x_t + W_{hh}h_{t-1} + b_h),$$

$$y_t = \sigma(W_{yh}h_t + b_y),$$

where W_{hx} is the input weight matrix, W_{hh} is the recurrent weight matrix, and W_{yh} is the output weight matrix; b_y and b_h are the latent bias vector and the output bias vector, and σ is the sigmoid function.

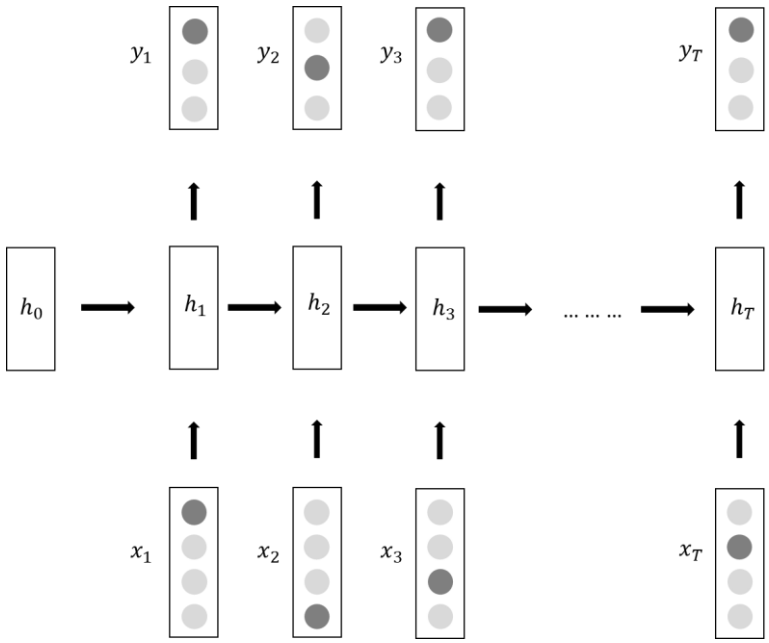

Figure 2. *Recurrent Neural Network (RNN) Representation of Deep Knowledge Tracing (DKT)*

The deep learning approach (Goodfellow et al., 2016; LeCun, Bengio, & Hinton, 2015) captures the temporal information of student actions to make more accurate performance predictions. DKT also has some variants including Long Short-Term Memory DKT (LSTM DKT; Piech et al., 2015) and DKT with Gated Recurrent Units (GRU; Chung et al., 2014) to improve performance predictions. Compared with the vanilla RNN, the RNN-variant LSTM introduces a forget gate to retain information from previous steps. Figure 3 illustrates an LSTM cell in which the output at time *t* depends on the input *x* at both time (*t* – *1*) and *t*:

$$y_t = W_{yh}h_t + b_y,$$

$$h_t = H(W_{hx}x_t + W_{hh}h_{t-1} + b_h),$$

where y_t is the output unit, x_t is the input at time *t*, W_{hx} is the input weight matrix, W_{hh} is the recurrent weight matrix, and W_{yh} is the output weight matrix, b_y and b_h are bias vectors, and H is usually a series of element-wise operation functions:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i),$$

$$f_t = \sigma\big(W_f x_t + U_f h_{t-1} + b_f\big),$$

$$g_t = tanh\big(W_g x_t + U_g h_{t-1} + b_g\big),$$

$$s_t = i_t \odot g_t + f_t s_{t-1},$$

$$O_t = \sigma(W_o x_t + U_o h_{t-1} + b_o),$$

$$h_t = O_t \, tanh(s_t)\,,$$

in which x_t is the input vector at time step t, i_t is the input gate's activation vector, f_t is the forget gate's activation vector, g_t is the cell input activation vector, s_t is cell state vector, O_t is the output gate's activation vector, and, lastly, h_t is the output vector of this LSTM unit. W_i, W_f, W_g, W_o, U_i, U_f, U_g, U_o are the estimated weight matrices, b_i, b_f, b_g, b_o are the bias vectors, σ is the sigmoid function, $tanh$ is the tangent function, and $\odot$ is the element-wise multiplication operator.

DKT also has many extensions including DKT plus (Yeung & Yeung, 2018), DKT with Convolutions (Yang et al., 2020), DKT with Rich Features (Zhang et al., 2017), and Graph-based Knowledge Tracing (GKT; Nakagawa et al., 2019). DKT and its variants gained its popularity in the field of EDM in the last decade, as it displayed higher predictive accuracy and abilities of processing sequential data and exploiting the temporal dynamics compared with probabilistic approaches.

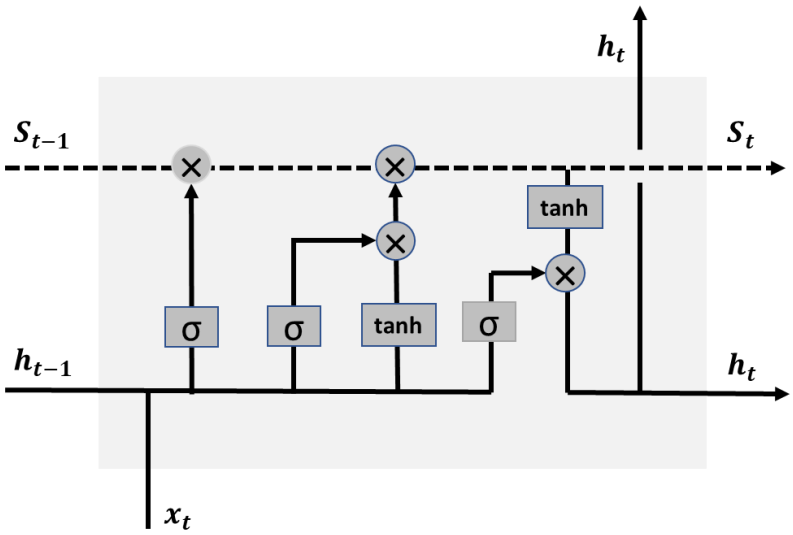

Figure 3. *Conceptual Representation of the LSTM Cell*

Recent studies also employed the Convolutional Neural Network (CNN) instead of RNN to process student interactions with ITSs. CNN is a commonly used model for image recognition, and currently has been widely used in text analysis. The convolutional layer is seen as a function which could learn features from n-grams, and can be represented as:

$$Z_i = \mathrm{f}\left(W_z\left[x_i^j : x_i^{j+h_w-1}\right] + b_z\right)$$

where x_i is the *i*th embedded word, W_z is the weight matrix, b_z is the bias vector, h_w is the window size of the convolutional layer, ƒ is a non-linear activation function (i.e., *sigmoid* or *tanh*), and Z_i is the output of feature representation. A conceptual representation of a 5-dimension CNN for dichotomous text classification is shown in Figure 4.

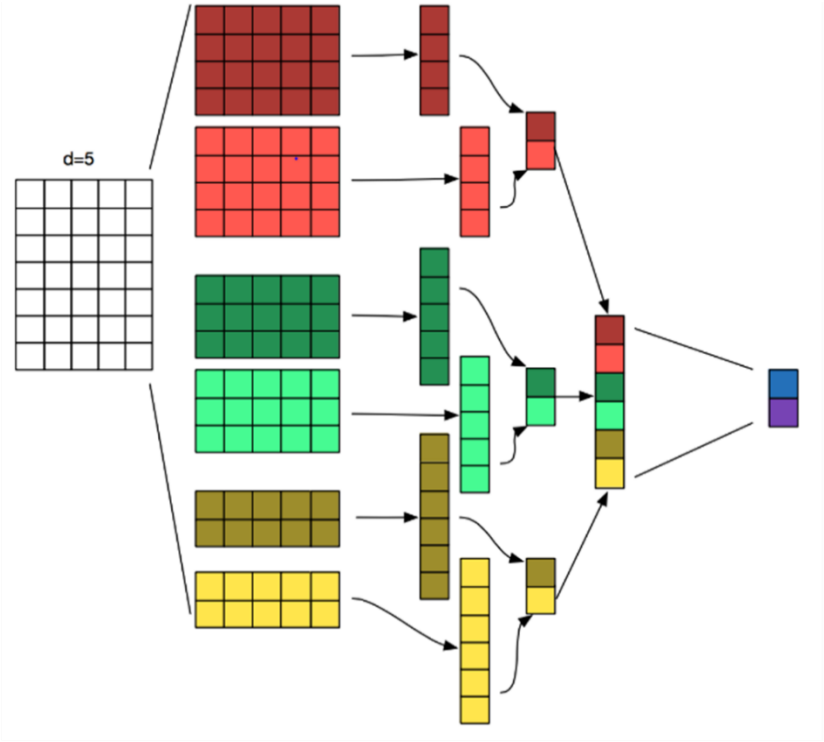

Figure 4. *Conceptual Representation of CNN*

Automated Feedback Generation in Digital Learning Systems

Precise learner modeling is important, but it is not enough for effective delivery of online tutoring and e-learning. Providing feedback is a key factor for improving knowledge. In education, feedback is defined as the information provided by an agent regarding aspects of one's performance or understanding (Hattie & Timperley, 2007; Shute, 2008). High-quality personalized and timely feedback can improve learners' performance (Hattie & Timperley, 2007), but feedback provision is often reported as the long-standing weakness of ITSs and e-learning systems (Maniktala et al., 2020; McFarland & Hamilton, 2005). On the one hand, students complain that they receive too little quality feedback in the process of learning (Boud & Molloy, 2013; Ferguson, 2011). On the other hand, students are reported to misuse and abuse the feedback or hints provided by the ITSs (Price et al., 2017). Thus, knowing how and when to provide real-time personalized feedback that guides and motivates students' learning remains a challenge. Some of the earlier attempts of implementing feedback within ITSs focused on providing real-time online tutoring by humans in those

environments (Merrill et al., 1992; Heffernan & Koedinger, 2002). Findings show that human tutoring is effective in improving students' performance, but it is time and labor consuming, and not applicable to large-scale practice and open-ended platforms. Although providing feedback is crucial for scaffolding students to improve performance (Hattie & Timperley, 2007), research on how learning technologies effectively estimate learners' skills and automatically generate individualized feedback is lagging (Deeva et al., 2021; Maniktala et al., 2020). Specifically, discussions of automated feedback systems are still ongoing regarding when to provide feedback (immediate vs. delayed), how to provide feedback (summative vs. formative, adaptive vs. non-adaptive, expert-derived vs. data-driven), and how to assemble the feedback content (generic vs. specific, KC based vs. item based).

Based on a review of automated feedback systems for learners in the last decade, Deeva et al. (2021) proposed a framework for TAF-ClaF (Technologies for Automated Feedback – Classification). They identified the main elements for automated feedback systems as outlined in Figure 5. The elements can be categorized into three dimensions: *where* feedback is deduced, *how* it should be delivered, and *when* to provide it. The first question of 'where' is addressed by the *domain model*, which is a structured domain knowledge representation based on Bloom's taxonomy (Bloom et al., 1984). It defines the subjects (e.g., math, literature, etc.) and the corresponding sequential order of teaching of the knowledge components. The second question of 'how' is addressed by the feedback generation model, which is determined by two factors: (1) whether the rules for feedback generation is derived from *expert knowledge* or *student data*; and (2) whether the approach of feedback delivery is *expert-driven*, *data-driven*, or a mixture of both. More specifically, *expert knowledge* refers to the learning and teaching theories dictated by experts for predefining feedback for students, whereas *student data* represents the learning traces from students that can be analyzed through educational data mining or learning analytical

techniques to empirically induce knowledge representation architecture and to trigger feedback delivery. Regarding the feedback generation model, the *expert-driven* model uses expert knowledge as the set of rules for both feedback generation and feedback provision, the *data-driven* model is solely derived from student data and it delivers feedback purely based on empirical evidence, whereas the mixed model uses both sources of expert knowledge and student data to derive feedback. The last question of 'when' is defined by the timing of feedback provision. Recent technologies have advanced the implementation of hint generation systems and feedback generation for unstructured domains such as programming tasks and essay writing.

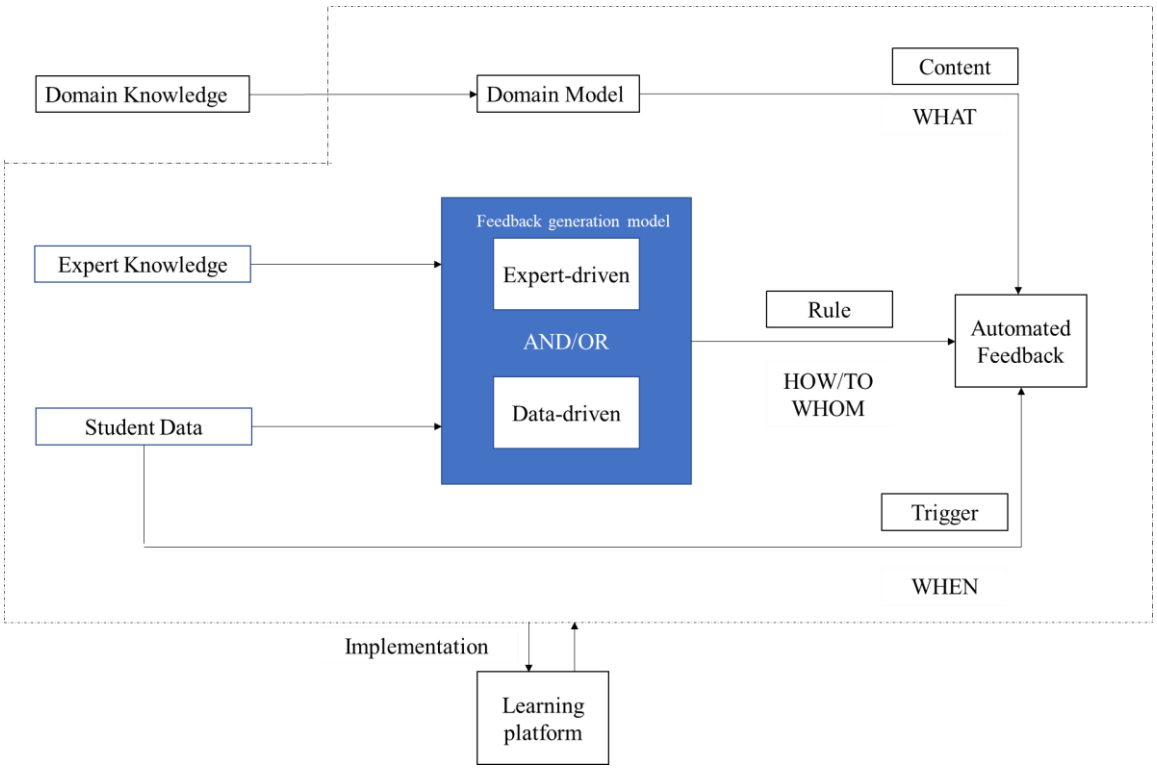

Figure 5. *A Representation of a Typical Automated Feedback Technology by Deeva et al. (2021)*

Hint Generation. Most empirical studies focus on implementing automated hint generation methods for intelligent tutoring systems to scaffold students and recommend next-step learning. For example, Price et al. (2017) sampled 171 students to compare the quality of

data-driven hint-generation algorithms. The hints produced by the algorithms were evaluated using a "QualityScore" procedure, which sampled a set of "gold standard" hints produced by a group of three expert tutors familiar with the problems to set a benchmark for hint comparison. The data-driven hints were distinguished into three categories: full match, partial match, and no match in comparison to the quality of a "gold standard" hint. This study included two experiments. The first experiment aimed to validate the "QualityScore" procedure as a benchmark for hint evaluation. QualityScore matched 82.9% of manual hint ratings, which is why it can be deemed as a reasonable procedure. The second experiment compared 6 hint generation algorithms and found that the Intelligent Teaching Assistant for Programming (ITAP) and SourceCheck outperformed every other algorithm in terms of producing reasonable hints. ITAP was designed specifically for Python, which suggests that it holds an advantage over the other algorithms for the Python dataset, which could be a possible reason for its success.

