Variational Deep Knowledge Tracing for Language Learning

Sherry Ruan
ssruan@stanford.edu
Stanford University
Stanford, California, United States

Wei Wei
wewei@google.com
Google
Sunnyvale, California, United States

James A. Landay
landay@stanford.edu
Stanford University
Stanford, California, United States

ABSTRACT

Deep Knowledge Tracing (DKT), which traces a student's knowledge change using deep recurrent neural networks, is widely adopted in student cognitive modeling. Current DKT models only predict a student's performance based on the observed learning history. However, a student's learning processes often contain latent events not directly observable in the learning history, such as partial understanding, making slips, and guessing answers. Current DKT models fail to model this kind of stochasticity in the learning process. To address this issue, we propose Variational Deep Knowledge Tracing (VDKT), a latent variable DKT model that incorporates stochasticity into DKT through latent variables. We show that VDKT outperforms both a sequence-to-sequence DKT baseline and previous SoTA methods on MAE, F1, and AUC by evaluating our approach on two Duolingo language learning datasets. We also draw various interpretable analyses from VDKT and offer insights into students' stochastic behaviors in language learning.

CCS CONCEPTS

• **Social and professional topics** → **Student assessment**; • **Computing methodologies** → *Neural networks*; • **Mathematics of computing** → *Variational methods.*

KEYWORDS

knowledge tracing, language learning, student modeling, deep learning, variational inference

ACM Reference Format:
Sherry Ruan, Wei Wei, and James A. Landay. 2021. Variational Deep Knowledge Tracing for Language Learning. In *LAK21: 11th International Learning Analytics and Knowledge Conference (LAK21), April 12–16, 2021, Irvine, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3448139.3448170

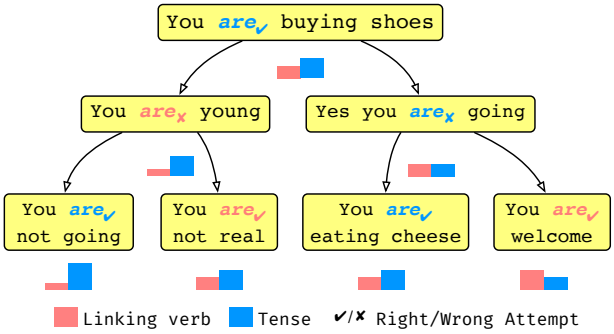

Figure 1: A probabilistic knowledge tracing system tracks the distribution of two concepts concerning the word *are*: understanding it as a linking verb and understanding it in the present progressive tense. ✓ and ✗ represent whether a student answers a certain question correctly or not. The height of the red and blue knowledge bars represents the system's beliefs in the student's knowledge about two different grammar rules over time. Higher bars represent higher probabilities of mastering the concept. Initially, the system has an equal belief in the student's knowledge of two concepts. As the student practices more problems, the system grows its confidence and updates its estimates based on the student's performance.

1 INTRODUCTION

Knowledge tracing is the task of modeling a student's knowledge acquisition and loss process based on the student's past trajectories of interactions with a learning system [2]. Knowledge tracing on language learning is particularly interesting for two reasons. First, modern language learning systems such as Duolingo [37, 38] provide vast amounts of interaction data that fuel the performance of advanced machine learning models. Second, since a language usually takes a long period to learn and retain, most of the language learning interaction trajectories are long and noisy sequences. As a result, a unique challenge is posed to apply knowledge tracing models to language learning.

Many existing methods are based on psychological insights. For example, Pimsleur [29] and Leitner [22] used psycho-linguistically inspired models with hand-picked parameters. Corbett and Anderson [7] proposed Bayesian Knowledge Tracing (BKT), a well-known Bayesian-based method for knowledge tracing. More recently, Settles and Meeder [38] and Renduchintala et al. [31] investigated more advanced statistical learning models in combination with psycho-linguistic theories. However, due to the limitations of parameter size and the form of distributions they can choose from, these models could not effectively capture sophisticated patterns of the data provided by language learning systems.

