LUPINE PUBLISHERS

Current Trends in Computer Sciences & Applications

DOI: 10.32474/CTCSA.2022.02.000137

ISSN: 2643-6744

Case Study

Human-in-the-loop Computational Cognitive Modeling for Human Learning

Jinjin Zhao[1]* and Rebeqa Rivers[2]

[1]*Machine Learning Scientist, USA*

[2]*Amazon, Luxembourg, Europe*

***Corresponding author:** Jinjin Zhao, Machine Learning Scientist, USA

Received: September 07, 2022 **Published:** September 22, 2022

Abstract

Learning Experience Designers (LXDs) who use a cognitive model of learning to guide their design of online courses often create a knowledge map that articulates the learning outcomes that learners are expected to achieve in the course. LXDs engage in a variety of labor-intensive processes to unpack expertise and create a knowledge map. Further, the first iteration of the knowledge map often does not accurately represent the cognitive model of topic the course is designed to teach. In this work, we introduce a human-in-the-loop modeling process that leverages learning data and human expertise for discovering cognitive models of learning and constructing knowledge maps. In our proposed method, human actors - Machine Learning Scientists (MLS) and Learning Experience Designers (LXD) - interpret results from the modeling process and gain insights to optimize the course knowledge map and improve the course. This process includes applying machine learning (ML) algorithms to model binary learner assessment response data, converting the binary data into continuous data modeling tasks, optimizing the ML algorithm to deliver accurate, reliable assessment question clustering, and interpreting the modeling results by comparing the original knowledge map to the ML recommended clusters to identify design insights and make design recommendations for continuous improvement of the course.

Keywords: Computational cognitive modeling; learning behavior; human-in-the-loop; continuous improvement; cognitive process

Introduction

In learning experience design, structuring information and learning activities carefully is critical for success. Well-designed learning experiences enhance retention, stimulate prior knowledge, guide learners effectively through new con- tent, provide practice opportunities with targeted feedback, and assess the learner's performance throughout [1]. Additionally, organizing learning tasks to span a range of cognitive complexity supports learners to connect new knowledge to existing knowledge in a way that moves them from recall to comprehension, and finally to synthesis [2]. To source instructional content to support learners to build knowledge, LXDs use a variety of methods to identify and elicit knowledge from Subject Matter Experts (SMEs), such as Cognitive Task Analysis (CTA) [3], think-aloud protocols, and interviews. LXDs then organize the knowledge into a knowledge map, which articulates the learning outcomes (LOs) that learners are expected to achieve in the course. On the knowledge map, the LOs are associated with assessments that, through learner actions, produce data that is modeled to predict learner proficiency. The process of organizing knowledge into a knowledge map and writing well-aligned assessments for each LO is labor-intensive. The first iteration is often a native knowledge map, or a knowledge map that may not accurately represent the cognitive model of the topic the course is designed to teach. Additionally, LXDs do not always have the opportunity to return after learners have engaged with a course to use learning data to optimize the native knowledge map and assessments for the desired learning results. The process can

produce misalignment between LOs and assessments which can lead to inaccurate representations - to the learner and the LXD, alike - about the learner's current knowledge state. E-learning systems can support a data-driven process for knowledge map discovery. Previous work [4] shows promising results for using learning data and data modeling to discover learning patterns that can be utilized to evaluate and continuously improve knowledge map and course design. In this work, we introduce a human-in-the-loop cognitive modeling process using learner response data from assessment questions for discovering, interpreting, and making continuous improvement design recommendations to refine a native knowledge map for an e-learning course. In our proposed method, human actors.