Marwan, Williams, and Price (2019) also conducted two studies to investigate next-step programming hints' effect on learners' performance, overall understanding, and perspectives. The first study sampled 10 students from an introductory engineering course who had no prior programming experience and the second study sampled 201 paid workers from Amazon's Mechanical Turk platform. Both studies used a block-based programming platform called iSnap. The first study's goal was to understand student perceptions of four types of hints: code hint only, code hint with textual explanation, code hint with self explanation prompts, and code hint with both textual and self-explanation prompts. A variety of these hints would be offered every 2 minutes over the course of a 15-minute programming task that would update with each code change. Students could use these hints immediately or choose to wait as the hints would accumulate over time, hence they were given the option to request multiple hints in a row. This feature was specifically put in place to encourage hint

usage while preventing overreliance on them. The study spanned two programming tasks in which the first one was followed up with questions about the timing, helpfulness, trustworthiness of each hint and what motivated them to use it. Task 2 was similar to the first task, but it was noticeably more difficult and, thus, had an easier and harder version. Students who finished Task 1 within ten minutes were given the harder version while the rest were given Task 2 to ensure a fair level of challenge while attempting the second task. A similar interview to Task 1 was conducted after Task 2 as well. The interview answers showed a positive sentiment towards textual explanations (7 out of 10 students), as it helped them understand the "how" and "why" of a code hint. Self-explanation prompts were appreciated (4 out of 10) as they prompted students to think more deeply about the given hint. However, 3 out of 10 students criticized self-explanation prompts for being frustrating and confusing. The interview additionally revealed that there was no specific time when the students agreed to get hints. The second study's goal was to investigate performance and learning transfer on a larger sample size. The procedure of the study was similar to the first study, except for a few differences. There were three types of hints: No hint (control), Code hints with Textual Explanations (CT) and Code hints with Textual and Self-explanation prompts (CTE). Each learner was assigned one of the three hint types provided in Task 1. However, no hints were given in Task 2, to measure one's ability to perform a similar task without help. The results of immediate performance were compared based on the completion of 4 objectives during both tasks. More learners completed the task in CT condition (27.8%) and CTE condition (45.8%) than control condition (22.2%). The difference between the three groups became more pronounced over time. In Task 2, the measure of learning was compared through a post-hoc Dunn's test with Benjamini-Hochberg correction, which showed a significant difference between CTE learners and both the control group ($z = 2.73$; $p = .01$) and CT learners ($z = 2.35$; $p = .028$), respectively. However, there was no significant difference between CT

learners and the control group. This suggests that only code hints with self-explanation prompts improve learners' performance. There was a weak but significant negative Spearman correlation between the number of hints requested on Task 1 hints and the performance on Task 2 in the CT group (r = -.243, p = .03) but not in the CTE group (r = -.102, p = 0.40). This suggested that the number of hints requested does not strongly predict the performance of current or future tasks. Ratings of learners on a scale of 1-10 were much less in the control group (*Median* = 5, IQR = 4) compared with CT learners (*Median* = 7, IQR = 3.4) and CTE learners (*Median* = 8, IQR = 2). This suggests that hints were perceived as much more helpful when providing textual and self-explanation hints. Overall, these results provided an important step in understanding the potential benefits and limitations of coding hints and they also suggest significant performance improvement with textual explanations.

Another study (Marwan, Lytle, Williams, & Price, 2019) introduces a straightforward method for generating textual explanations to accompany automated, next-step programming hints. Next-step hints can support students during program construction and automatically adapt to the student's current code to support different student solutions. The authors evaluated the impact of adding these textual explanations to code hints in iSnap through two controlled experiments with different populations. In Experiment 1, they conducted a controlled study on novices in an introductory programming course for non-CS majors and found that explanations may increase students' willingness to use and follow the hints. However, they also claimed that the majority of students did not use hints, so the sample size was small, and the results were inconclusive. The further experiment was conducted with crowd workers recruited on Amazon's Mechanical Turk platform. They found that learners who received code hints with textual explanations rated hints as significantly more useful and were more likely to follow hints which showed a similar trend to Experiment 1. In addition, learners who received textual explanations were also significantly more likely to explain the

relationship between the received hints, their code, and the assignment objectives. The authors argued that they isolated the impact of one specific element of support – textual explanations – to evaluate it directly. In addition, they evaluate the systems by ascertaining student performance on future tasks as a measure of learning and by using self-explanations as an alternative technique to measure the impact of hints on students' knowledge. The platform generates textual explanations for a given problem in iSnap by identifying all common abstract syntax tree (AST) nodes in the database of solutions, and then manually annotating each of these solution AST nodes with a textual explanation for the corresponding hint. Therefore, the study designed complementary textual explanations for existing code hints in iSnap, with the goal of overcoming the limitations of code hints that only tell a student what to do.

A study (Mao et al., 2019) sampled 171 undergraduate University students (non-CS majors) over the course of four months in an introductory programming course on iSnap, a block-based coding platform. The goal of the study was to effectively predict whether the student would succeed eventually and if the student would need intervention at any given time. The binary measure of success was classified as Trajectory Level prediction and the need for intervention was classified as Event Level prediction. The study uses the Recent Temporal Pattern (RTP) versions of classic machine learning models such as K-nearest neighbors (KNN; Altman, 1992), Support Vector Machine (SVM; Cortes & Vapnik, 1995), and Logistic Regression (LR; Wright, 1995) to build interpretable models and compare them to the performance of their standalone versions as well as a deep learning LSTM model. A student's state was determined by the identification of data-driven features (DDF) and expert features (EF). The results reflected that the RTP-based models were able to predict student success within a minute of an otherwise 20-minute programming task. The RTP-based models also successfully predicted the need for intervention 85% of the time during repeated

application every 5 minutes. However, the study was limited by a lack of progressive features with multiple values and the measure of success was restricted to a binary metric. A larger sample size and longer course duration could also result in the incorporation of recency measures for the temporal patterns. The next steps for the study include incorporating recency measures into the temporal patterns and expanding the binary measure of success and need for intervention.

To summarize, results from the above literature review show that most studies: (1) were based on unstructured domains; (2) used supervised-learning techniques to track students' learning status; (3) and generated hints within the systems for improving students' performance and adapting future learning. However, few studies addressed skill-level automated feedback generation for intelligent tutoring systems in other domains.

Feedback Generation for Unstructured Domains. Among a handful of studies on automated feedback generation systems, most were developed for programming tasks, constructed response questions, or essays. Silva et al. (2019) sampled 34 students in an introductory programming class to present an approach to provide adaptive feedback while the programmer solves a problem in the form of text, video, and flowchart feedback. The study explored various previous attempts at providing adaptive feedback but found that most systems provided low-quality feedback out of which only 32.7% of the systems generated feedback for error correction and only 18.8% generated messages that help the student proceed with the next step of the problem. Thus, the researchers developed an intelligent tutoring system (ITS) aiming to provide high-quality adaptive feedback that the students can use to program in C/C++ language. When students ask for feedback, the system checks if the student has indicated which part of the solution they need help with by selecting that portion. Then, the system looks for specific feedback whose content has been created and associated with code parts of the model solution. If there is no selection, the system uses the current

state of the student solution to search for general feedback. The process of generating feedback involves (1) extracting information from each question posed to the learner; (2) extracting the part of the code provided by the student; (3) extracting the gold-standard solution given by the ITS; and (4) performing a similarity calculation between the student's code and the solution, using the Levenshtein's distance algorithm, which returns a score ranging from zero to one. The study involved a quasi-experiment to determine the pre-test and post-test performance of each student giving them one basic programming problem to solve in each test. The results of the test were evaluated using test cases and scores between zero to ten. They found that the students' performance on the post-test was significantly better than their pre-test performance, as evidenced by a statistically significant t-test (p-value $< .001$). In the second session of the experiment, the students' interaction log with the system was analyzed to check their response and behaviour during feedback requests. It was found that some students showed atypical behaviour by making several feedback requests in sequence but did not use the feedback to make changes to the solution. By removing these students from the analysis, it was found that 85% of the feedback messages received by the group were useful for the student to make progress on their solution.

An ongoing study (Katan & Anstead, 2020) explores student behaviour on a gamified platform, Sleuth, for teaching introductory programming to large student cohorts. The study sampled 1,500 students over an assessed coursework assignment to empirically test automated feedback generation and gamification of educational material. The platform provides a set of code puzzles based on a film-noir detective story where a 'Chief' provides immediate feedback (i.e., for runtime or compile-time errors) on every attempt, which is then graded automatically. The coding exercises are generated with variations between each attempt to create an inexhaustible supply of puzzles and reduce the scope of plagiarism. The preliminary results show that the student cohorts performed significantly better in Sleuth with

an impressive mean grade of 90.67%, whereas module tests yielded a much poorer mean grade of 66.94%. A positive correlation was found between the two Sleuth assignments through a Spearman's rank correlation ($r = .63$, $p < .01$). A significant improvement in student motivation and activity was observed, as there was an average of 158 puzzle attempts per student. Students also seemed to enjoy the presence of Easter Egg levels that had no graded reward associated with them. The perceived task difficulty of 2.7 on a 5-point Likert scale and the high levels of achievement indicate some degree of intrinsic motivation. Some of the limitations of this study include undesirable and obsessive behaviours to get a 100% on each puzzle due to the lack of an upper limit on time or attempts for them. Also, rather than adopting a strategic route towards solving the problems, students approached them in a much more linear way and compensated with multiple failed attempts instead. The next steps for the study are to explore the role and method of feedback delivery in student performance and mitigate the sequential approach when students solve the puzzles.

Keuning et al. (2014) examined the rising number of individuals who wish to learn how to program and started to introduce a programming prototype that helps students with feedback and hints to progress towards a solution for an introductory imperative programming problem. The study's main goal is to provide an intelligent strategy-based feedback tutor, trying to generate relevant hints and tips to the programmer to solve a somewhat simple problem. The study implies that many of the programming tutors' comments on a program are based on a complete program, but the authors' programming tutor is a step-by-step tutor who comments on each of the statements of the program. There is a web interface that enables the student to select the exercise from a list of available exercises. The editor, in which the code can be typed, provides syntax highlighting. When the student first asks for a hint, the first option (branch) of the hint tree is shown. The student has the opportunity to 'expand' (denoted by the -+ symbol) a specific path and view a hint that provides more details. The tutor is built on top of the IDEAS

framework (Interactive Domain-specific Exercise Assistants), which provides services to build exercise assistants to help students solve problems incrementally. The tutor has three components: (1) Domain specification: a domain is described, among other things, by a grammar for its abstract syntax and an accompanying parser to process submitted work. (2) Steps: a step is a transformation on values of the domain, such as refining or rewriting a student submission. (3) Strategies: a strategy combines basic steps and specifies which steps can be taken, in which order. A strategy to solve an exercise is composed of several steps. Two types of steps are used in the tutor for imperative programming: append steps and refinement steps. A programming exercise can be specified by providing a set of model solutions and an exercise description in a text file. Students can do the exercises by creating a solution and asking for feedback. The tutor can understand different algorithms, different statements, and different statement orders which helps to differentiate between the different solutions. The study evaluated its tutor by collecting data from first-year IT-students during their Web programming course and their Java programming course. They were given three exercises. Exercises 1 and 2 are relatively simple PHP exercises from the Web programming course. The tutor was capable of recognizing 75% (24 out of 32, for the first exercise) and 33% (for the second exercise) of the solutions that they considered similar to a model if they would be manually assessed.

Research Gaps

Several research gaps were identified by Deeva et al. (2021) in the existing literature based on a review of automated feedback generation technologies from 2008 to 2020. First, the transparency and implementation accessibility of feedback generation systems are deficient. Most of the related studies did not share the technical details on how their system was created and what tools or languages were used. Second, most studies did not report the educational frameworks or learning theories underlying their automated feedback generation

systems, and they did not report the context in which the system was built. Third, only half of the systems reviewed reported that they adopted a data-driven approach for feedback generation, whereas most of the previous studies still heavily relied on expert knowledge. Although an expert-driven approach is not a drawback by itself, it might slow down the full automation of tutoring and assessment. Fourth, while most research emphasized adaptiveness as an important factor for feedback, only one third of the reviewed studies adapted the feedback to student characteristics and personality. Fifth, the selected studies still show deficiencies in providing personalized feedback, especially on determining when the feedback should be delivered. In addition, the review of empirical studies on automated feedback generation revealed that most automated feedback systems aimed to generate item-level comments or feedback (i.e., created based on the learners' performance on items), whereas only few incorporated measurement models to detect students' latent skill mastery states and provide higher-level feedback on their performance (Lu et al., 2021).

Based on the comprehensive review of the previous studies, we identified the following additional research gaps. First, most automated feedback systems only evaluated student performance and mastery of skills based on product data, whereas the process data collected from the ITSs were unused. Second, most automated feedback generation models aimed to provide item-related hint to guide students to complete the items for structured domains or corrective feedback for unstructured domains such as essay writing and programming using natural language processing techniques. Research on providing skill-level feedback for structured domains is lacking.

To address the gaps identified from previous studies, we propose a data-driven based, personalized feedback generation system informed by cognitivist learning theory that suits different learner environments and learner models for structured domain knowledge (i.e., math, algebra, and statistics). Details on the implementations and evaluations of different

components of the automated feedback generation framework were addressed and elaborated to ensure transparency, accessibility, and reproducibility. Specifically, we implemented an evaluation model that incorporates both student product data and process data and used natural language processing (NLP) techniques to augment our corpus of feedback templates for both positive and negative feedback to avoid word repetitions. We also detailed the feedback generation process for different learner models.

Chapter Summary

Chapter 3 first reviewed the current trends of studies on technology-enhanced learning and digital tutoring and assessment. Next, it summarized the features and models that were commonly used to model students' log-event data and introduced several mainstream learner models that apply to large-scale log-event datasets. Third, it examined the classifications and empirical studies of automated feedback technologies including hint generation and feedback generation systems for unstructured and structured domains within digital learning environments. It further highlighted the advantages and potentials of a data-driven approach to automated feedback generation. Finally, it identified the existing research gaps. The next chapter describes the proposed novel data-driven framework of automated feedback generation for different learner models.

Chapter 4 Methods

Overview

This chapter introduces the proposed framework of *Automated Feedback Generation for Learner Models*. The organization of this chapter is guided by the three proposed research questions:

1. To what extent do the learner models perform accurate and interpretable estimations of students' performance? *What are the predictive accuracies, input representations, output representations, and characteristics of the different learner models? Does the proposed learner model that incorporates both product data and process data yield better prediction performance compared with previous learner models?*
2. To what extent does the feedback generation method produce fluent and related feedback? *Can the feedback generation method create a variety of feedback templates that are grammatically correct and semantically related?*
3. To what extent is the proposed framework of Automated Feedback Generation for Learner Models feasible for structured knowledge domains? *How does the proposed feedback generation method fit into different learner models? What format and information does the generated feedback provide for learners?*

Drawing upon Deeva et al.'s (2020) framework for TAF-ClaF (Technologies for Automated Feedback – Classification), we adapted the structure of the automated feedback technology and specified each component within the proposed framework of feedback generation for learner model as shown in Figure 6. Compared with Deva's framework, the proposed system makes the following contributions: (1) it replaces *Student Data* with *Student Product Data and Process Data* and (2) it defines the *Automated Feedback* model as being an *Augmented Templated-based Automated Feedback* model. The main components (i.e., the KC/learner model, student data, data-driven feedback generation model, the feedback trigger,

and feedback content), their implementations (i.e., the implementations of the learner model and the feedback augmentation model), and the evaluation methods are detailed later in this section. The implementations and evaluation metrics of different learner models are first described. Then, the implementations and evaluation metrics of the unsupervised feedback generation algorithm are described. This chapter ends with a summary of the synthesized framework.