To improve the performance of knowledge tracing models, researchers have built RNN-based knowledge tracing models, e.g., Deep Knowledge Tracing (DKT) by Piech et al. [28]. Current DKT models achieve superior performance compared to the aforementioned methods in many domains [28, 39, 42, 47] including language learning [37]. However, unlike latent variable models such as BKT models, a DKT model works purely on observed data and ignores

LAK21, April 12–16, 2021, Irvine, CA, USA

ACM ISBN 978-1-4503-8935-8/21/04.
https://doi.org/10.1145/3448139.3448170

the underlying scholastic patterns made by the unobserved latent variables. Such a simplification makes it difficult for DKT methods to model language learning datasets, which are often stochastic [10].

A student's interaction with a language learning system is noisy and uncertain. For instance, one could slip on questions even when already mastering the underlying concepts. One could also guess an answer to a question correctly without fully understanding the corresponding concepts. Moreover, a student learns by acquiring new knowledge, understanding parts of it, and then forgetting parts of the learned concepts. As a result, the student's knowledge state fluctuates during the learning process. All the fluctuations lead to the student's stochastic behaviors when practicing knowledge. Therefore, the student's raw observed behavior is not always the best measure (or loss function in deep knowledge tracing) to optimize for, and these many latent events need to be uncovered at an abstraction level higher than the raw observance. Unfortunately, current DKT models only learn from the observed inputs and cannot adequately generalize the stochastic learning processes that contain these latent events.

As further illustrated in Figure 1, a student is learning two linguistic rules pertaining to the word *are*. The first one uses *are* as a linking verb followed by a noun phrase or an adjective, e.g., "you *are* not real". The second one uses *are* to represent the present progressive tense, e.g., "you *are* buying shoes". The student needs to answer a series of questions that involve the use of *are* in sentences. A beginning language learner is likely to be confused by these rules and has uncertain learning behaviors when practicing the word *are*. Although a DKT model could implicitly model some stochastic behaviors, it would largely ignore the latent information and fail at generalizing these learner behaviors. In contrast, a probabilistic knowledge tracing system is capable of maintaining a probabilistic distribution of a student's knowledge representations (as demonstrated by red and blue belief bars in Figure 1) and separate noisy events from learning actual knowledge representations to achieve better modeling performance.

Much of the previous work attempted to incorporate latent information by including richer features, e.g., problem tags, linguistic features, and user information [45, 48], into DKT. However, the critical difficulty in representing uncertainty still exists due to the non-existence of latent variables in current DKT models. In this work, we tackle the challenge by introducing latent variables to DKT to capture the stochasticity and latent information existing in a student's learning process. To our knowledge, our proposed model is the first comprehensive attempt towards incorporating uncertainty into deep learning-based knowledge tracing models.

Our contribution is three-fold: (1) We built a baseline Sequence-to-Sequence Deep Knowledge Tracing model (SDKT) adapted to the language learning domain. It systematically combines a student's sequential learning trajectories with question-level linguistic features and user-level features. (2) We presented **Variational Deep Knowledge Tracing (VDKT)**, a latent variable model that introduces variation into the SDKT model. We evaluated VDKT and SDKT on two large second language learning datasets collected by Duolingo: one for vocabulary learning and the other for sentence learning. Results show that adding latent variables to the SDKT model substantially improved performance on MAE, F1, and AUC. Additionally, VDKT achieved state-of-the-art (SoTA) performance on these two real-world language learning datasets. (3) We performed various analyses comparing VDKT and SDKT, including creating visualizations to make these analyses more interpretable. These analyses provide further insights towards a better understanding of latent information in language learning modeling. Our work has many real-world educational applications including helping to better assess student learning, understanding students' stochastic behavior, and providing insight into students' underlying learning processes. All of these are directly related to the interests of the learning analytics community.

2 RELATED WORK

Our work tackles the classical knowledge tracing problem and leverages variational autoencoders, so we summarize prior work in these two related areas: knowledge tracing and variational encoder-decoder.

2.1 Knowledge Tracing

The goal of knowledge tracing is to learn a representation of student knowledge over time based on their past learning performance. The representation is evaluated by predicting student performance on future questions. The two most popular families of knowledge tracing models are Bayesian knowledge tracing (BKT) [7], based upon Hidden Markov Models (HMMs) [12], and Deep Knowledge Tracing (DKT) [28], back-boned by sequence to sequence models [40].