Machine Learning Scientists (MLS) and Learning Experience Designers (LXD) - interpret results from the modeling process and gain insights to optimize the course knowledge map and design recommendations for continuous improvement of the course. This process includes applying a ML algorithm/architecture to model binary learner assessment response data, converting the binary data into continuous data modeling tasks so that ML algorithms can be leveraged, optimizing the ML algorithm to deliver accurate, reliable assessment question clustering, and interpreting the modeling results by comparing the original knowledge map to the ML recommended clusters to identify design insights for the knowledge map. The human- in-the-loop process introduced here is as follows:

a) MLS applies and reasons Autoencoders when modeling binary behavior data. Autoencoders applied in this work are well adapted in continuous data modeling tasks. There are few available techniques for modeling binary data.

b) MLS introduces domain knowledge (learning factors) and converts the binary data modeling problem into a continuous data modeling task so that the ML algorithms can be leveraged to derive useful data insights.

c) MLS uses commonsense knowledge to optimize the ML algorithm to get better result iteratively. In the beginning, a few algorithms are applied to find common learning patterns. After scrutinizing the common patterns, MLS incorporates the patterns into the algorithm fine tuning step to deliver more accurate, reliable clustering results.

d) LXD interprets the modeling results by comparing the clustering with the native knowledge map of the e- learning course, including LOs and sub-learning out- comes (sub-LOs), to identify design insights, or insights about how the learning design might be affecting the learner's implied cognitive processing.

e) LXD uses learning design domain knowledge to identify ways to use the design insights to create design recommendations for continuous improvement for the native knowledge map of the e-learning course.

We also introduce a use case study from a workforce learning environment to demonstrate the process of applying the proposed method for supporting knowledge map design and continuous course improvement.

Human-in-the-Loop Process

We describe the overall process in which human actors, MLS and LXD, leverage Autoencoders to discover data in- sights about the learning behavior and interpret the data insights for continuous improvement on course design. In the human-in-the-loop process presented in (Figure 1), human actors use ML algorithms of learners' assessment response data to infer the cognitive process, or mental action a learner experiences when answering a specific assessment question. For the sake of simplicity, the authors use the term 'cognitive process' throughout this paper, however it is always meant to indicate the inferred cognitive process, or the cognitive process deduced from ML analysis of learners' assessment response data. LXD Initial Learning Design and Learning Data. As a common design practice, a course is designed with a knowledge map that features three components:

a) LOs, or statements describing the desired state of a learner's knowledge at the end of the learning experience;

b) Sub-LOs, or more granular outcomes that define the specific skills and knowledge learners should be able to do or know; and

c) Assessment questions, each of which assesses knowledge for a specific sub-LO.

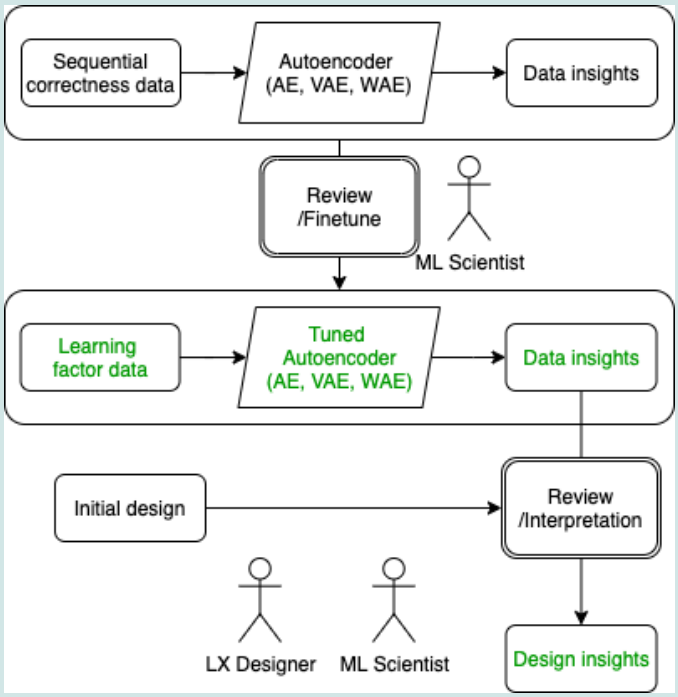

Figure 1: Human-in-the-loop Process for Cognitive Modeling.