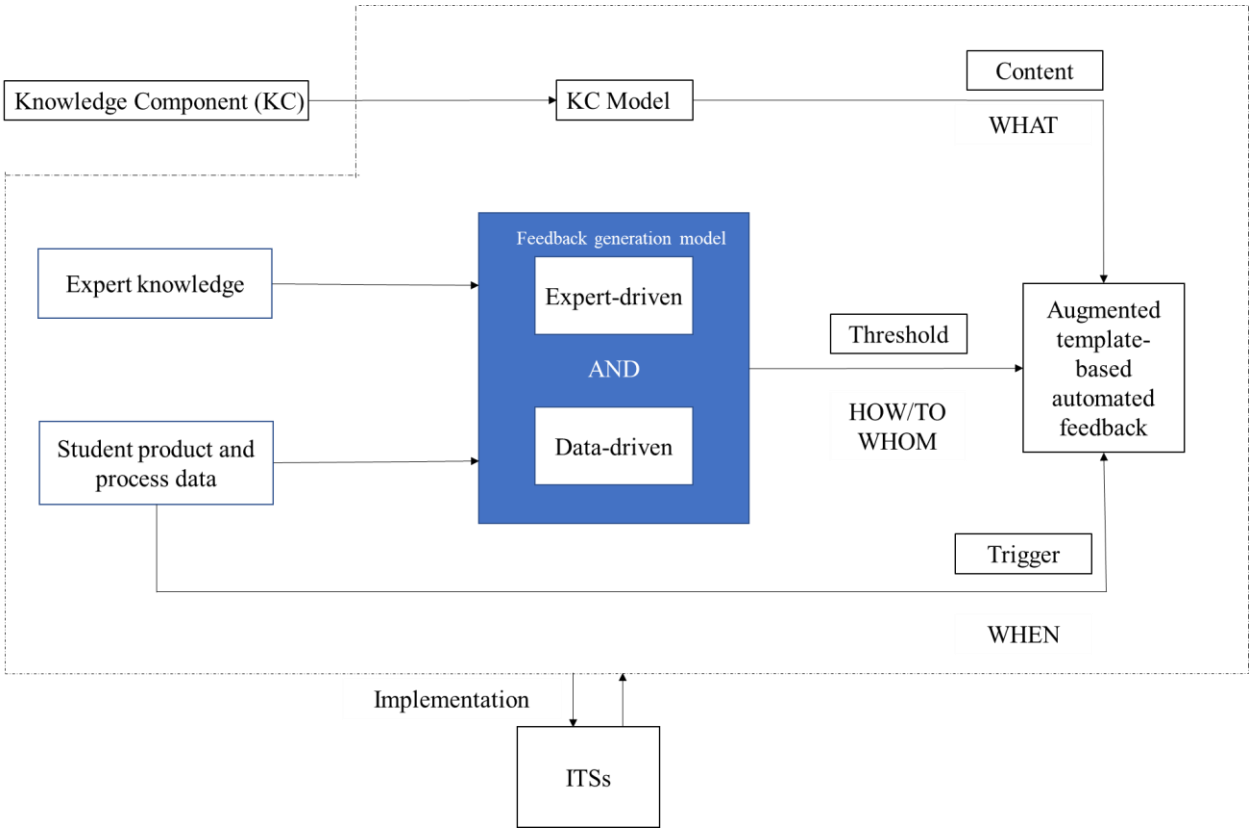

Figure 6. *Automated Feedback Generation for Learner Models*

Datasets for the Learner Models

Three datasets of various sizes, commonly used as benchmarks in previous studies, were selected to compare the characteristics and performance of the learner models and to demonstrate that the proposed framework of feedback generation works well with different learner models: the ASSISTment 2009-2010 dataset (Heffernan & Heffernan, 2014), the KDD Cup 2010 EDM Challenge-Algebra I 2005-2006 dataset (Koedinger et al., 2010), and the OLI Engineering Statics - Fall 2011 (Koedinger et al., 2010). Table 2 summarizes the

numbers of students, Knowledge Components (KCs), items, and total steps (interactions with the system) for these datasets.

Table 2

Dataset Information

	ASSISTment 2009-2010	**Algebra I 2005-2006**	**OLI Fall 2010**
Student	4,163	574	333
KCs	110	178	80
Items	17,709	173,113	300
Interaction	459,209	809,695	261,948

ASSISTment 2009-2010 is one of the largest datasets collected from intelligent tutoring systems and the benchmark for learner modeling studies. ASSISTment is a computer-based learning and assessment that is often used to teach math after school. This dataset was collected from the ASSISTment skill builder problem, which assigns a student to work on similar consecutive questions (normally set to answer 3 questions correctly in a row) until the student can answer problems on the KCs correctly. After completion, students commonly do not practice at the KC again. This dataset is the largest of all three datasets employed in this work. This dataset contains the main following columns as shown in Table 3.

Table 3

Summary of the ASSISTment 2009-2010 Dataset

Column	**Descriptions**
order_id	The chronological identifier (ID) of the original problem log.
assignment_id	Two different assignments can have the same sequence ID. Each assignment is specific to a single teacher/class.
user_id	The identifier of the student who solves the problem.

problem_id	The identifier of the problem.
orignial	1 = Main problem. 0 = Scaffolding problem.
correct	1 = Correct on the first attempt. 0 = Incorrect on the first attempt or asked for help. This column is often the target for prediction.
attempt_count	The number of attempts on this problem.
ms_first_response	The time in milliseconds for the first response.
skill_id	The identifier of the skill associated with the problem. For the skill builder dataset, different skills for the same data record are represented in different rows. Thus, if a student answers a multi-skill question, this record is duplicated several times and each duplication is tagged with one of the multi skills.
skill_name	Skill name associated with the problem. For the skill builder dataset, different skills for the same data record are represented in different rows. Thus, if a student answers a multi-skill question, this record is duplicated several times, and each duplication is tagged with one of the multi skills.
hint_count	Number of hints requested on this problem.
first_action	The type of the first action: attempt or ask for a hint.
opportunity	The number of opportunities the student has to practice this skill. For the skill builder dataset, opportunities for different skills of the same data record are represented in different rows. Thus, if a student answers a multi-skill question, this record is duplicated several times, and each duplication is tagged with one of the multi-skills and the corresponding opportunity count.
opportunity_original	The number of opportunities the student has to practice this skill counting only original problems. For the skill builder dataset, original opportunities for different skills of the same data record are represented in different rows. Thus, if a student answers a multi skill question, this record is duplicated several times, and each duplication is tagged with one of the multi-skills and the corresponding original opportunity count.

Algebra I 2005-2006 KDD Cup 2010 EDM Challenge is another benchmark dataset for learner modeling. The dataset was gathered within an Intelligent Tutoring System (ITS) from The Cognitive Tutors suite of tutors provided by the Carnegie Learning Inc. and hosted by the PSLC DataShop. This dataset was collected from three high schools for an entire year and contains most of the main elements of ASSISTment (e.g., student id, start time, duration, attempt, correct, and problem name), except for the skill_id column shown in Table 3. Thus, in the present work, we used the text skill name (i.e., the KC column in Table 4) as the equivalent of the column skill_id, which was represented as an integer in the ASSISTment datasets. Similar to ASSISTment, students must master a skill to progress to the next skill. This dataset is the second largest of all three datasets employed in this work. Table 4 presents the main columns of the dataset.

Table 4

Summary of the Algebra I 2005-2006 KDD Cup 2010 EDM Challenge

Column	**Descriptions**
Anon Student Id	Unique, anonymous identifier of a student solving the problem.
Problem Hierarchy	Hierarchy of curriculum levels containing the problem.
Problem Name	Unique identifier of a problem.
Problem View	Total number of times the student encountered the problem.
Step Name	Each problem consists of one or more steps. The step name is unique within each problem, but there may be collisions between different problems, so the only unique identifier for a step is the pair of Problem Name and Step Name.
Duration (sec)	Elapsed time of the step in seconds, calculated by adding all of the durations for transactions that were attributed to the step. It can be null (if step start time is null).
Correct First Attempt	Tutor's evaluation of the student's first attempt on the step. 1 = Correct on the first attempt. 0 = Incorrect on the first attempt. This is used for prediction.

Attempt at Step	Number of attempts on this problem.
Incorrects	Total number of incorrect attempts by the student on the step.
Hints	Total number of hints requested by the student for the step.
Corrects	Total correct attempts by the student for the step. It increases if the step is encountered more than once.
KC (KC Model Name)	The identified skills that are used in a problem, where available. A step can have multiple KCs assigned to it. Multiple KCs for a step are separated by ~~ (two tildes). Since opportunity describes practice by knowledge component, the corresponding opportunities are similarly separated by ~~.
Opportunities	A count that increases by one each time the student encounters a step with the listed KC. Steps with multiple KCs will have multiple opportunity numbers separated by ~~.

The Open Learning Initiative (OLI) Engineering Statics - Fall 2011 is a computer learning system developed at Carnegie Mellon University. The OLI provides online engineering static courses and assessments at college levels. The OLI embeds assessment into instruction and collects real-time data of student use. We used the OLI Engineering Statics - Fall 2012 dataset accessed via DataShop (Koedinger et al., 2010). This dataset is the smallest of all three datasets employed in this work. Table 5 presents the main columns of this dataset.

Table 5

Summary of the Open Learning Initiative (OLI) Engineering Statics - Fall 2011

Column	**Descriptions**
Anon Student Id	Unique, anonymous identifier of a student solving the problem.
Problem Hierarchy	Hierarchy of curriculum levels containing the problem.
Problem Name	Unique identifier of a problem.
Problem View	Total number of times the student encountered the problem.
First Action	Type of the first action: attempt or ask for a hint.
Step Name	Each problem consists of one or more steps (e.g., "find the area of rectangle ABCD" or "divide both sides of the equation by x"). The step name is unique within each problem, but there may be

	collisions between different problems, so the only unique identifier for a step is the pair of Problem Name and Step Name .
Step Duration (sec)	Elapsed time of the step in seconds, calculated by adding all of the durations for transactions that were attributed to the step. Can be null (if step start time is null).
Attempt Level	Number of attempts on this problem.
correct	Tutor's evaluation of the student's first attempt on the step. 1 = Correct on the first attempt. 0 = Incorrect on the first attempt. This is used for prediction.
KC (Fall 2011)	The identified skill that is used in a problem, where available. This dataset adopts a single KC model.

Prior to model training, we performed data cleaning and preprocessing in R using the *dplyr* library (Wickham and Francois, 2014). We removed students who practiced fewer than two KCs or items, and interactions containing missing KCs (i.e., denoted as *NA* in the dataset). In addition, learners who had fewer than ten interactions or tagged with no KCs were also removed, according to the common procedure followed by the other studies employing these datasets (Gervet et al., 2020). Among the three datasets, some interactions are tagged with a single KC, whereas others are tagged with several KCs. We regarded combinations of KCs as constituting new KCs, according to the procedure outlined in most previous research.

Learner Models

Baseline Models

The baseline models for the current study are Rasch model (IRT based), BKT (HMM based), DKT-RNN, and DKT-RNN-LSTM (both are deep learning models). The detailed descriptions of the baseline models can be found in the previous Chapter on learner models.

The DKT with Contextualized Information (DKT-CI)

The four baseline models all used static or sequential product data as the input vectors (i.e., item ID/skill ID vs. student response where 1 denotes correctness and 0 denotes incorrectness) to predict the probabilities of answering the next item correctly. With the advent of learning technologies, the ITSs not only stored student product data but also collected contextualized information including duration of the tasks, number of hints requested, skill hierarchy, and total trials on the tasks. Thus, this study implemented a deep knowledge tracing model with contextualized information (DKT-CI). Specifically, we incorporated the contextualized information to a standard DKT using embedding techniques (Mikolov et al., 2013) to learn the feature representations of the following columns in the three datasets: the time (or duration) spent on a problem, the first action took to answer an item, the total number of attempts (or opportunities) on a step, and the total number of hints requested from the system. Then, the DKT-CI concatenated the embedding layer of contextualized information and the input vector of student product data as in the standard DKT. Figure 7 illustrates the conceptual representation of the DKT-CI that incorporates both the product data (x_t) and the contextualized information related to task activities at different time steps (i_t).

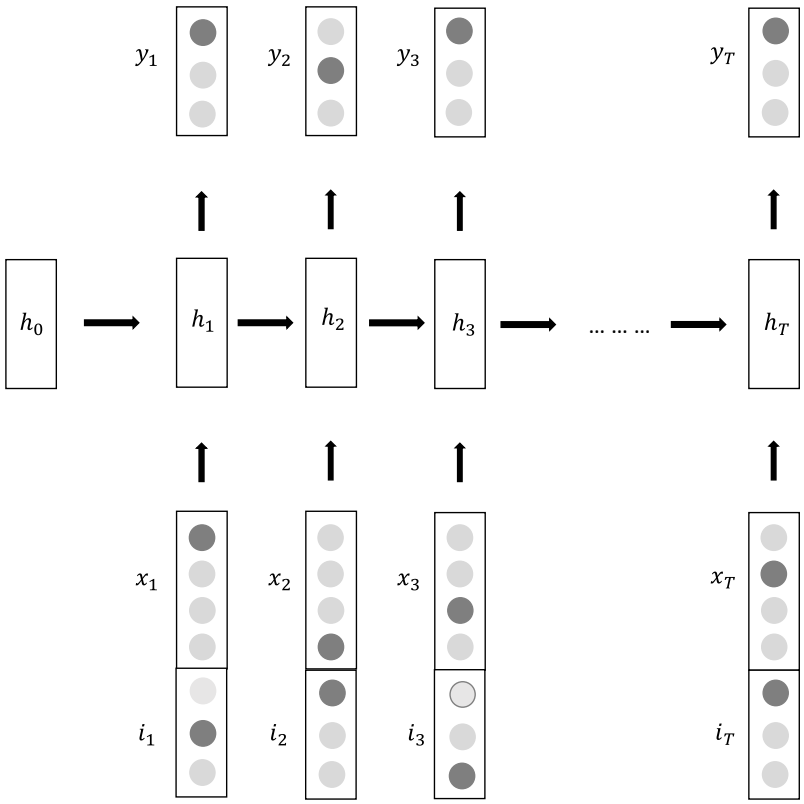

Figure 7. *Representation of Deep Knowledge Tracing with Contextualized Information*

Implementation of the Learner Models

IRT. The 1PL IRT model (i.e., the Rasch model) was implemented using the *glmer* function built in the *lme4* library in R (Bates et al., 2007). The generalized linear mixed model employed the family=binomial("logit") and it was fitted using the maximum likelihood with Laplace approximation.

BKT. The standard BKT model was implemented in C++ using the *hmm-scalable* algorithm (Yudelson et al., 2013). In the process of model training on the three selected datasets, we used the Baum-Welch method to estimate the parameters and adopted the default settings for the hyperparameters. Specifically, we set the initial probability of mastering a skill $P(L_0)$ = 0.5, the transition probability from non-mastery to mastery of a skill $P(T)$ = 0.4, the probability that unmastered skill is applied correctly $P(G)$ = 0.2, and the probability that a mastered skill is applied incorrectly $P(S)$ = 0.2.

DKT-RNN, DKT-LSTM, and DKT-CI. We implemented the standard DKT-RNN, its variants DKT-LSTM, and DKT-LSTM with contextualized information in Keras, using the TensorFlow backend (Abadi et al., 2016). We first performed grid search to select the best hyperparameters on the training set. Specifically, we chose among the *RMSprop*, *Adam,* and *Adagrad* optimizers with a learning rate of 0.001, batch size chosen from the set {5, 10, 32, 64, 128}, and dropout probability chosen from {0.2, 0.4, 0.6}. For the hidden layers, we selected the dimension in {50, 100, 150}, the number of recurrent layers in {1, 2, 3}, and the $\ell 1$ and $\ell 2$ regularizers in {0.01, 0.02, 0.05}. For the embedding layer, we selected the dimension in {250, 500, 1000}. After training and validation, we obtained the hyperparameters that yield the best results. Appendix 1 presents the best hyperparameter sets for DKT-RNN, DKT-LSTM, and DKT-CI on the three selected datasets.