BKT, the long-standing method for knowledge tracing, uses HMMs to model a student's prior knowledge and learning speed. To model latent information in a student's stochastic learning process, BKT introduces a *slip* parameter (fail a question despite mastering the skill) and a *guess* parameter (guess a question correctly despite not yet mastering the skill). These parameters together with other HMM parameters are typically optimized using past data by expectation-maximization algorithms [8]. BKT infers a student's future performance based on these estimated parameters. Many extensions to the original BKT have been proposed. For example, latent factors such as individual differences could be integrated into BKT to enhance model performance [19]. Heathcote et al. [14] associated the student performance on multiple skill questions with all required skills by listing the performance sequence repeatedly. Khajah et al. [18] incorporated a forgetting parameter to allow for forgetting of skills. Overall, the main advantage of BKT is that it has strong prior assumptions, which make states easily interpretable. However, BKT generally requires manual topic labeling, has overly simplistic independent topics assumptions, and often underperforms DKT on large educational datasets [28].

DKT has been developed in recent years with the resurgence of recurrent neural networks (RNNs) [34]. RNNs are a natural way to encode a student's interactions - the knowledge states are represented as hidden states in an RNN and forwarded to future time steps. DKT has shown more promising results than traditional models on various domains including mathematics [28, 44], programming exercises [42], engineering statics [47], and language learning [17, 27]. Many variations of the original DKT [28] have emerged in recent years to augment the modeling performance. Su

et al. [39] used a second recurrent neural network to encode question text to better characterize the relationship between questions. Zhang et al. [47] used two static matrices to store knowledge concepts and students' understanding of concepts separately so as to uncover some underlying concepts. However, all the existing DKT models only learn from the raw observed data. As a result, they cannot capture the latent student behaviors such as guessing or slipping smoothly, which on the contrary can be naturally modeled in BKTs. To this end, our VDKT can be regarded as the first step towards unifying DKT with BKT.

2.2 Variational Encoder-Decoder

Autoencoders [13, 15, 35] encode inputs to low-dimensional intermediate representations and then decode back to the original inputs. They share similar technological traits to our problem because of its encoder-decoder architecture. In knowledge tracing, the student's historical performance is encoded using an encoder, and when predicting the student's future performance, a decoder is used to generate predictions based on the encoded representation of the student's learning trajectories.

However, the difficulty with autoencoders is that the intermediate latent representation is not continuous, hence making the decoded results less accurate. Recently, variational versions of autoencoders [20, 32] have been proposed to impose probabilistic distributions on latent representations to solve this issue. Instead of encoding each input into a distinct representation, a variational autoencoder (VAE) encodes inputs into a probabilistic distribution, and the decoder samples from this distribution when generating outputs. VAEs have been proven to be effective in many application domains such as image modeling [30] and music [33]. When applied to natural language, variational methods can be used to better model the stochastic nature of text and have produced promising results in areas including sentence generation [4], question answering [26], and dialogue systems [9, 36].

There are two major ways to incorporate latent variables into the encoder-decoder architecture. One is to add variations using a global latent variable that encodes either the entire input sequence [4] or the entire output sequence [5]. The other approach is to incorporate a local latent variable that is unique to each sample [36]. Our method aligns more closely with the second line of variational sequence modeling since uncertainty may occur every time a student attempts a question.

3 VARIATIONAL DEEP KNOWLEDGE TRACING

We first present Sequence-to-Sequence Deep Knowledge Tracing (SDKT), a sequence-to-sequence deep knowledge tracing model that naturally separates a student's learning history and future performance through the encoder-decoder structure. The encoder is used to encode a student's learning history and the decoder is used to predict the student's future performance. SDKT also systematically incorporates a wide range of linguistic and user-specific features to aid the characterization of students' knowledge states. Figure 2abc shows a model diagram of SDKT. Building upon SDKT, we then present Variational Deep Knowledge Tracing (VDKT) by replacing the regular prediction layer in SDKT with a variational inference layer as shown in Figure 2d.

3.1 Notations and Embedding Representations

As can be seen from Figure 2a, given a student's learning history from time step 1 to $k-1$, the goal of knowledge tracing is to predict the student's performance on all the questions after the time step k (inclusive). We use $LSTM$ and $BiLSTM$ as basic building blocks in our model where $LSTM$ stands for long short-term memory [16] and $BiLSTM$ stands for bidirectional LSTM [23].