All LOs and sub-LOs are written using Bloom's Revised Taxonomy, which indicate the degree of intended cognitive processing [5]. Each assessment is written to assess the specific degree of cognitive processing indicated by the Bloom's Taxonomy used in the sub-LO it supports. For example, sub-LO in the cognitive processing domain of "remember" (lower-order) are assessed with recall exercises whereas sub- LOs in the cognitive processing domain of "apply" (higher- order) are assessed with more complex assessments that require a greater degree of cognitive processing. Learning data including browsing, clicking, assessment attempts, and assessment correctness are collected as learners engage with the course. Assessment attempts and attempt correctness are commonly used to estimate the knowledge state of a learner [6]. In this work, we use the same data to discover learning patterns. Assessment attempt correctness is sequential behavior data. For example, if Learner A attempts to answer Assessment One three times with two wrong attempts and one correct attempt in sequence, the learner data would look like [Learner A, Assessment One, Attempt One, 0, Attempt Two, 0, Attempt Three, 1].

MLS Applies and Reasons Autoencoders. In the initial experimental phase, MLS applies Autoencoders to model the sequential correctness data to extract learning patterns. In order to mitigate bias from a singular algorithm, MLS applies a few techniques, including Autoencoder (AE), Variational Autoencoder (VAE), Wasserstein Auto-Encoders (WAE) [7]. All three methods are proposed to conduct representation learning. Each has advantages over the others in some contexts. For example, VAE and WAE are generative modeling methods that can be used to generate data on top of representation learning. WAE has a less restricted optimization objective function compared to VAE in constructing the representation and that helps produce a more robust result. We are interested in testing both generative and discriminative methods with different distribution modeling hypotheses in representation learning for an assessment. After getting the representation, a Gaussian Mixture Model [8] is applied to group the assessments into clusters that represent different learning patterns. As a measure, MLS uses the optimization trajectory of a training process to observe if the algorithm is converging towards a global minimum (a necessary condition). The metrics include optimization loss trajectory, loss value, and loss variance. MLS observes the metrics and identifies if the optimization procedure is progressing. If not, MLS reasons the loss trajectory with the neural net design.

MLS uses domain knowledge to transform learning data. After identifying the root cause of the optimization failure (not converging issue), the misalignment between the neural net design and the sequential binary data, MLS aims to resolve the misalignment by engineering learning data to fit the algorithm using domain knowledge. The reasons are: 1. The Autoencoders are well recognized in various research and applications in handling continuous data and are potentially powerful techniques if learning data can be transformed into the right format; 2. Some learning factors have been proposed in the literature and can be used to represent the sequential binary data as features. MLS uses commonsense knowledge to improve model performance. Building a sense-making solution, given a data set is an iterative process. MLS uses commonsense knowledge in terms of optimizing an ML algorithm to fit the modeling task and deliver a more accurate and reliable result. Initially, there is no guidance on model selection and no ground truth to guide the ML model towards the optimal

clustering solution. Thus, MLS conducts initial experiments to discover common patterns across various Autoencoders, then reviews and evaluates these learning patterns with LXD in order to design metrics to quantify accuracy and reliability. As common patterns are observed and reviewed, MLS use the patterns to guide the fine-tuning process of the model for more accurate modeling results. In other words, with the common patterns, an unsupervised training process is turned into a semi-supervised training process. In this work, it is manual effort that MLS observes the patterns from the result and use the patterns to evaluate the result from the fine-tuned models.

LXD interprets data insights for design insights. To leverage data insights and derive design insights, the LXD compares data insights (ML clusters) with the course knowledge map components and the cognitive processing domain verbs (i.e., Bloom's Revised Taxonomy). When comparing these elements, the LXD first considers alignments between ML clusters and the existing knowledge map and cognitive processes, asking questions such as "Do the ML cluster patterns align with existing delineators, such as sub-LOs, LOs, or cognitive process dimensions?", "Where ML clusters do not align with existing delineators, what factors in the course design or instructional content might be influencing this?" When encountering areas where the ML cluster patterns do not align with existing delineators, the LXD also uses this as a prompt to closely examine alignment within the original course design. In such cases, the LXD considers alignment between components of the knowledge map and cognitive processing domain verbs, asking questions such as "Does each assessment accurately assess its associated sub-LO and LO?" and "Are assessments written to correctly target the cognitive process dimensions learners need in order to demonstrate proficiency in the associated sub-LO and LO?" Through this process of comparing data insights with course elements and inspecting alignment, LXD is able to identify five distinct design insights, or specific misalignment within the knowledge map, cognitive process domain, and/or course instructional content.