Evaluation Metrics of the Learner Models

In the training process, we split the datasets into training (60%), validation (20%), and testing (20%). For comparisons among the algorithms, we computed the predictive accuracy using the Area Under the Curve (AUC), because the response variable is binary. The AUC plots the true-positive rate against the false-positive rate at all decision thresholds. The AUC is a commonly used evaluation metric for learner models in which a score of 1 reflects a perfect discrimination and 0 reflects no discrimination. Following related studies, we also reported the Root Mean Squared Error (RMSE) to assess the squared error of prediction.

Summary of the Learner Model Experiment

Previous studies mainly focused on providing item-level hints or feedback, with only few examining personalized skill-level feedback. Incorporating learner models into feedback-generation systems addresses this gap by providing higher-order diagnosis based on learners' latent skill mastery states and knowledge transfer as measured by the learner models. In

addition, the proposed learner model incorporates contextualized information to the product data for more accurate calibration of student latent abilities.

Feedback Augmentation

Previous studies mainly provided automated feedback using expert-derived feedback templates (Zhu et al., 2020) or demonstrated the correct answer (Singh et al., 2013), where the feedback is greatly limited regarding the quantity, diversity, and communication efficiency. The present study used an unsupervised sentence generation method to augment expert-derived feedback templates for digital learning and assessment systems. The feedback generation phase includes three steps. In Step 1 (Corpus Development), we developed a corpus of feedback, which included positive and negative feedback along with the insertion position for KCs. In Step 2 (Corpus Augmentation), we expanded the feedback corpus by augmenting the feedback templates using the *Constrained Sentence Generation by Metropolis-Hastings Sampling* method (CGMH; Miao et al., 2019). Most sentence generation methods are based on RNNs so that a sentence can only be generated in a sequential order from left to right, whereas the MH sampler for sentence generation allows more flexible manipulations of sentences (Miao et al., 2019). Moreover, the unsupervised method does not require parallel corpus for sentence generation. In Step 3 (Feedback Generation & Provision), we generated item-student specific feedback based on students' performance on the Intelligent Tutoring System.

Implementation of the Feedback Augmentation System

Step 1: Expert-Derived Feedback Templates. Table 6 presents a few examples of feedback templates that we devised for our corpus. A positive or negative feedback was retrieved from the corpus based on: (1) a learner's performance and (2) a performance threshold.

Table 6

The Feedback Template Corpus, Including Both Positive and Negative Feedback Messages and Insertion Positions for Knowledge Components (KCs)

Positive	Negative
You are on your way to mastering [Insertion].	[Insertion] could use some focused practice.
It looks like you have a good handle on [Insertion].	Consider practicing [Insertion] a bit more.
Great work so far on mastering [Insertion].	[Insertion] requires a bit more attention.
Great job mastering [Insertion].	See if you can fine tune your skill on [Insertion].
Keep up the good work on [Insertion].	You will master [Insertion] with a bit more practice.

Step 2: Feedback Augmentation. To expand the corpus, we adopted the *Constrained Sentence Generation by Metropolis-Hastings Sampling* method (CGMH; Miao et al., 2019) to perform unsupervised paraphrase generation. The CGMH is a subtype of Markov chain Monte Carlo (MCMC; Geyer, 1992) methods and it supports more flexible operations on word tokens in a sentence space. Thus, it is easier to generate content with constraints and varying sentence lengths. Miao et al. (2019) tested the CGMH on three generation tasks: keywords-to-sentence generation with hard constraints, paraphrase, and error correction with soft constraints. In the present research, we implemented the unsupervised paraphrasing to augment the feedback corpus. Specifically, we first trained a language model based on the IMDB review corpus (Maas et al., 2011) that contains 25,000 positive and 25,000 negative reviews. Then, we performed the paraphrase generation.

A Markov model is used to train the language model on the selected corpus. The Markov Chain is commonly used to model natural language as a function of the probability that a word appearing in position *n* is only dependent on the previous $z \in [1, n-1]$ such that:

$$p(w_1, w_2, \dots, w_n) = p(w_1)\, p(w_2|w_1), \dots\dots, p(w_n|w_{n-z}, \dots, w_{n-1}),$$

where $p(w_1, w_2, \dots, w_n)$ refers to the probability of a specific sentence based on the trained corpus, that is, the joint probability of all words within the sentence. In the present research, we used forward-backward dynamic programming to train the language model.

In Step 2 (feedback paraphrase), we performed the CGMH task of unsupervised sentence paraphrasing. The CGMH is concerned with a goal of stationary distribution that defines the sentence distribution sampled from the corpus and three actions, namely, replacement, insertion, and deletion. Specifically, $\pi(x)$ was set as the distribution from which we plan to sample sentences, where *x* denotes a particular sentence and x_0 refers to the feedback template that is fed to the algorithm at time step 0. The MH sampler either accepts or rejects a word from the given distributions $\pi(x)$ to finally form a desired joint distribution of all words based on a predefined stationary distribution. The process is intuitive, as it mainly involves two actions: accepting or rejecting a word monitored by the acceptance rate α:

$$\alpha = \min\{1, \frac{\pi(x')g(x_{t-1}|x')}{\pi(x_{t-1})g(x'|x_{t-1})}\}$$

At time step *t*, the word sampling is conducted to update the previous state *x* to a candidate distribution x' from a proposed distribution $g(x'|x_{t-1})$, where x_{t-1} refers to the distribution from previous step (*t*-1), thus $x' = x_t$. Therefore, α determines the acceptance or rejection of a sample. In our paraphrase generation, the desired distribution denotes the most likely and logical sentence to the original sentence fed to the model.

At each step, a selected word in the sentence will be randomly updated by the actions such as insertion, deletion, and replacement, where the respective probabilities are

$[p_{insert}, p_{delete}, p_{replace}]$. At the first time step, these probabilities are set as being equal. At the following step, if *Replacement* is applied on a selected word w_m in a sentence $x = [w_1, w_2, \dots, w_{m-1}, w_m, w_{m+1}, \dots, w_n]$, then the conditional probability of choosing w_m^{new} to replace w_m to form candidate sentence x' from *x* can be computed as:

$$g_{replace}(x) = \pi(x_{-m}) = \frac{\pi(w_1, w_2, \dots, w_{m-1}, w_m^{new}, w_{m+1}, \dots, w_n)}{\sum_{w \in V} (w_1, w_2, \dots, w_{m-1}, w, w_{m+1}, \dots, w_n)},$$

where *V* refers to the vocabulary, and w_m is the selected word. If, on the other hand, *Insertion* is applied, an additional step of inserting a placeholder will be conducted before taking the action *Replacement*, and then a real word will be sampled to replace the placeholder token with the *Replacement* token. Finally, if *Deletion* is applied, the w_m word selected will be deleted, and $g_{deletion}(x)$ =1 if $x' = [w_1, w_2, \dots, w_{m-1}, w_{m+1}, \dots, w_n]$, and 0 otherwise.

At the end of all operations, we want to achieve a stationary distribution of $\pi(x)$, such that $\pi(x) \propto p_{LM}(x) \cdot X_{match(x|x_*)}$, where $p_{LM}(x)$ is the probability of the language model and $X_{match(x|x_*)}$ is a matching score defined by word embedding similarity (Pennington, Socher, & Manning, 2014). The value of $X_{match(x|x_*)}$ is 1, if the constraints are met, and 0 otherwise.

Evaluation Metrics of the Feedback Augmentation System

Previous studies mainly used the BLEU score to evaluate the quality of a sentence (Papineni et al., 2002). Miao et al. (2019) used an alternative method to compare the BLEU score of the paraphrases (i.e., BLEU-ref) against the original input sentence (i.e., BLEU-ori) to evaluate the accuracy of the paraphrasing. The model is regarded as performing well if the BLEU-ref is high, whereas BLEU-ori is low. However, in the current study, we used the MH sampler to generate diverse feedback. Therefore, the evaluation metrics used in the previous NLP studies are not appropriate for this study. Since the purpose is to generate fluent, semantically-related, and diverse feedback, we used the negative likelihood (NLL) of the

sentences to evaluate the feedback fluency using the Reuters corpus released by NLTK modules. The lower the NLL is, the more fluent the sentences are. In addition, we invited 2 volunteers to rate the quality of feedback in terms of the paraphrased feedback regarding the fluency and relatedness at a scale of 0-1 and a grain size of 0.1, and the higher the scores are, the more fluent and related the feedback sentences are. The inter-rater reliability was reported using Kendall's Kappa (Kendall, 1938). Kendall rank correlation coefficient is a non-parametric rank order correlation that measures ordinal correlation among raters. Kendall's Kappa ranges from 0 to 1, with higher values indicating higher levels of agreement. The null hypothesis of Kendall's Kappa is that there is no agreement among the judges. The raters are regarded rating consistently if the null hypothesis is rejected.

Summary of the Feedback Augmentation System

Our recent literature review revealed that text-based feedback was more effective in improving performance (Lu et al., 2021). However, it is laborious to manually devise a large amount of quality feedback. Compared with sentence-generation supervised-learning methods, the unsupervised CGMH sentence-paraphrasing method can augment the expert-driven feedback template corpus by generating feedback phrases with higher efficiency and flexibility. Thus, the proposed method is promising in promoting text-based feedback generation within ITSs.

Synthesis of the Learner Models and Feedback Generation System

Figure 8 illustrates the synthesis process of the proposed learner modeling and feedback generation approach. Specifically, at the experiment stage, an augmented corpus of positive and negative feedback is constructed. Meanwhile, learner models are trained based on the process data of student interactions with the ITS. At the tutoring stage, the proposed framework generates feedback based on student performance on items and the threshold of mastering the KC underlying the items. Students can receive the individualized feedback that

includes the specific KC and valence (i.e., positive or negative) based on their performance. The purpose of this framework is to automatically provide specific feedback on the underlying KCs for users to reflect on their past performance.

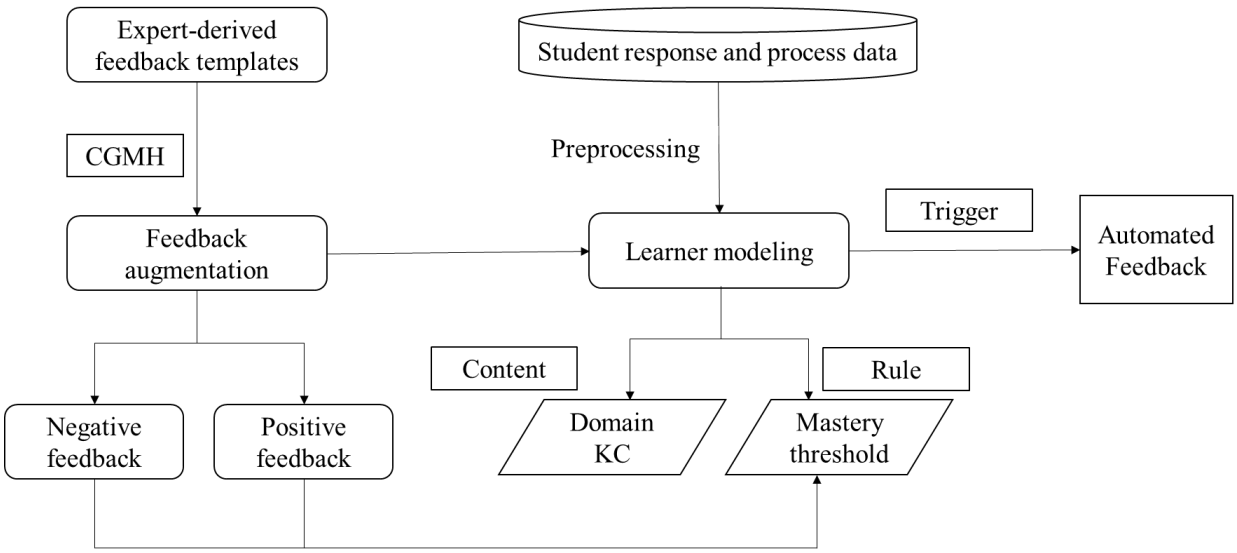

Figure 8. *The Framework of the Feedback Generation for the Learner Models*

Chapter Summary

Chapter 4 described the proposed methodological framework of automated feedback generation for different learner models. It first detailed the overall framework and structure of the automated feedback system. Then, the implementations, model setup, training, and evaluation processes of the learner models and the feedback augmentation methods were presented, followed by the synthesis of learner modeling and automated feedback generation. The next chapter presents the results and discussion.

Chapter 5 Results and Discussion

This chapter is organized as follows. First, it compares the accuracy of the learner models on each of the three selected datasets. Second, this chapter reports the output representations of the different learner models. Specifically, example outputs from the three large-scale datasets (ASSISTment 2009-2010, KDD Algebra 05-06, and OLI Fall 2011) are presented with interpretations in detail. Third, evaluations of the feedback augmentation methods are presented. Then, the feedback generation processes are illustrated for different learner models. The last section discusses the learner model selection and feedback generation model selection for various educational purposes and contexts.

Learner Model Prediction Performance

The performances of the IRT, BKT, DKT-RNN, DKT-LSTM, and the proposed DKT-CI on each of the three selected datasets are presented in Table 7. The DKT-CI performs best on the larger datasets including the ASSISTment 2009-2010 (AUC = 0.859, RMSE = 0.370) and the KDD Algebra I 2005-2006 (AUC = 0.834. RMSE = 0.472), which contain 4,163 and 574 students, respectively. Apart from DKT-CI, the DKT-LSTM also yielded satisfactory performance on the two larger datasets, with ASSISTment 2009-2010 having an AUC = 0.842, RMSE = 0.386, and KDD Algebra I 2005-2006 having an AUC = 0.802. RMSE = 0.346. The DKT-CI outperformed the DKT mainly because it incorporated the process data including the response time, number of attempts, number of hints, and actions took collected by the ITSs, whereas DKT only modeled the product data. Within the proposed automated feedback generation system, the DKT-CI is preferable for it not only improved the prediction performance of the evaluation model, but also had the potentials to inform when to provide feedback prompted by the response time or the number of attempts.

The BKT has the best predictive performance on the smallest dataset (i.e., the OLI Fall 2011 dataset containing 333 students), with an AUC of 0.802, followed by the IRT

(AUC = 0.791, RMSE = 0.514). The OLI Fall 2011 is regarded as too small for deep learning models and only yielded a lower AUC of 0.686 (RMSE = 0.523) for DKT-RNN and an AUC of 0.700 (RMSE = 0.523) for DKT-LSTM. The DKT-CI has the best performance among the deep learning models (AUC = 735, RMSE = 0.397); however, it is still not comparable to the BKT and IRT. The big gap of AUC between BKT and DKT on OLI datasets reveals that BKT tends to overfit on small datasets and yield high AUC, whereas DKT circumvents the overfitting issue by using regularization and dropout techniques. Thus, we conclude that the probabilistic approach performs faster and more accurate estimations on smaller-scale datasets. By contrast, the deep learning approach better exploits the temporal information within large-scale datasets, and thus, makes more accurate predictions.

Table 7

Results of The Model Performance on the Test Dataset for the Five Algorithms.