The first task we need to perform is to convert input features, including questions students worked on, answers students attempted, and other meta-level features such as linguistic features and students' personal information, to vectors. Similar to word embedding [41], we use trainable embeddings to convert inputs to vectors. These input embedding representations are obtained by passing question IDs, binary answers (correct or incorrect), and meta-level feature converted IDs to trainable embedding layers that are optimized together with the rest of the neural network. At a time step t, we use q_t to denote the embedded question representation, a_t to denote the embedded student answer representation, and m_t to denote the embedded meta-level representation.

We use S_t to denote a state tuple that contains two vectors, c_t and h_t, the memory and hidden state of an LSTM cell, respectively. We use f and g to denote activation functions and W and b to denote weight matrices and bias terms in dense layers. Lastly, we use $softmax$ to denote the softmax [13] function, $(,)$ to denote tuples, and $[,]$ to denote concatenations.

3.2 Sequence-to-Sequence Deep Knowledge Tracing (SDKT)

We employ an encoder-decoder architecture for the SDKT model. The encoder (Figure 2a) first learns forward and backward representations S_t^f and S_t^b over the past learning trajectories $q_{1..t}$ and $a_{1..t}$ by using a $BiLSTM$ where $LSTM_F$ is the forward one and $LSTM_B$ is the backward one. The encoder is mathematically defined as follows: for $t = 1 \cdots k-1$,

$$S_t^f = LSTM_F([a_t, q_t], S_{t-1}^f) \tag{1}$$

$$S_{t-1}^b = LSTM_B([a_t, q_t], S_t^b) \tag{2}$$

The outputs are further encoded using a one-layer $LSTM$ to form S_t. The decoder design is divided into two versions, a concurrent decoder and an auto-regressive decoder, to accommodate the formats of the two language learning datasets described later in Section 4. The **concurrent decoder** assumes no dependencies between questions and predict all answers at the same time. As illustrated in Figure 2b, m_t packs meta-data related to the question q_t and enters a regular prediction layer together with h_t. The concurrent decoder is formulated as follows: for $t \geq k$,

$$S_t = LSTM(q_t, S_{t-1}) = (c_t, h_t) \tag{3}$$

$$o_t = [m_t, h_t] \tag{4}$$

The **auto-regressive decoder** assumes temporal dependencies between outputs. It is widely used in sequence-to-sequence modeling, e.g., machine translation [43]. Figure 2c shows the illustration