Case Study

The course used in this experiment is an e-learning coaching course developed through multiple interviews with SMEs. The course includes four LOs, which address the following knowledge and capability areas: 1) identifying what effective coaching looks like; 2) knowing how to use a collection of specified coaching models and tools; 3) identifying examples of bias and bias-interruption strategies in coaching; and 4) using coaching to support talent management and development. Each sub-LO is measured by assessment questions ranging from one to three questions per sub-LO - that assess the learner's proficiency in the knowledge or skill represented. A collection of sub-LOs fall under one LO - ranging from two to six sub-LOs per LO. The proficiency for the sub-LOs roll up to estimate the proficiency for their overarching LO. Because coaching is a complex capability that requires navigating complicated inter-personal dynamics and social-emotional cues, assessments include relevant, real-world scenarios that prompt learners to activate prior knowledge and require learners to weigh various factors to solve a given challenge [9].

MLS Applies and Reasons Autoencoders

In this section, we introduce the initial experiments of applying AE, VAE, and WAE to discover the learning patterns from learning data. During the experiments, we identify and reason the limitation of Autoencoders when dealing with binary values. The observation is captured in (Table 1). The first row presents the loss trajectories for constructing binary data with AE, VAE, and WAE. For all three Autoencoders, both the loss value and the loss variance are far from the ideal (an ideal loss value "zero" means a perfect model fit for the observation data). Usually, a high and stable loss value indicates the optimization process is trapped at a local minimum. A high loss variance indicates the gradient descent direction is bounding back and forth at the local minimum. Both measures indicate the optimization process could not continue. As MLS dives into the activation function of neural networks, MLS realizes that the sigmoid function might be the reason that the binary value reconstruction is difficult. A sigmoid function [10] is a bounded, differentiable, real function that is defined for all real input and has a non-negative derivative at each point. It is widely used as the activation function of neural networks. It most often shows a return value in the range 0 to 1. In our application, the behavior data on assessment correctness is binary, either 0 or 1, which is on the boundary of the sigmoid function output value. Thus, the large loss value and loss variance indicate that a set of variables and a set of weights associated to the variables are difficult to fit the binary data within a latent space. We hypothesize it is a limitation for Autoencoders to deal with discrete binary value modeling.

Table 1: Relationship between LO, sub-LO, Assessment, and ML Clustering.

Learning Outcome (LO)	Sub-Learning Outcome (sLO)	Cognitive Process Dimension Verb	Assessment	ML Model Clusters
LO 1	sLO 1.1	Define	A 1.1.1	1
	sLO 1.2	Define	A 1.2.1	1
	sLO 1.3	Define	A 1.3.1	1
	sLO 1.4	Define	A 1.4.1	1
			A 1.4.2	0
	sLO 1.5	Recognize	A 1.5.1	0
			A 1.5.2	0
LO 2	sLO 2.1	Use	A 2.1.1	no data
	sLO 2.2	Use	A 2.2.1	1
			A 2.2.2	2
			A 2.2.3	2
			A 2.2.4	2
	sLO 2.3	Modify	A 2.3.1	0
			A 2.3.2	0
	sLO 2.4	Use	A 2.4.1	no data
	sLO 2.5	Recognize	A 2.5.1	0
LO 3	sLO 3.1	Identify	A 3.1.1	0
			A 3.1.2	2
	sLO 3.2	Identify	A 3.2.1	1
			A 3.2.2	1
			A 3.2.3	1
LO 4	sLO 4.1	Implement	A 4.1.1	1
			A 4.1.2	0
			A 4.1.3	1
			A 4.1.4	1
	sLO 4.2	Implement	A 4.2.1	1
			A 4.2.2	1
			A 4.2.3	1
	sLO 4.3	Implement	A 4.3.1	0
			A 4.3.2	0
			A 4.3.3	0