Algorithm	**AUC**	**RMSE**
ASSISTment 2009-2010		
IRT	0.770	0.383
BKT	0.761	0.406
DKT-RNN	0.833	0.337
DKT-LSTM	0.842	0.386
DKT-CI	0.859	0.370
Algebra I 2005-2006		
IRT	0.757	0.419
BKT	0.791	0.393
DKT-RNN	0.708	0.338
DKT-LSTM	0.802	0.346
DKT-CI	0.834	0.472

OLI Fall 2011		
IRT	0.791	0.514
BKT	0.802	0.367
DKT-RNN	0.686	0.523
DKT-LSTM	0.700	0.572
DKT-CI	0.735	0.397

Output Representation of Different Learner Models

Output Representation for IRT

The output of learner models is the input of the template-based generation. Therefore, it is crucial to understand the output representation of learner models rather than simply compare their predictive accuracy. The output of the Rasch model yields several item parameters including the item discrimination fixed to 1. The difficulty values normally range between -2.5 and 2.5. If the difficulty level is 0, it means that the correct rate for an item is 50%. Any negative values indicate that the item was easier than average (i.e., more than 50% participants scored correctly), and positive values indicate more difficult items. The higher the value, the more difficult the item. If the item difficulty parameters are larger than 2.5 or smaller than -2.5, the item is deemed as too difficult or too easy for the candidates. Table 8 presents the item parameter estimations for the first six items on the ASSISTment 2009-2010 dataset. Results show that the three items corresponding to the skill Box and Whisker are considerably easier for the students, with item difficulty ranging from -2.86 to -1.13. The three items corresponding to Circle Graph are more difficult than Box and Whisker, with item difficulty ranging from -0.44 to -0.20.

Table 8

The Item Parameter Estimations from the IRT Output on the ASSISTment 2009-2010 Dataset

Item	Item Discrimination	Item Difficulty
Box and Whisker Item 1	1	-1.13
Box and Whisker Item 2	1	-1.66
Box and Whisker Item 3	1	-2.86
Circle Graph Item 1	1	-0.44
Circle Graph Item 2	1	-0.39
Circle Graph Item 3	1	-0.20

Output Representation for BKT

The goal of the present study is to employ the output of different learner models to generate KC-level feedback that scaffolds learning. Tables 9, 10, and 11 present the output representation of BKT on the ASSISTment 2009-2010, Algebra 2005-2006, and OLI Fall 2011 datasets, respectively. The initial probability $P(L_0)$ was set to 0.5 for the initial states. The transition probability $P(T)$ reflects the extent that students can learn and improve from consecutive practice with items on a given KC. For example, in Table 9, students improve most on the *Scatter Plot* KC ($P(T)$ = 0.464) after consecutive problem trials but show the least improvement on the *Circle Graph* KC ($P(T)$ = 0.059). The slip parameter $P(S)$ represents the probability of students making a mistake when applying KCs, even if they have mastered those KCs. The guess parameter $P(G)$ represents the probability of a student's correct response by guessing. Finally, students' mastery of KCs reflected in $P(Mastery)$ is used as a reference for feedback generation. Specifically, we set 0.55 as the threshold of mastery. The threshold is flexible, and it is determined based on each dataset and student proficiency level.

Table 9

The First Ten Knowledge Components (KCs) from the BKT Output on the ASSISTment 2009-2010 Dataset

Skill	$P(L_0)$	$P(T)$	$P(S)$	$P(G)$	$P(Mastery)$
Box and Whisker	0.5	0.180	0.202	0.239	0.812
Circle Graph	0.5	0.059	0.300	0.133	0.567
Histogram as Table or Graph	0.5	0.213	0.260	0.190	0.811
Number Line	0.5	0.075	0.300	0.300	0.636
Scatter Plot	0.5	0.464	0.067	0.297	0.640
Stem and Leaf Plot	0.5	0.141	0.239	0.156	0.717
Table	0.5	0.117	0.214	0.147	0.815
Venn Diagram	0.5	0.066	0.132	0.078	0.750
Mean	0.5	0.164	0.300	0.081	0.604
Median	0.5	0.091	0.205	0.289	0.593

Table 10

The First Ten Knowledge Components (KCs) from the BKT Output on the KDD Algebra I 2005-2006 Dataset

	$P(L_0)$	$P(T)$	$P(S)$	$P(G)$	$P(Mastery)$
Eliminate Parentheses	0.5	0.044	0.145	0.300	0.756
Remove Constant	0.5	0.014	0.188	0.300	0.633
Remove Coefficient	0.5	0.031	0.183	0.300	0.690
Remove Positive Coefficient	0.5	0.042	0.079	0.300	0.757
Add/Subtract	0.5	0.052	0.156	0.300	0.717

Multiply/Divide	0.5	0.097	0.082	0.300	0.789
Consolidate Vars with Coefficient	0.5	0.016	0.095	0.300	0.914
Combine-like-terms	0.5	0.017	0.300	0.300	0.735
Calculate Eliminate Parentheses	0.5	0.425	0.178	0.233	0.869
Simplify Fractions	0.5	0.024	0.300	0.300	0.642

Table 11

The First Ten Knowledge Components (KCs) from the BKT Output on the OLI Fall 2011 Dataset

	$P(L_0)$	$P(T)$	$P(S)$	$P(G)$	$P(Mastery)$
Represent Interaction Spring	0.5	0.357	0.000	0.003	0.311
Identify Interaction	0.5	0.248	0.000	0.001	0.237
Gravitational Forces	0.5	0.460	0.000	0.004	0.902
Distinguish Rotation Translation	0.5	0.667	0.000	0.014	0.833
Motion Dependence on Force	0.5	0.422	0.000	0.003	0.343
Rotation Sense of Force	0.5	0.424	0.000	0.003	0.367
Find Moment Arm	0.5	0.150	0.000	0.000	0.035
Simple Step	0.5	0.154	0.000	0.001	0.232
Moment Sign Sense Relation	0.5	0.422	0.000	0.002	0.246
Represent Interaction Contacting Body	0.5	0.338	0.000	0.002	0.284

Output Representation for DKT and Its Variants

BKT is a probabilistic approach based on a two-layer HMM and it yields probabilities of learning, slipping, guessing, and mastery. On the other hand, DKT is a deep learning

approach (i.e., recurrent networks) and it yields vectors of predictions at different time steps. Specifically, the input x_t is a vector of length equal to the number of problems. If the number of problems attempted varies, then padding techniques are used. Each entry of output y_t represents the predicted probability that the student would solve that problem correctly at time step *t*. Table 12 presents the exact predicted probabilities of a student's (with ID = 1) correct attempts of the five KCs (32: Ordering Positive Decimals; 33: Ordering Fractions; 45: Subtraction Whole Numbers; 55: Absolute Value; 98: Equation Solving Two or Fewer Steps) after ten practice rounds. The first column is a skill-correctness dyad.

The predicted probabilities of the five KCs are updated after each practice round. Figure 9 plots the knowledge tracing heatmap of a student's predicted responses while solving 50 ASSISTment exercises. The x-axis presents the student's actual attempts (1: correct; 0: incorrect) on a set of KCs, in the format (skill_ID: KC). The y-axis shows the predicted probabilities of a student's correct attempts of the KC after an exercise. Figure 9 not only provides information of student mastery profiles after completing all the exercises but also plots the knowledge transfer of the student during the whole practice process. Appendix 3 presents a complete heatmap of this student with all the KCs attempted.

Table 12

The Adaptive DKT Output for 10 Actions and 5 KCs for the ASSISTment09-10 Dataset (A complete output of predicted probabilities can be found in Appendix 2)

Attempts (skill ID, correct or not)	**32: Ordering Positive Decimals**	**33: Ordering Fractions**	**45: Subtraction Whole Numbers**	**55: Absolute Value**	**98: Equation Solving Two or Fewer Steps**
(55,1)	0.4173	0.5133	0.3680	0.4703	0.5641
(45,1)	0.3574	0.8526	0.5014	0.3749	0.4338
(55,1)	0.3714	0.7322	0.4427	0.4686	0.4776
(55,1)	0.5310	0.6108	0.5783	0.5147	0.5209
(55,0)	0.2793	0.6292	0.4974	0.5713	0.4899

(45,0)	0.3221	0.6397	0.5526	0.3927	0.3128
(98,1)	0.3388	0.5687	0.4477	0.4094	0.5792
(98,1)	0.3669	0.6980	0.4929	0.4686	0.5741
(98,1)	0.4644	0.7553	0.5098	0.5074	0.6554
(33,0)	0.2838	0.1308	0.3025	0.4147	0.4536

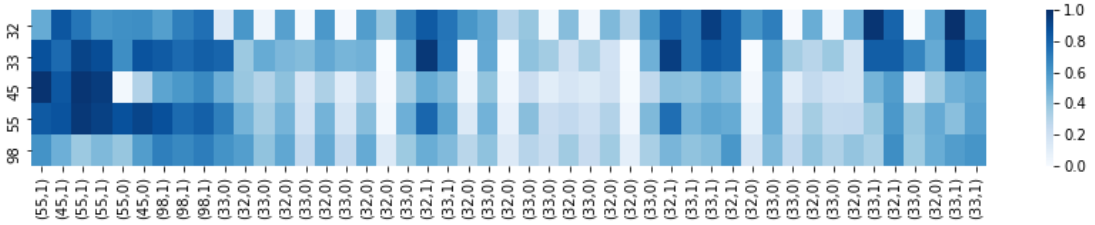

Figure 9. *DKT Output (skill-correctness) Heatmap*

Automated Feedback Generation for Learner Models

This section presents the augmented feedback corpus and demonstrates the feedback generation for learner models. Examples of feedback for standard BKT, DKT, and its variants are provided to illustrate the automated feedback provision process at skill (KC) level.

The Augmented Feedback Corpus

Based on the 30 expert-derived feedback templates, the CGMH automatically paraphrased them into 300 augmented feedback sentences. To evaluate the quality of the paraphrased feedback, we first randomly selected 50 paraphrased feedback sentences and evaluated them using NLL and human ratings. Table 13 shows the NLL and human-rater evaluations of the generated sentences regarding Fluency and Relatedness on a scale of 0 to 1. The higher the scores, the more fluent and related the generated feedback sentences, and a score of 0.5 indicates acceptable quality. The results revealed that the MCMC method can generate generally fluent and semantically relevant sentences (Human Rating: Fluency =

0.65; Human Rating: Relatedness = 0.59). The bottom of Table 13 presents an example of augmented feedback based on an expert-derived template.

Table 13

Paraphrase Feedback Generation Performance and Examples (n=50)

Evaluation Metrics	**Measures**
NLL	11.28
Human Rating: Fluency	0.65
Human Rating: Relatedness	0.59
Examples of feedback	
Original feedback templates	**Paraphrased feedback templates**
You excel at [Insertion].	You are strong at [Insertion].
	You demonstrated best on [Insertion].
	You master [Insertion].
	You do profound job in [Insertion].
Great work on [Insertion].	Profound job in [Insertion].
	Good job in [Insertion].
	Great work for [Insertion].
	Good work on this [Insertion].

To further evaluate the quality of the augmented feedback, we did a simple filtering of the augmented feedback corpus and compared the filtered feedback (n=120) with the expert-derived feedback (n=30) on human ratings using Welch Two Sample t-test. Results revealed that there are no significant differences on the two raters' ratings on Fluency between the paraphrased feedback and the expert-derived feedback. Regarding feedback Relatedness, Rater 1 assigned significantly higher scores to expert-derived feedback [$t(86.06) = 3.58$, $p < .01$], compared with augmented feedback. However, there is no significant difference on

Relatedness scores assigned by Rater 2. The results confirm that the augmented feedback is fluent and semantically related compared with expert-derived feedback. The inter-rater reliabilities show that the inter-rater reliabilities are 0.80 [$X^2(149) = 238$, $p<.01$] on fluency and 0.76 on relatedness [$X^2(149) = 225$, $p<.01$], validating that the two independent raters demonstrated consistent ratings.

Table 14

Comparisons of Human Ratings Between Expert-Derived and Augmented Feedback

Measure	**Expert-derived Feedback (n=30)**	**Augmented Feedback (n=120)**	***Diff.***	***df***	***t***	***p***
Fluency Rater 1	0.84	0.80	0.04	41.47	0.57	0.57
Fluency Rater 2	0.83	0.78	0.05	40.45	0.81	0.42
Relatedness Rater 1	0.97	0.80	0.17	86.06	3.58	<.01
Relatedness Rater 2	0.68	0.71	0.03	38.73	-0.39	0.70

Note: *Diff.* = Difference of mean scores; *df* = degree of freedom; *t* = t statistic; *p* = *p* value.

Feedback for Different Learner Models

Feedback for BKT. For BKT, we used expert-derived KC thresholds for feedback generation. Mastery thresholds for KCs were set by domain experts for students with different levels of knowledge proficiency. After setting the mastery threshold, we retrieve the positive feedback template from the augmented corpus if $P(Mastery)$ is estimated as larger than the threshold, and we retrieve the negative feedback template from the corpus if $P(Mastery)$ is estimated as less than the threshold. Then, a feedback message with a specific KC and valence (i.e., positive or negative) will be triggered and provided for the learners. Figure 10 presents an illustration of automated feedback provision for BKT, where

$P(Mastery)$ is set as 0.55. If a student practices the KC "Box and Whisker" and the BKT estimates that the $P(Mastery)$ of the student is updated to 0.812 from the previous practice, the system is notified that the student has successfully mastered this skill and is triggered to retrieve a positive feedback template from the augmented feedback corpus. Then, the keyword "Box and Whisker" is inserted to the placeholder in the feedback template. At the end of the feedback generation stage, the feedback on "Box and Whisker" is generated and provided for the student.

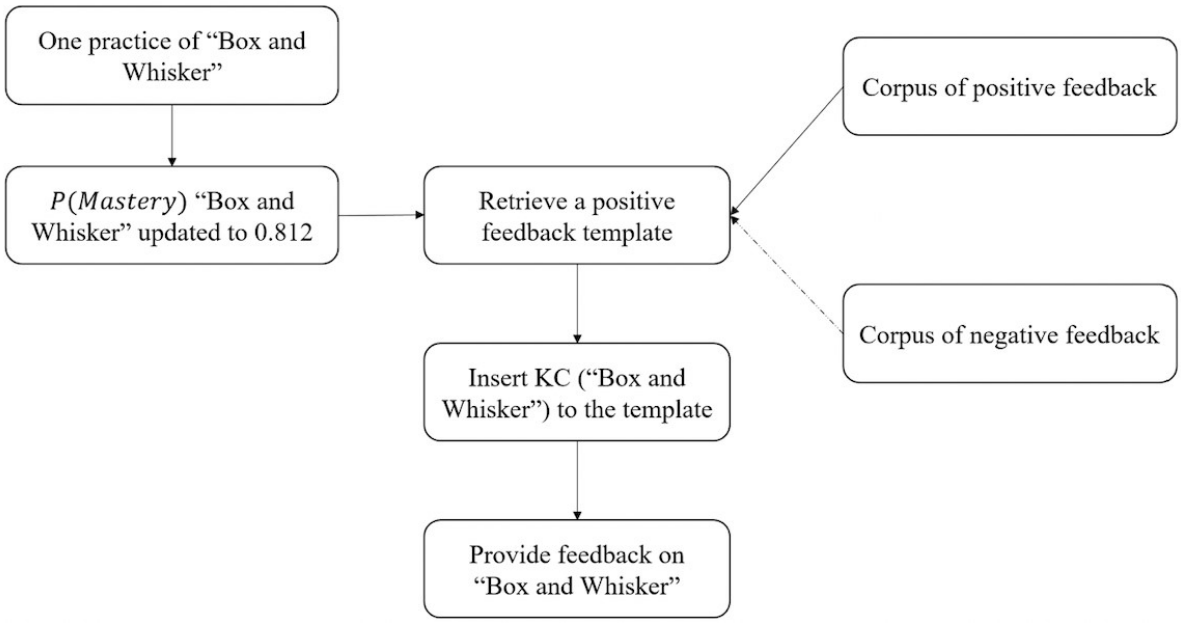

Figure 10. *Feedback Generation for BKT on KC "Box and Whisker"*

Feedback for DKT and its variants. For DKT and its variants, we extract the predicted probabilities for each KC at a time step, retrieve positive or negative feedback templates based on the mastery thresholds (like BKT), and insert the KCs in the templates. A student can choose to receive feedback on any KC at any time step since their mastery of KCs are updated stepwise. Unlike BKT, which can only model one KC at a time, the feedback for DKT will contain multiple KCs and generate adaptive feedback for every KC at each time step. Figure 11 details an example of the feedback generation process for DKT. In

Figure 11, a student's predicted mastery probabilities of all KCs are updated after practicing an item associated with the "Absolute Value" KC. More specifically, the predicted probability of answering an item associated with "Absolute Value" KC correctly is updated to 0.470, for "Ordering Positive Decimals" is updated to 0.417, "Ordering Fractions" is updated to 0.513, "Subtraction Whole Numbers" to 0.368, and "Equation Solving Two or Fewer Steps" to 0.564. In this scenario, the threshold for mastery is also set as 0.55. Therefore, four negative and one positive feedback messages will be retrieved from the corpus of negative and positive feedback, and all five pieces of feedback are triggered by student performance updates.

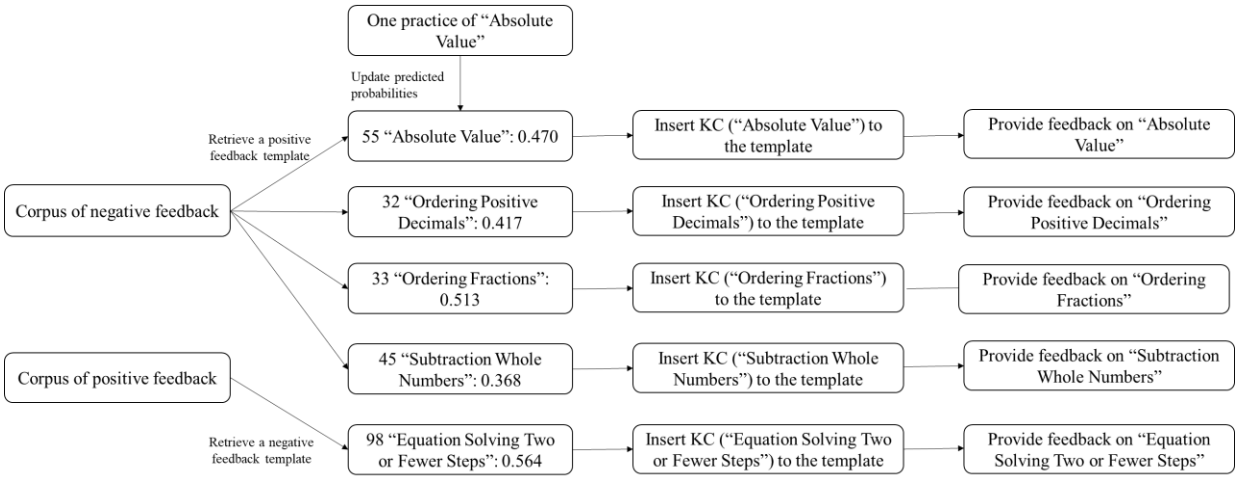

Figure 11. *Feedback Generation for DKT after Practice of the KC "Box and Whisker"*

Discussion

Providing feedback is essential for improving performance. However, understanding the context of learning is fundamental prior to learner modeling and feedback generation. Thus, it is important to conduct in-depth investigations regarding which learner model and corresponding output representation should be applied to a certain context (i.e., system, student, or domain) and what forms of feedback (i.e., summative vs. formative, static vs. adaptive, single KC vs. all KCs) students need for knowledge acquisition. From a comparison among the learner models' analytical methods, performance, output representations, and the

feedback generation process for different learner models, we found the following elements important for the implementation of automated feedback generation frameworks.

Learner Model Selection

The present study argues that the learner model and output representation are the foundation for automated feedback generation. Thus, caution is needed when choosing the most appropriate learner model for feedback generation. The comparison of model performance among IRT, BKT, DKT, and its variants on the three datasets reveals that the probabilistic approaches IRT and BKT outperform the DKT deep learning approach on smaller datasets (OLI Fall 2011), whereas deep learning (DKT-CI, DKT-RNN, and DKT-LSTM) boosts predictive accuracy on larger datasets (ASSISTment 09-10). Further, the DKT that incorporates both the product data and the contextualized information yields the best performance among all the implemented models on larger datasets.

Therefore, we conclude that BKT requires easily adapts to immediate KC-level feedback generation. It is more suitable for small-scale datasets including contexts such as local computer-based tutoring and assessment (e.g., OLI datasets) or classroom-based tutoring and assessment. On the other hand, DKT is powerful on modeling large datasets with higher prediction accuracy. It is more suitable to large-scale datasets extracted from open online learning systems, such as the ASSISTments and Cognitive Tutor datasets. In technology-enhanced education, digital tutoring and assessment systems can capture both product data and process data, such as action and response time. The DKT-CI implemented in the present study used both product data and process data as the input of the prediction model and demonstrated better performance than the model with fewer input features. More investigations should be conducted regarding the trade-off between better calibrations of students' abilities and faster immediate personalized feedback.

Feedback Generation Model Selection

Most studies focused on improving the predictive performance of learner models, or optimally prompting hint generation/provision within ITSs, in which researchers examined when and whether to provide hints for learners (e.g., Maniktala et al., 2020). Few studies scrutinized the underlying assumptions and output representations of learner models, or how to provide feedback on learners' performance. The present study explains the output representations of IRT, BKT, and DKT in detail and interprets the meaning of the outputs and take-home messages for learners. In addition, the proposed feedback generation framework specifies how to prepare and provide adaptive feedback for learners based on different learner models. Specifically, IRT yields only item-level estimations. By contrast, BKT and DKT performs KC-level estimations. Thus, BKT and DKT are more suitable for feedback generation in structured domains at KC-level. In addition, BKT regards KCs as independent from each other, and it updates the probabilities of mastery for each KC separately, whereas DKT assumes the underlying KCs are interconnected with each other. BKT has a unique advantage over DKT, because it can both predict accurately, make inferences, and explain its decisions. Thus, it contributes to the understanding of learning at the specific skill level. Results of BKT can also be generalized to groups. By contrast, DKT considers the interconnected relations among KCs but can not trace back and recover the cumulative developmental process of a single KC. DKT better exploits the temporal information embedded in the datasets and models the sequential actions recorded in the log data. It updates the predicted probabilities of all KCs at a time whenever a learner is practicing on a KC. Thus, learners' complex knowledge transfer on all KCs can be plotted using DKT.

DKT is a pure data-driven approach for knowledge transfer pattern discovery, whereas BKT can be interpreted with learning theories. DKT can learn conceptual knowledge transfer and progression without domain-expert annotations. By contrast, BKT requires

experts to manually set thresholds for mastery/non-mastery. We recommend that researchers adopt BKT if they are interested in inferring the acquisition process of a specific KC and generalizing to a group to generate learning theories. By contrast, researchers could adopt DKT to depict the individualized complex knowledge transfer (both on a single KC and among all KCs) for a student. Thus, one implication from the present research is that DKT is better suited for individualized, adaptive formative feedback, whereas BKT for summative feedback. Moreover, instructors could combine both methods to achieve better teaching effectiveness in practice. For example, DKT with adaptive formative feedback can be used to guide the students to learn, whereas BKT with summative feedback can be used by instructors to induce the learning curve of specific skills.

Chapter Summary

Chapter 5 reported the classification and predictive accuracy of several baseline learner models and the proposed DKT-CI (i.e., DKT with Contextualized Information). The output representations of learner models were presented and compared regarding their interpretation, adaptiveness, and assumptions of the underlying KCs. Then, the data-driven feedback templates generated by the unsupervised feedback augmentation methods were demonstrated and evaluated. Based on the results of the learner model comparisons and the data-driven feedback evaluations, descriptions of the automated feedback generation processes for different learner models were provided, detailing the conceptualization and operationalization of the main elements of the proposed automated feedback generation framework for learner models. This chapter ended with a discussion on learner model selection and automated feedback generation model selection for various educational tasks and assessments.

Chapter 6 Educational Implications

By implementing the automated feedback generation system for different learner models, this research makes the following contributions.

Theoretical and Methodological Implications

The present research proposes and implements a data-driven automated feedback generation system for learner models in structured knowledge domains. It reviews the mainstream learner models within the EDM community and explains their statistical assumptions, input representations, and output representations for various educational purposes. Specifically, the meanings of the output representations and take-home messages for learners are interpreted for more efficient use of learning technologies. A deep knowledge tracing model with contextualized information is also proposed and implemented to use both product and process data to make more accurate estimations of learners' abilities and better understandings of learners' knowledge acquisition processes. In addition, the proposed data-driven feedback generation framework specifies how to augment expert-derived feedback templates and create a corpus of academic feedback using unsupervised sentence generation method. The framework also details the process of preparing and providing adaptive feedback for learners based on different learner models using a combination of data-driven and expert-driven approaches.

Practical Implications

Providing feedback is essential for improving performance. However, understanding the context of learning is fundamental prior to learner modeling and feedback generation. Thus, it is important to conduct in-depth investigations regarding which learner models should be applied to a certain context (i.e., system, student, or domain) and what forms of feedback (i.e., summative vs. formative, static vs. adaptive, single KC vs. all KCs) students need for knowledge acquisition that best suit the learner model and output representation.

Practically and pedagogically, this research contributes to the educational data mining and learning analytics research on learner modeling and adaptive feedback generation based on different contexts. The proposed framework aims to facilitate online tutoring and assessment systems in monitoring their students' learning activities and conducting real-time feedback generation to motivate students and to prevent failure.

Future Research

Future research can be conducted in the following aspects. First, the proposed framework will be extended from structured domains to unstructured domains such as essay writing. Second, future studies will prompt adaptive feedback with dynamic thresholds of KC mastery that fluctuate with learners' knowledge gains to counterbalance the need for expert-derived thresholds. Third, the feedback models will incorporate more learner characteristics including demographic information, mindsets, and affective features to better calibrate student performance for personalized learning. Another potential direction is to induce feedback provision policies using deep reinforcement learning to determine the optimal strategies of feedback provision with data-driven approaches.

Chapter Summary

Chapter 6 summarized the theoretical, methodological, and practical implications of this research and outlines the main future research directions. Theoretically, the proposed approach leverages the understanding of students' learning behaviour and the dynamic knowledge acquisition process on different knowledge components in digital learning systems. Methodologically, it implements a novel approach that incorporates both data-driven methods and expert-derived strategies to inform decision making on feedback generation and provision. Pedagogically, the real-time monitoring and feedback prompting system enables the existing digital learning systems to detect students' dynamic skill mastery progressions and to provide feedback that adapts to learners' future practice.

Chapter 7 Conclusion and Limitation

The study implements five learner models (Rasch model, BKT, DKT-RNN, DKT-LSTM, and DKT-CI) on three datasets of various sizes, and further develops an automated feedback generation framework for learner models. The characteristics, pros and cons, interpretations, and applicability of the feedback generation method for each model are compared and discussed. Results show that IRT and BKT outperform DKT on smaller datasets, whereas DKT-CI outperforms all other models on larger datasets. For BKT, the proposed template-based feedback generation can produce KC-dependent feedback corresponding to each learner's performance as well as expert-derived thresholds, in which KCs are estimated independently. For DKT, the feedback generation methods can produce adaptive feedback for all KCs at every time step and plot the knowledge transfer for individuals. Thus, DKT is more suitable for individualized formative tutoring. Learner models and feedback provision are both key elements of ITSs. The synthesis of learner models in the present study can facilitate researchers and educators to better incorporate instructional design and adapt pedagogical policy in the ITSs. Future research can be conducted on prompting adaptive feedback with dynamic thresholds of KC mastery that fluctuate with learners' knowledge gains to counterbalance the need for expert-derived thresholds. Another potential direction is to induce feedback provision policies using deep reinforcement learning to determine the optimal strategies of feedback provision with data-driven approaches.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... & Zheng, X. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Abyaa, A., Khalidi Idrissi, M., & Bennani, S. (2019). Learner modelling: Systematic review of the literature from the last 5 years. *Educational Technology Research and Development, 67*(5), 1105-1143. https://doi.org/10.1007/s11423-018-09644-1

Agudo-Peregrina, Á. F., Iglesias-Pradas, S., Conde-González, M. Á., & Hernández-García, Á. (2014). Can we predict success from log data in VLEs? Classification of interactions for learning analytics and their relation with performance in VLE-supported F2F and online learning. *Computers in Human Behavior*, *31*, 542-550.

Aleven, V., Mclaren, B., Roll, I., & Koedinger, K. (2006). Toward meta-cognitive tutoring: A model of help seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence in Education*, *16*(2), 101-128.

Allen, I. E., & Seaman, J. (2008). *Staying the course. Online education in the United States, 2008*. Newburyport, MA: The Sloan Consortium.

Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, *46*(3), 175-185.

Anderson, J. R., Conrad, F. G., & Corbett, A. T. (1989). Skill acquisition and the LISP tutor. *Cognitive Science*, *13*(4), 467-505.

Anderson, L. W., & Bloom, B. S. (2001). *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives*. New York: Longman.

Ashenafi, M. M., Riccardi, G., & Ronchetti, M. (2015, October). Predicting students' final exam scores from their course activities. In *IEEE Frontiers in Education Conference* (pp. 1-9). IEEE.

Baker, R. S. J. d., Corbett, A. T., & Aleven, V. (2008). More accurate student modeling through contextual estimation of slip and guess probabilities in Bayesian Knowledge Tracing. Paper presented at the *Intelligent Tutoring,* 406-415.

Baker, R. S. J. d., & Inventado, P. S. (2014). Educational data mining and learning analytics. In J. A. Larusson, & B.White (Eds.), *Learning analytics: From research to practice*. New York: Springer.

Baker, R. S. J. d., Lindrum, D., Lindrum, M. J., & Perkowski, D. (2015). Analyzing early at-risk factors in higher education e-Learning courses. In *Proceedings of the 8th International Conference on Educational Data Mining*, Madrid, Spain, June 26-29, 2015.

Baker, R. S. J. d., & Yacef, K. (2009). The state of educational data mining in 2009: A review and future visions. *Journal of Educational Data Mining, 1*(1), 3-17.

Bao, W. (2020). COVID-19 and online teaching in higher education: A case study of Peking University. *Human Behavior and Emerging Technologies*, *2*(2), 113-115.

Barnes, T., & Stamper, J. (2010). Automatic hint generation for logic proof tutoring using historical data. *Journal of Educational Technology & Society, 13*(1), 3-12.

Barton, M. A., & Lord, F. M. (1981). An upper asymptote for the three-parameter logistic Item-Response Model. *ETS Research Report Series, 1981*(1), i-8. https://doi.org/10.1002/j.2333-8504.1981.tb01255.x

Bates, D., Mächler, M., Bolker, B., & Walker, S. (2014). Fitting linear mixed-effects models using *lme4*. *arXiv preprint arXiv:1406.5823*.

Beck, J. M., Ma, W. J., Kiani, R., Hanks, T., Churchland, A. K., Roitman, J., Shadlen, M. N., Latham, P. E., & Pouget, A. (2008). Probabilistic population codes for Bayesian decision making. *Neuron (Cambridge, Mass.), 60*(6), 1142-1152. https://doi.org/10.1016/j.neuron.2008.09.021

Becker, K., Vanzin, M., Marquardt, C., & Ruiz, D. (2006). Applying web usage mining for the analysis of behavior in web-based learning environments. In C. Romero & S. Ventura, (Eds.), *Data mining in e-learning* (pp. 117-137). Billerica, MA: WitPress.

Beutel, A., Covington, P., Jain, S., Xu, C., Li, J., Gatto, V., & Chi, E. H. (2018, February). Latent cross: Making use of context in recurrent recommender systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining* (pp. 46-54).

Biggs, J. (1993). What do inventories of students' learning processes really measure? A theoretical review and clarification. *British Journal of Educational Psychology*, *63*(1), 3-19.