MLS uses Domain Knowledge to Transform Learning Data

Autoencoders are well adapted in applications where the numbers are continuous. We want to leverage the advantage of the Autoencoder framework to conduct the representation learning of the assessment from the behavior data. Autoencoders can be applied if the binary data can be represented in a continuous manner with some data engineering effort. In learning behavior modeling, some learning factors have been engineered and proposed to be a proxy of the sequential assessment correctness data. One of the learning factors is defined as 1-1/d where d is the attempt number until the first successful attempt. It assumes learners will make as many as attempts as they need to correctly solve the assessment before they move on to another section. The difficulty is defined as zero if a learner skips an assessment. The underlying assumption is when a learner skips as assessment, one has achieved proficiency on the skill and needs no more practice. The assumption aligns with our use case where learners take attempts until success or skip assessments when they think they've achieved proficiency. Thus, we use the learning difficulty factor to proxy the sequential binary correctness data and mark the difficulty factor value as zero if the assessment is skipped. In this way, the discrete binary value is

transformed into a continuous value. For example, if the sequential data is [0,0,1] - the learner has two failed attempts before the successful one, the difficulty factor is calculated as 1-1/3 (which is 2/3) to represent the sequential correctness data [0,0,1]. The representation construction results based on the continuous values for methods AE, VAE, and WAE are listed in the second row of Table 1. Loss variance is significantly reduced from all Autoencoders. It indicates that Autoencoders can learn a more stable latent representation when the learning difficulty factor is used to as a proxy of the sequential learning behavior data. Performance on loss value is also increased with AE and WAE and is similar with VAE. For VAE, the loss value stays around a certain value (50) and could not continue optimizing. It is expected because the VAE method forces each data point to follow a mixed Gaussian distribution (the KL divergence measure) and that conflicts with the reconstruction loss [11]. As a result, the loss value could not decrease when the reconstruction loss conflicts with the KL divergence loss. For AE and WAE, the loss value is reduced to near zero. It is also expected because AE aims to minimize the MSE and usually the loss can result close to zero as the optimization procedure progresses. WAE aims to find a surrogate model that can averagely model the data points, where the objective function is much less restricted compared to VAE. Thus, the reconstruction loss is less likely to conflict with the distribution discrepancy and the optimization of objective function can keep progressing.

MLS uses Commonsense Knowledge to Improve Model Performance

MLS performs initial experiments with various Autoencoders to discover common patterns. After evaluating the common patterns with LXD, MLS designs stability and accuracy metrics as measures to guide hyperparameter fine-tuning. Loss variance/loss value is one indicator of the convergence and result stability. But a low loss and a low loss variance do not guarantee a stable clustering result because the algorithm might end up in different local minimums in each run. If the result varies after each run, it is hard to claim the insights are solid and learning patterns are confidently discovered. ML solution consistency (reproducibility) indicates the stability of the insights. MLS uses clustering result stability as a measure (commonsense knowledge) to guide when it's time for result review. Accuracy is defined as the alignment between the model clustering result and sub-LOs from LXD. The sub-LOs will be refined with the data insights from the model result. In this use case, MLS observed that part of the clustering result, [0,1,0,1] for the first four assessments, is a common grouping result from various Autoencoder methods. After reviewing with LXD, the sub result is recognized as an accurate grouping for the first four assessments. Thus, in hyperparameter fine- tuning, MLS fine-tunes the result that can retain the grouping and also improve the stability [12].

This section describes how MLS fine-tunes the hyper-parameter to improve the accuracy and stability. The fine- tuned hyperparameters include latent dimension and hidden layer dimension. The latent dimension describes the number of latent factors/attributes that represent the assessment. The hidden layer size is the number of variables the Autoencoder creates to fit the data in the representation learning task. Latent dimension depends on the number of actual data points. The hidden layer dimension can be relatively larger than the latent dimension to provide flexibility in finding the accurate latent space. As results shown in (Table 2), both VAE and WAE achieve better accuracy with latent dim = 5 than latent dim = 20. AE performs better when both latent dimension and hidden dimension are reduced. We suspect that VAE and WAE could benefit from reducing the latent dimension because both models use generative modeling approaches and try to fit the data using the hypothesized surrogate model. The degree of freedom of VAE and WAE is less than that of AE because the surrogate models add constraints in finding the parameters in fitting the data. That said, it is possible that AE has too many variables than it needs in finding the right model fit. Thus, the result with hidden dim = 50 could be local optimum. Thus, we attempt to further reduce the hidden dimension for AE with the setting of AE (3). As a result, it achieves better accuracy compared to AE (1) and AE (2). The data points from the above experiment are limited. More experiments need to be conducted to scrutinize the hypothesis. On this attempt, we report the thought process of how MLS review, iterate, and optimize the Autoencoder framework to deliver a more accurate clustering result.

Table 2: Clustering with Commonsense knowledge.

Acc	AE (1)	AE (2)	AE (3)	VAE (1)	VAE (2)	WAE (1)	WAE (2)
1	0.4567	0.2496	0.7283	0.5800	0.6800	0.2111	0.6800
2	0.4828	0.2595	1.0	0.4083	0.4567	0.5213	0.5213
3	0.4828	0.5311	0.7283	0.4567	0.4828	0.2496	0.5213
4	0.6800	0.4083	0.7283	0.3241	0.8027	0.2595	0.4083
5	1.0	0.4083	0.4567	0.6800	0.5311	0.2595	0.4083
Validity	0.62	0.37	0.72	0.49	0.59	0.30	0.51
Reliability	0.053	0.013	0.036	0.019	0.021	0.015	0.012

AE (1)/VAE (1)/WAE (1): latent dim = 20, hidden dim = 50; AE (2)/VAE (2)/WAE (2): latent dim = 5, hidden dim = 50; AE (3): latent dim = 5, hidden dim = 10.

LXD Interprets Data Insights for Design Insights

LXD compares data insights (ML clusters) with the knowledge map, including LOs, sub-LOs, and cognitive processing domain verbs. The relationship between knowledge map com- ponents, cognitive processing domain verbs, and ML clusters is captured in (Figure 2). In the ML Model Clusters result, 'no data' means no learning data is observed for that assessment. By comparing data insights with the knowledge map, the LXD identifies design insights and subsequently makes design recommendations for continuous improvement of the course. The design insights and recommendations are captured in (Table 3). First, the LXD uses data insights to identify different cognitive processes learners experience and distinguish low- to-higher order cognitive processing within one LO. For example, within LO 1, the clusters indicate that learners experience two cognitive processes; sub-LOs with the verb "recognize" cluster together and sub-LOs with the verb "define" cluster together. Additionally, the clusters indicate that sub-LOs with the verb "define" require a higher-order cognitive process than sub-LOs with the verb "recognize". Second, the LXD uses data insights to infer two distinct cognitive processes learners experience when answering a narrative assessment sequence, or a series of assessments embedded throughout a scenario, in which learners are introduced to new pieces of information and prompted to select how they will respond to move the scenario toward a specific goal.