Birnbaum, A. L. (1968). Some latent trait models and their use in inferring an examinee's ability. *Statistical Theories of Mental Test Scores,* https://ci.nii.ac.jp/naid/10011544105/en/

Black, P., & Wiliam, D. (1998). Assessment and classroom learning. *Assessment in Education: Principles, Policy & Practice*, *5*(1), 7-74.

Bloom, B. S. (1956). Taxonomy of educational objectives. Vol. 1: Cognitive domain. *New York: McKay*, *20*, 24.

Bloom, B. S., Krathwohl, D. R., & Masia, B. B. (1984). Bloom taxonomy of educational objectives. In *Allyn and Bacon*. Pearson Education.

Boud, D., & Molloy, E. (2013). *Feedback in higher and professional education: Understanding it and doing it well*. Routledge.

Boud, D., & Molloy, E. (2013). Rethinking models of feedback for learning: The challenge of design. *Assessment & Evaluation in Higher Education*, *38*(6), 698-712.

Bousbia, N., & Belamri, I. (2014). Which contribution does EDM provide to computer-based learning environments? In A. Peña-Ayala (Ed.), *Educational Data Mining: Applications and Trends*. New York: Springer.

Bradford, P., Porciello, M., Balkon, N., & Backus, D. (2007). The Blackboard learning system: The be all and end all in educational instruction?. *Journal of Educational Technology Systems*, *35*(3), 301-314.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5-32.

Brophy, J. (1981). Teacher praise: A functional analysis. *Review of Educational Research*, *51*(1), 5-32.

Bull, S. (2004). Supporting learning with open learner models. *Planning, 29*(14), 1.

Butler, D. L., & Winne, P. H. (1995). Feedback and self-regulated learning: A theoretical synthesis. *Review of Educational Research*, *65*(3), 245-281.

Cantabella, M., Martínez-España, R., Ayuso, B., Yáñez, J. A., & Muñoz, A. (2019). Analysis of student behavior in learning management systems through a Big Data framework. *Future Generation Computer Systems*, *90*, 262-272.

Carless, D. (2006). Differing perceptions in the feedback process. *Studies in Higher Education*, *31*(2), 219-233.

Carless, D., Salter, D., Yang, M., & Lam, J. (2011). Developing sustainable feedback practices. *Studies in Higher Education*, *36*(4), 395-407.

Carver, C. S., & Scheier, M. F. (1981). The self-attention-induced feedback loop and social facilitation. *Journal of Experimental Social Psychology*, *17*(6), 545-568.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Chung, J. Y., & Lee, S. (2019). Dropout early warning systems for high school students using machine learning. *Children and Youth Services Review*, *96*, 346-353.

Conijn, R., Snijders, C., Kleingeld, A., & Matzat, U. (2016). Predicting student performance from LMS data: A comparison of 17 blended courses using Moodle LMS. *IEEE Transactions on Learning Technologies*, *10*(1), 17-29.

Corbett, A. T., & Anderson, J. R. (1994). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction, 4*(4), 253-278. https://doi.org/10.1007/BF01099821

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273-297.

Deeva, G., Bogdanova, D., Serral, E., Snoeck, M., & De Weerdt, J. (2021). A review of automated feedback systems for learners: Classification framework, challenges and opportunities. *Computers & Education, 162*, 104094. https://doi.org/10.1016/j.compedu.2020.104094

Donkers, T., Loepp, B., & Ziegler, J. (2017, August). Sequential user-based recurrent neural network recommendations. In *Proceedings of the Eleventh ACM Conference on Recommender Systems* (pp. 152-160).

Duchaine, E. L., Jolivette, K., & Fredrick, L. D. (2011). The effect of teacher coaching with performance feedback on behavior-specific praise in inclusion classrooms. *Education and Treatment of Children*, 209-227.

Dutt, A., Ismail, M. A., & Herawan, T. (2017). A systematic review on educational data mining. *IEEE Access, 5*, 15991-16005.

Eddy, S. R. (1996). Hidden Markov models. *Current Opinion in Structural Biology, 6*(3), 361-365. https://doi.org/10.1016/S0959-440X(96)80056-X

Embretson, S. E., & Reise, S. P. (2013). *Item response theory*. Psychology Press.

Ferguson, P. (2011). Student perceptions of quality feedback in teacher education. *Assessment & Evaluation in Higher Education, 36*(1), 51-62. https://doi.org/10.1080/02602930903197883

Ferguson, R. (2012). Learning analytics: Drivers, developments and challenges. *International Journal of Technology Enhanced Learning*, *4*(5-6), 304-317.

Fontana, D. (1981). Learning and teaching. In *Psychology for physiotherapists* (pp. 66-100). Palgrave, London.

Gardner, J., & Brooks, C. (2018, April). Dropout model evaluation in MOOCs. The 8th AAAI Symposium on Educational Advances in Artificial Intelligence. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence* (pp. 7906-7912).

Gašević, D., Dawson, S., Rogers, T., & Gasevic, D. (2016). Learning analytics should not promote one size fits all: The effects of instructional conditions in predicting academic success. *The Internet and Higher Education*, *28*, 68-84.

Gast, D. L., Doyle, P. M., Wolery, M., Ault, M. J., & Kolenda, J. L. (1994). Instructive feedback: Effects of number and type. *Journal of Behavioral Education*, *4*(3), 313-334.

Geigle, C., & Zhai, C. (2017, April). Modeling MOOC student behavior with two-layer hidden Markov models. In *Proceedings of the Fourth ACM conference on Learning@ Scale* (pp. 205-208).

Gervet, T., Koedinger, K., Schneider, J., & Mitchell, T. (2020). When is deep learning the best approach to knowledge tracing?. *JEDM| Journal of Educational Data Mining*, *12*(3), 31-54.

Geyer, C. J. (1992). Practical Markov Chain Monte Carlo. *Statistical Science, 7*(4), 473-483. https://www.jstor.org/stable/2246094

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1, No. 2). Cambridge: MIT press.

Hannafin, M. J., & Land, S. M. (1997). The foundations and assumptions of technology-enhanced student-centered learning environments. *Instructional Science*, *25*(3), 167-202.

Hassan, S. U., Waheed, H., Aljohani, N. R., Ali, M., Ventura, S., & Herrera, F. (2019). Virtual learning environment to predict withdrawal by leveraging deep learning. *International Journal of Intelligent Systems*, *34*(8), 1935-1952.

Hattie, J., & Timperley, H. (2007). The power of feedback. *Review of Educational Research, 77*(1), 81-112. https://doi.org/10.3102/003465430298487

He, J., Bailey, J., Rubinstein, B. I., & Zhang, R. (2015, February). Identifying at-risk students in massive open online courses. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence* (Vol. 29, No. 1).

Heffernan, N. T., & Heffernan, C. L. (2014). 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. https://doi.org/10.1007/s40593-014-0024-x

Heffernan, N. T., & Koedinger, K. R. (2002). An intelligent tutoring system incorporating a model of an experienced human tutor. Paper presented at the *International Conference on Intelligent Tutoring Systems,* 596-608.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735-1780.

Hung, J. L., & Crooks, S. M. (2009). Examining online learning patterns with data mining techniques in peer-moderated and teacher-moderated courses. *Journal of Educational Computing Research, 40*(2), 183-210.

Ifenthaler, D., & Widanapathirana, C. (2014). Development and validation of a learning analytics framework: Two case studies using support vector machines. *Technology, Knowledge and Learning*, *19*(1-2), 221-240.

Jayaprakash, S. M., Moody, E. W., Lauría, E. J., Regan, J. R., & Baron, J. D. (2014). Early alert of academically at-risk students: An open source analytics initiative. *Journal of Learning Analytics*, *1*(1), 6-47.

Jovanovic, J., Gasevic, D., Dawson, S., Pardo, A., & Mirriahi, N. (2017). Learning analytics to unveil learning strategies in a flipped classroom. *Internet and Higher Education, 33*, 74-85.

Juhaňák, L., Zounek, J., & Rohlíková, L. (2019). Using process mining to analyze students' quiz-taking behavior patterns in a learning management system. *Computers in Human Behavior*, *92*, 496-506.

Kamiński, B., Jakubczyk, M., & Szufel, P. (2018). A framework for sensitivity analysis of decision trees. *Central European Journal of Operations Research*, *26*(1), 135-159.

Katan, S., & Anstead, E. (2020, April). Work In Progress: Sleuth, a programming environment for testing gamification. In *2020 IEEE Global Engineering Education Conference (EDUCON)* (pp. 1503-1507). IEEE.

Kendall, M. G. (1938). A new measure of rank correlation. *Biometrika*, *30*(1/2), 81-93.

Keogh, E., & Kasetty, S. (2003). On the need for time series data mining benchmarks: A survey and empirical demonstration. *Data Mining and Knowledge Discovery*, *7*(4), 349-371.

Keuning, H., Heeren, B., & Jeuring, J. (2014, November). Strategy-based feedback in a programming tutor. In *Proceedings of the Computer Science Education Research Conference* (pp. 43-54).

Khajah, M., Lindsey, R. V., & Mozer, M. C. (2016). How deep is knowledge tracing?. *arXiv preprint arXiv:1604.02416*.

Khalil, M., & Ebner, M. (2017). Clustering patterns of engagement in massive open online courses (MOOCs): The use of learning analytics to reveal student categories. *Journal of Computing in Higher Education*, *29*(1), 114-132.

Kim, D., Kim, S., Zhao, H., Li, S., Rossi, R. A., & Koh, E. (2019, January). Domain switch-aware holistic recurrent neural network for modeling multi-domain user behavior. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining* (pp. 663-671).

Kim, D., Yoon, M., Jo, I. H., & Branch, R. M. (2018). Learning analytics to support self-regulated learning in asynchronous online courses: A case study at a women's university in South Korea. *Computers & Education*, *127*, 233-251.

Kleinbaum, D. G., Dietz, K., Gail, M., Klein, M., & Klein, M. (2002). *Logistic regression*. New York: Springer-Verlag.

Kloft, M., Stiehler, F., Zheng, Z., & Pinkwart, N. (2014, October). Predicting MOOC dropout over weeks using machine learning methods. In *Proceedings of the EMNLP workshop on Analysis of Large Scale Social Interaction in MOOCs* (pp. 60-65).

Koedinger, K. R., Baker, R. S., Cunningham, K., Skogsholm, A., Leber, B., & Stamper, J. (2010). A data repository for the EDM community: The PSLC DataShop. *Handbook of educational data mining, 43*, 43-56.

Kluger, A. N., & DeNisi, A. (1996). The effects of feedback interventions on performance: A historical review, a meta-analysis, and a preliminary feedback intervention theory. *Psychological Bulletin*, *119*(2), 254.

Kluger, A. N., & DeNisi, A. (1998). Feedback interventions: Toward the understanding of a double-edged sword. *Current Directions in Psychological Science*, *7*(3), 67-72.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436-444.

Li, S., & Zhao, H. (2020). A survey on representation learning for user modeling. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence* (pp. 4997-5003).

Liu, H., Wang, Z., Benachour, P., & Tubman, P. (2018). A time series classification method for behaviour-based dropout prediction. In *Proceedings of the IEEE 18th International Conference on Advanced Learning Technologies* (pp. 191-195), Mumbai.

Liu, Z., Xiong, F., Zou, K., & Wang, H. (2018). Predicting learning status in MOOCs using LSTM. *arXiv* preprint arXiv:1808.01616.

Lord, F. M. (1980). *Applications of item response theory to practical testing problems*. Routledge.

Lu, C., Ganguly, P., Sabharwal, P., Karimi Abdolmaleki, M., Southcott, J., Jin, H.-Y., González Esparza, L., & Cutumisu, M. (2021). A review of automated-feedback generation in Intelligent Tutoring Systems. Paper presented at the *49th of the Canadian Society for the Study of Education (CSSE) Conference*. Edmonton, AB, May 29 - June 3.

Lu, O. H., Huang, J. C., Huang, A. Y., & Yang, S. J. (2017). Applying learning analytics for improving students' engagement and learning outcomes in an MOOCs enabled collaborative programming course. *Interactive Learning Environments*, *25*(2), 220-234.

Maas, A., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011). Learning word vectors for sentiment analysis. Paper presented at the *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies,* 142-150.

Macfadyen, L. P., & Dawson, S. (2012). Numbers are not enough. Why e-learning analytics failed to inform an institutional strategic plan. *Journal of Educational Technology & Society*, *15*(3), 149-163.

Macfadyen, L. P., Dawson, S., Pardo, A., & Gaševic, D. (2014). Embracing big data in complex educational systems: The learning analytics imperative and the policy challenge. *Research & Practice in Assessment*, *9*, 17-28.

Maniktala, M., Barnes, T., & Chi, M. (2020). Extending the hint factory: Towards modelling productivity for open-ended problem-solving. Paper presented at the *In Proceedings of the 13th International Conference on Educational Data Mining (DC Paper).*

Marwan, S., Lytle, N., Williams, J. J., & Price, T. (2019, July). The impact of adding textual explanations to next-step hints in a novice programming environment. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education* (pp. 520-526).

Marwan, S., Jay Williams, J., & Price, T. (2019, July). An evaluation of the impact of automated programming hints on performance and learning. In *Proceedings of the 2019 ACM Conference on International Computing Education Research* (pp. 61-70).

Mao, Y., Lin, C., & Chi, M. (2018). Deep Learning vs. Bayesian Knowledge Tracing: Student models for interventions. *Journal of Educational Data Mining*, 10(2).

Mao, Y., Zhi, R., Khoshnevisan, F., Price, T. W., Barnes, T., & Chi, M. (2019, January). One minute is enough: Early prediction of student success and event-level difficulty during novice programming tasks. In C. F. Lynch, A. Merceron, M. Desmarais, & R. Nkambou (eds.). In *Proceedings of the 12th International Conference on Educational Data Mining* (pp. 119 - 128).

Masters, G. N. (1982). A Rasch model for partial credit scoring. *Psychometrika, 47*(2), 149-174.

McHugh, M. L. (2012). Interrater reliability: The kappa statistic. *Biochemia Medica*, *22*(3), 276-282.

McFarland, D., & Hamilton, D. (2005). Factors affecting student performance and satisfaction: Online versus traditional course delivery. *Journal of Computer Information Systems*, *46*(2), 25-32.

Merrill, D. C., Reiser, B. J., Ranney, M., & Trafton, J. G. (1992). Effective tutoring techniques: A comparison of human tutors and intelligent tutoring systems. *Journal of the Learning Sciences, 2*(3), 277-305. https://doi.org/10.1207/s15327809jls0203_2

Miao, N., Zhou, H., Mou, L., Yan, R., & Li, L. (2019). CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence, 33*(01), 6834-6842. https://doi.org/10.1609/aaai.v33i01.33016834

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Karafiát, M., Burget, L., Černocký, J., & Khudanpur, S. (2010). Recurrent neural network based language model. In *INTERSPEECH-2010, 1045-1048.*

Molloy, E. K., & Boud, D. (2014). Feedback models for learning, teaching and performance. In *Handbook of research on educational communications and technology* (pp. 413-424). Springer, New York, NY.

Moreno, R. (2004). Decreasing cognitive load for novice students: Effects of explanatory versus corrective feedback in discovery-based multimedia. *Instructional Science*, *32*(1), 99-113.

Nakagawa, H., Iwasawa, Y., & Matsuo, Y. (2019, October). Graph-based knowledge tracing: Modeling student proficiency using graph neural network. In *2019 IEEE/WIC/ACM International Conference on Web Intelligence (WI)* (pp. 156-163). IEEE.

Narciss, S., & Huth, K. (2004). How to design informative tutoring feedback for multimedia learning. *Instructional Design for Multimedia Learning*, *181195*.

Newell, A., & Simon, H. A. (1972). *Human problem solving* (Vol. 104, No. 9). Englewood Cliffs, NJ: Prentice-hall.