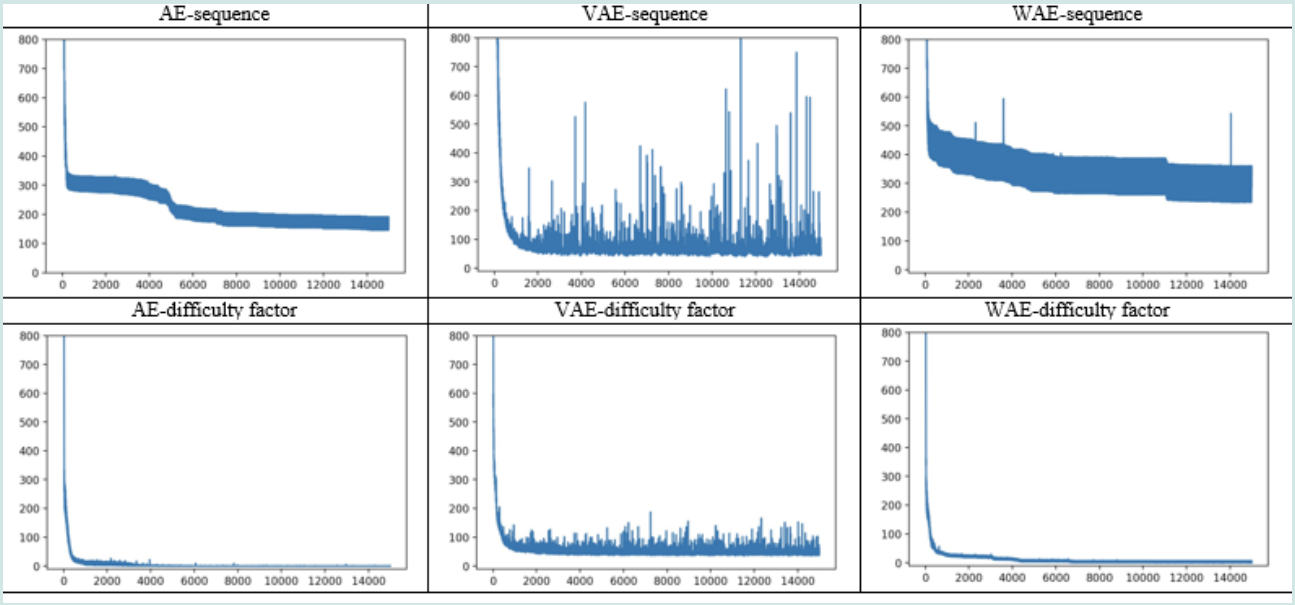

Figure 2: Optimization trajectory of computational assessment representation.

Table 3: Design Insight and Design Recommendation for Continuous Improvement.

Design Insight	Design Recommendation for Continuous Improvement
Learner's evidence different inferred cognitive processes within one LO	Revise knowledge map to align LO and sub-LO designations with learners' inferred cognitive processes
Learner's evidence different inferred cognitive processes within a narrative assessment sequence	Revise knowledge map to align LO and sub-LO designations with learners' inferred cognitive processes.
Misalignment between the inferred cognitive process prompted by an assessment and the sub-LO that assessment is meant to measure	Revise affected assessments to align with the Bloom's Taxonomy they are intended to measure
Potential knowledge and/or capability gaps in instructional content	Revise instructional content to address potential knowledge and/or capability gaps

In one example, four assessments in a narrative assessment sequence are intended to support the same sub-LO; however, assessments two, three, and four in the sequence clustered together, while the first assessment clustered with a different cognitive process. This clustering difference prompts the LXD to examine the structure of each assessment, which reveals that the first assessment requires a different level of contextualizing than the other assessments. Specifically, the first assessment introduces a scenario and prompts the learner to respond to the situation as a whole (e.g. "how would you respond in this scenario?"), which produces a higher-order cognitive process; however, the subsequent assessments introduce new variables and prompt learners to factor these into their response (e.g. "considering [new variable], how would you respond in this scenario?"), which produces a lower-order cognitive process [13]. Third, the LXD uses data insights to identify misalignment between some assessments and the sub-LO they are intended to measure. Where an assessment did not cluster as expected with sub- LOs, closer examination of the assessment reveals that it does not align as intended with the original knowledge map. For example, one assessment within LO 1 was originally meant to support a sub-LO with the verb "define", but clustered with sub-LOs with the verb "recognize". Upon examining the assessment, the LXD finds that the assessment is incorrectly written in a way that assesses "recognize" and does not assess "define", as intended. Finally, the LXD uses data insights to identify potential knowledge and/or capability gaps in the course instructional content. Inferring design insights about instructional content gaps was not an intended focus of the investigation but arises as a pleasant surprise when comparing the cluster data against the knowledge map. The fourth LO includes three sub-LOs, which are assessed by ten assessments.

All ten assessments are written to prompt a similar cognitive process. Although six of the assessments clustered together (Cluster 1), four of the assessments clustered into a second group and indicate a higher-order cognitive process (Cluster 0). This unexpected clustering prompts the LXD to examine the content of Cluster 1 assessments and Cluster 0 assessments in an effort to determine why they cluster into two groups instead of all together, as expected. Upon initial examination, the LXD finds that all ten assessments are written using a similar structure and with strong alignment to the Bloom's Revised Taxonomy used in the sub-LOs and LO. This rules out learning design variations as the cause for the different cluster groupings. Next, the LXD accesses subject matter expertise on the content of the assessments to determine whether clustering might be influenced by nuance within the subject matter. This investigation reveals that Cluster 1 assessments measure areas of coaching practice meant to support measurable workplace outcomes; however, Cluster 0 assessments measure areas of coaching practice that require a higher tolerance for navigating ambiguous conversations. For example, Cluster 1 assessments address coaching to improve measurable metrics of an employee's job performance, such as meeting project deadlines. The Cluster 0 assessments, however, address coaching for career development, which does not generally support specific deliverable, but rather explores the employee's personal goals, preferences, and even personality-fit for potential career paths [14].

Conclusion, Future Work, and Discussion

In this work, we introduce a human-in-the-loop process for discovering cognitive models of learning, in which MLS models the data for data insights and LXD interprets data insights to derive design insights for continuous improvement of the e-learning course. We also provide a use case study for applying the proposed method to derive design insights and make design recommendations for continuous improvement of a course. As future work, we would like to conduct A/B experiments to test the LXD's design change recommendations and whether these might potentially improve learning efficacy. We would like to continue to automate the modeling, fine tuning, and reviewing process to scale the process to other use cases and applications.

Acknowledgement

We would like to thank Amazon's Learning Science and Engineering organization, particularly the Learning Design, Assessment, and Science Team for helping us during the course of this project.

References

1. Shreyansh Bhatt, Jinjin Zhao, Candace Thille, Dawn Zimmaro, Neelesh Gattani (2020) A novel approach for knowledge state representation and prediction. In Proceedings of the Seventh ACM Conference on Learning @ Scale pp. 353-356.
2. Hao Cen, Kenneth Koedinger, Brian Junker (2006) Learning factors analysis–a general method for cognitive model evaluation and improvement. In International Conference on Intelligent Tutoring Systems pp. 164-175.
3. Albert T Corbett, John R Anderson (1994) Knowledge tracing: Modeling the acquisition of procedural knowledge. User modeling and user-adapted interaction 4(4): 253-278.
4. Peter E Doolittle, David Hicks (2003) Constructivism as a theoretical foundation for the use of technology in social studies. Theory & Research in Social Education 31(1): 72-104.
5. Robert M Gagne, Leslie J Briggs (1974) Principles of instructional design. Holt, Rinehart & Winston, USA.
6. Diederik P Kingma, Max Welling (2013) Auto-encoding variational baye. arXiv preprint, 1312.6114.
7. David R Krathwohl (2002) A revision of bloom's taxonomy: An overview. Theory into practice 41(4): 212-218.
8. Bruce G Lindsay (1995) Mixture models: theory, geometry, and applications. In NSF-CBMS regional conference series in probability and statistics pages i-163. JSTOR.
9. David E Rumelhart, Geoffrey E Hinton, Ronald J Williams (1988) Learning internal representations by error propagation. Readings in Cognitive Science pp. 399-421.
10. Jan Maarten Schraagen, Susan F Chipman, and Valerie L Shalin (2000) Cognitive task analysis. Psychology Press.
11. Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, Bernhard Schoelkopf (2017) Wasserstein auto-encoders.

12. Wikipedia (2021) Sigmoid function. The free encyclopedia.

13. Jinjin Zhao, Candace Thille, Neelesh Gattani, Dawn Zimmaro (2021) A novel framework for discovering cognitive models of learning. Proceedings of the Eighth ACM Conference on Learning @ Scale pp. 271-274.

14. Jinjin Zhao, Candace Thille, and Dawn Zimmaro (2021) Data mining for discovering cognitive models of learning. In 2021 The 5th International Conference on Advances in Artificial Intelligence (ICAAI) pp. 130-139.

DOI: 10.32474/CTCSA.2022.02.000137