Olivé, D. M., Huynh, D. Q., Reynolds, M., Dougiamas, M., & Wiese, D. (2020). A supervised learning framework: Using assessment to identify students at risk of dropping out of a MOOC. *Journal of Computing in Higher Education*, *32*(1), 9-26.

O'Neill, K., Singh, G., & O'Donoghue, J. (2004). Implementing eLearning programmes for higher education: A review of the literature. *Journal of Information Technology Education: Research*, *3*(1), 313-323.

Papamitsiou, Z., & Economides, A. A. (2014). Learning analytics and educational data mining in practice: A systematic literature review of empirical evidence. *Educational Technology & Society, 17*(4), 49-64.

Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002, July). Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics* (pp. 311-318).

Pardo, A., Jovanovic, J., Dawson, S., Gašević, D., & Mirriahi, N. (2019). Using learning analytics to scale the provision of personalised feedback. *British Journal of Educational Technology*, *50*(1), 128-138.

Patel, C., & Patel, T. (2006). Exploring a joint model of conventional and online learning systems. *E-Service*, *4*(2), 27-46.

Pavlik Jr, P. I., Cen, H., & Koedinger, K. R. (2009). Performance Factors Analysis--A New Alternative to Knowledge Tracing. *Online Submission*. Retrieved from https://eric.ed.gov/?id=ED506305.

Peña-Ayala, A. (2014). Educational data mining: A survey and a data mining-based analysis of recent works. *Expert Systems with Applications, 41*(4), 1432-1462.

Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1532-1543).

Piech, C., Spencer, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L., & Sohl-Dickstein, J. (2015). Deep knowledge tracing. *arXiv preprint arXiv:1506.05908*.

Poornima, S., & Pushpalatha, M. (2019). Drought prediction based on SPI and SPEI with varying timescales using LSTM recurrent neural network. *Soft Computing*, *23*(18), 8399-8412.

Price, M., Handley, K., Millar, J., & O'Donovan, B. (2010). Feedback: All that effort, but what is the effect?. *Assessment & Evaluation in Higher Education*, *35*(3), 277-289.

Price, T. W., Zhi, R., & Barnes, T. (2017, June). Hint generation under uncertainty: The effect of hint quality on help-seeking behavior. In *International Conference on Artificial Intelligence in Education* (pp. 311-322). Springer, Cham.

Rasch, G. (1993). *Probabilistic models for some intelligence and attainment tests*. MESA Press, 5835 S.

Roll, I., Aleven, V., McLaren, B. M., & Koedinger, K. R. (2011). Improving students' help-seeking skills using metacognitive feedback in an intelligent tutoring system. *Learning and Instruction*, *21*(2), 267-280.

Sadler, D. R. (1989). Formative assessment and the design of instructional systems. *Instructional Science*, *18*(2), 119-144.

Schlkopf, B., Smola, A. J., & Bach, F. (2018). *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT Press.

Seber, G. A., & Lee, A. J. (2012). *Linear regression analysis* (Vol. 329). John Wiley & Sons.

Sedrakyan, G., Malmberg, J., Verbert, K., Järvelä, S., & Kirschner, P. A. (2020). Linking learning behavior analytics and learning science concepts: Designing a learning analytics dashboard for feedback to support learning regulation. *Computers in Human Behavior*, *107*, 105512.

Shatnawi, S., Gaber, M. M., & Cocea, M. (2014, September). Automatic content related feedback for MOOCs based on course domain ontology. In *International Conference on Intelligent Data Engineering and Automated Learning* (pp. 27-35). Springer, Cham.

Shi, C., Fu, S., Chen, Q., & Qu, H. (2015, April). VisMOOC: Visualizing video clickstream data from massive open online courses. In *Visualization Symposium (PacificVis), IEEE Pacific* (pp. 159-166). IEEE.

Shuell, T. J. (1986). Cognitive conceptions of learning. *Review of Educational Research*, *56*(4), 411-436.

Shute, V. J. (2008). Focus on formative feedback. *Review of Educational Research*, *78*(1), 153-189.

Silva, P., Costa, E., & de Araújo, J. R. (2019, June). An adaptive approach to provide feedback for students in programming problem solving. In *International Conference on Intelligent Tutoring Systems* (pp. 14-23). Springer, Cham.

Singh, R., Gulwani, S., & Solar-Lezama, A. (2013, June). Automated feedback generation for introductory programming assignments. In *Proceedings of the 34th ACM SIGPLAN conference on Programming language design and implementation* (pp. 15-26).

Smith, V. C., Lange, A., & Huston, D. R. (2012). Predictive modeling to forecast student outcomes and drive effective interventions in online community college courses. *Journal of Asynchronous Learning Networks*, *16*(3), 51-61.

Sun, L., Tang, Y., & Zuo, W. (2020). Coronavirus pushes education online. *Nature Materials*, *19*(6), 687-687.

Thurlings, M., Vermeulen, M., Bastiaens, T., & Stijnen, S. (2013). Understanding feedback: A learning theory perspective. *Educational Research Review*, *9*, 1-15.

Topîrceanu, A., & Grosseck, G. (2017). Decision tree learning used for the classification of student archetypes in online courses. *Procedia Computer Science*, *112*, 51-60.

UNESCO. (2020, June 21). COVID-19 educational disruption and response. https://en.unesco.org/themes/education-emergencies/coronavirus-school-closures

Van Der Aalst, W. (2012). Process mining. *Communications of the ACM*, *55*(8), 76-83.

van der Linden, Wim J, & Glas, C. A. (2010). *Elements of adaptive testing*. Springer.

Waheed, H., Hassan, S. U., Aljohani, N. R., Hardman, J., Alelyani, S., & Nawaz, R. (2020). Predicting academic performance of students from VLE big data using deep learning models. *Computers in Human Behavior*, *104*, 106189.

Wang, L., Sy, A., Liu, L., & Piech, C. (2017, April). Deep knowledge tracing on programming exercises. In *Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale* (pp. 201-204).

Washington, G. Y. (2019). The learning management system matters in face-to-face higher education courses. *Journal of Educational Technology Systems*, *48*(2), 255-275.

Weimer, M. (2002). *Learner-centered teaching: Five key changes to practice*. John Wiley & Sons.

Werts, M. G., Wolery, M., Holcombe, A., & Gast, D. L. (1995). Instructive feedback: Review of parameters and effects. *Journal of Behavioral Education*, *5*(1), 55-75.

Werts, M. G., Wolery, M., Holcombe, A., & Frederick, C. (1993). Effects of instructive feedback related and unrelated to the target behaviors. *Exceptionality*, *4*(2), 81-95.

Wickham, H., Francois, R., Henry, L., & Müller, K. (2014, June). Dplyr. In *useR! Conference*.

Winnie, P., & Butler, D. (1994). Student cognitive processing and learning. For the 2nd Edition of The International Encyclopaedia of Education. HUSEN, T. & POSTLETHWAITE, TN, Eds.

Wright, R. E. (1995). *Logistic regression.* In L. G. Grimm & P. R. Yarnold (Eds.), *Reading and understanding multivariate statistics* (p. 217–244). American Psychological Association.

Xing, W., Chen, X., Stein, J., & Marcinkowski, M. (2016). Temporal prediction of dropouts in MOOCs: Reaching the low hanging fruit through stacking generalization. *Computers in Human Behavior*, *58*, 119-129.

Xing, W., & Du, D. (2019). Dropout prediction in MOOCs: Using deep learning for personalized intervention. *Journal of Educational Computing Research*, *57*(3), 547-570.

Xu, X., Wang, J., Peng, H., & Wu, R. (2019). Prediction of academic performance associated with internet usage behaviors using machine learning algorithms. *Computers in Human Behavior*, *98*, 166-173.

Yang, S., Zhu, M., Hou, J., & Lu, X. (2020). Deep Knowledge Tracing with Convolutions. *arXiv preprint arXiv:2008.01169*.

Yeung, C. K., & Yeung, D. Y. (2018, June). Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale* (pp. 1-10).

You, J. W. (2016). Identifying significant indicators using LMS data to predict course achievement in online learning. *The Internet and Higher Education*, *29*, 23-30.

Yudelson, M. V., Koedinger, K. R., & Gordon, G. J. (2013, July). Individualized Bayesian Knowledge Tracing models. In *International Conference on Artificial Intelligence in Education* (pp. 171-180). Springer, Berlin, Heidelberg.

Zacharis, N. Z. (2015). A multivariate approach to predicting student outcomes in web-enabled blended learning courses. *The Internet and Higher Education, 27*, 44-53.

Zhang, Y., & Jiang, W. (2018). Score prediction model of MOOCs learners based on neural network. *International Journal of Emerging Technologies in Learning (iJET)*, *13*(10), 171-182.

Zhang, L., Xiong, X., Zhao, S., Botelho, A., & Heffernan, N. T. (2017, April). Incorporating rich features into deep knowledge tracing. In *Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale* (pp. 169-172).

Zhou, L., Wu, S., Zhou, M., & Li, F. (2020). 'School's Out, But Class' On', The largest online education in the world today: Taking China's practical exploration during the COVID-19 epidemic prevention and control as an example. *Best Evidence of Chinese Education, 4*(2):501-519. http://dx.doi.org/10.2139/ssrn.3555520

Zhu, M., Liu, O. L., & Lee, H. S. (2020). The effect of automated feedback on revision behavior and learning gains in formative assessment of scientific argument writing. *Computers & Education*, *143*, 103668.

Appendices

Appendix 1

Hyperparameter Settings of the Best DKT Models

	Hyperparameters		
	ASSISTment 2009-2010	**Algebra I 2005-2006**	**OLI Fall 2010**
DKT-RNN	Optimizer: *Adam*; learning rate: 0.001; batch size: 32, dropout probability: 0.6; hidden layer dimension: 100; number of recurrent layers: 1; ℓ1 regularizers: 0.01; ℓ2 regularizers: 0.01	Optimizer: *Adam*; learning rate: 0.001; batch size: 32, dropout probability: 0.6; hidden layer dimension: 100; number of recurrent layers: 1; ℓ1 regularizers: 0.01; ℓ2 regularizers: 0.01	Optimizer: *Adam*; learning rate: 0.001; batch size: 5, dropout probability: 0.6; hidden layer dimension: 100; number of recurrent layers: 1; ℓ1 regularizers: 0.01; ℓ2 regularizers: 0.01
DKT-LSTM	Optimizer: *Adam*; learning rate: 0.001; batch size: 32, dropout probability: 0.6; hidden layer dimension: 100; number of recurrent layers: 1; ℓ1	Optimizer: *Adam*; learning rate: 0.001; batch size: 32, dropout probability: 0.6; hidden layer dimension: 100; number of recurrent layers: 1; ℓ1	Optimizer: *Adam*; learning rate: 0.001; batch size: 5, dropout probability: 0.6; hidden layer dimension: 100; number of recurrent layers: 1; ℓ1

	regularizers: 0.01; ℓ2 regularizers: 0.01	regularizers: 0.01; ℓ2 regularizers: 0.01	regularizers: 0.01; ℓ2 regularizers: 0.01
DKT-CI	Optimizer: *Adam*; learning rate: 0.001; batch size: 32, dropout probability: 0.6; hidden layer dimension: 100; number of recurrent layers: 1; ℓ1 regularizers: 0.01; ℓ2 regularizers: 0.01; embedding dimension: 1000	Optimizer: *Adam*; learning rate: 0.001; batch size: 5, dropout probability: 0.6; hidden layer dimension: 100; number of recurrent layers: 2; ℓ1 regularizers: 0.01; ℓ2 regularizers: 0.01; embedding dimension: 500	Optimizer: *Adam*; learning rate: 0.001; batch size: 5, dropout probability: 0.6; hidden layer dimension: 100; number of recurrent layers: 1; ℓ1 regularizers: 0.01; ℓ2 regularizers: 0.01; embedding dimension: 250

Appendix 2

The Adaptive DKT Output for 50 actions and 5 KCs for the ASSISTment 09-10 Dataset at one time step

Attempts (skill ID, correct or not)	32: Ordering Positive Decimals	33: Ordering Fractions	45: Subtraction Whole Numbers	55: Absolute Value	98: Equation Solving Two or Fewer Steps
(55,1)	0.4173	0.5133	0.3680	0.4703	0.5641
(45,1)	0.3574	0.8526	0.5014	0.3749	0.4338
(55,1)	0.3714	0.7322	0.4427	0.4686	0.4776
(55,1)	0.5310	0.6108	0.5783	0.5147	0.5209
(55,0)	0.2793	0.6292	0.4974	0.5713	0.4899
(45,0)	0.3221	0.6397	0.5526	0.3927	0.3128
(98,1)	0.3388	0.5687	0.4477	0.4094	0.5792
(98,1)	0.3669	0.6980	0.4929	0.4686	0.5741
(98,1)	0.4644	0.7553	0.5098	0.5074	0.6554
(33,0)	0.2838	0.1308	0.3025	0.4147	0.4536
(32,0)	0.2495	0.6049	0.3801	0.5093	0.4857
(33,0)	0.0571	0.0492	0.1131	0.3916	0.3764
(32,0)	0.2502	0.5591	0.3398	0.4836	0.4463
(33,0)	0.0307	0.0172	0.0695	0.2991	0.3083
(32,0)	0.2198	0.5945	0.2595	0.4663	0.4236
(33,0)	0.0164	0.0140	0.0363	0.2335	0.2552
(32,0)	0.1500	0.5769	0.1966	0.4387	0.4030
(32,0)	0.0023	0.3902	0.0172	0.1657	0.4962
(33,0)	0.2155	0.6740	0.4003	0.3985	0.4233
(32,1)	0.4916	0.8385	0.2973	0.4391	0.5519
(33,1)	0.1649	0.7374	0.2551	0.4138	0.5028
(32,0)	0.0136	0.6091	0.0669	0.3151	0.5622
(33,0)	0.1142	0.5292	0.2173	0.3897	0.4366
(32,0)	0.0128	0.2942	0.0432	0.2190	0.5666
(33,0)	0.1593	0.3907	0.2288	0.3712	0.4340
(33,0)	0.0633	0.0257	0.0581	0.2571	0.3425
(32,0)	0.0767	0.4318	0.1296	0.3800	0.4071
(33,0)	0.0185	0.0287	0.0404	0.2332	0.2650
(32,0)	0.0877	0.4507	0.1165	0.3719	0.3897
(32,0)	0.0029	0.2943	0.0151	0.1605	0.4757
(33,0)	0.2256	0.6106	0.3444	0.3675	0.4243
(32,1)	0.4733	0.7905	0.2316	0.3370	0.5346
(33,1)	0.1239	0.7155	0.1837	0.3527	0.4981
(33,1)	0.1727	0.9446	0.1914	0.4035	0.6139

(32,1)	0.2130	0.8012	0.3285	0.3880	0.4067
(32,0)	0.0015	0.6081	0.0177	0.2324	0.5458
(33,0)	0.1486	0.7020	0.2803	0.4095	0.4678
(33,0)	0.0366	0.0133	0.0426	0.2464	0.3329
(32,0)	0.1059	0.5061	0.1694	0.3897	0.4609
(33,0)	0.0234	0.0385	0.0550	0.2761	0.2727
(32,0)	0.0363	0.4957	0.0668	0.3709	0.3884
(33,1)	0.0534	0.9797	0.1621	0.3205	0.5700
(32,1)	0.3483	0.7996	0.3666	0.3989	0.3985
(33,0)	0.0236	0.0079	0.0313	0.2276	0.3718
(32,0)	0.3510	0.5579	0.2928	0.4043	0.4526
(33,1)	0.0789	0.9944	0.2976	0.3878	0.5800
(33,1)	0.1418	0.6365	0.2299	0.3633	0.4046

Appendix 3

The Complete KC Knowledge Transfer Heatmap (110 KCs) for the ASSISTment 09-10 Dataset