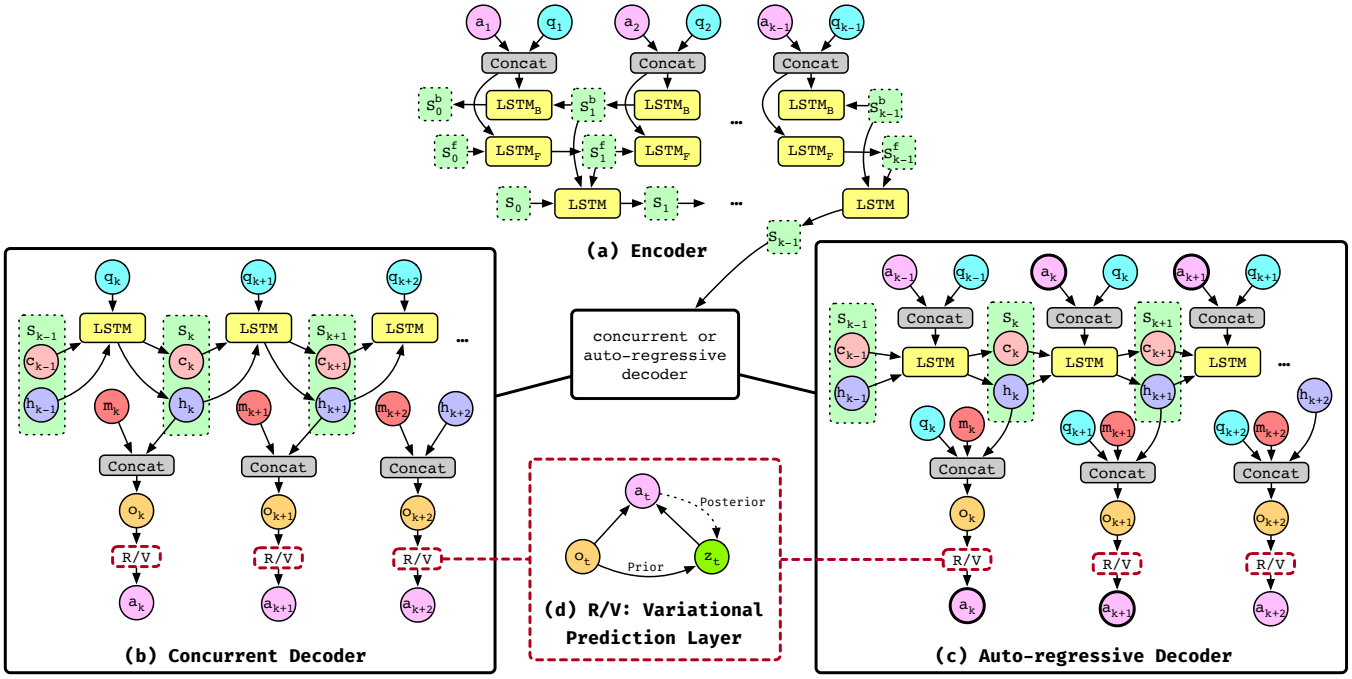

Figure 2: Model diagrams for SDKT (abc) and VDKT (abcd). Circular nodes are tensors, and square nodes are functions. Function R/V represents a regular or variational prediction layer. In SDKT, a bidirectional encoder (a) is used to encode student learning history, and a decoder is used for predicting the student's performance on future questions. The concurrent decoder (b) is designed for vocabulary learning (the HLD dataset), and the auto-regressive decoder (c) is designed for sentence learning (the SLAM dataset). VDKT shares the same encoder with SDKT and uses a variational prediction layer (d) instead of the regular final prediction layer in the decoder.

of the auto-regressive decoder. The mathematical formulation is as follows: for $t \geq k$,

$$S_t = LSTM([a_{t-1}, q_{t-1}], S_{t-1}) = (c_t, h_t) \tag{5}$$

$$o_t = [q_t, m_t, h_t] \tag{6}$$

where o_t is a concatenation of q_t, m_t, and h_t, denoting the input passed to **a regular or variational prediction layer** denoted by R/V in Figure 2d. The regular prediction layer, which is a dense layer followed by a $softmax$ function, makes final predictions on students' answers.

$$a_t = softmax(f(Wo_t + b)) \tag{7}$$

3.3 Variational Deep Knowledge Tracing (VDKT)

The deterministic nature of SDKT makes it difficult to model stochastic patterns in knowledge tracing. To address this limitation, we augment SDKT by adding a latent variable, z_t, to each time step. We first turn the variational input o_t into the mean θ_t and co-variance Σ_t of a Gaussian distributed random variable. Specifically, we use $\theta_t = g(W^\theta o_t + b^\theta)$ and $\Sigma_t = g(W^\Sigma o_t + b^\Sigma)$. Then the stochasticity in VDKT is represented using a random variable z_t drawn from the Gaussian distribution:

$$z_t \sim \mathcal{N}(\theta_t, \Sigma_t) \tag{8}$$

We calculate the final prediction, a_t, by passing both z_t and o_t to a dense layer followed by a $softmax$ function.

$$a_t = softmax(f(W^z[z_t, o_t] + b^z)) \tag{9}$$

Figure 2d shows performing these steps in the variational prediction layer.

3.4 Scalable Inference

We train the VDKT model by maximizing the Evidence Lower Bound (ELBO) [20]. The ELBO $\mathcal{L}$ is defined as:

$$\mathcal{L} = -\lambda \, \mathrm{KL}(q(z_t|a_t, o_t) \,||\, p(z_t|o_t)) + \mathbb{E}_{z_t \sim q} \log P(a_t|z_t, o_t) \tag{10}$$

where KL stands for Kullback-Leibler divergence [21], an effective way of measuring the distance of two probability distributions. λ is a variable weight to the KL term first introduced by Bowman et al. [4] to control the cost annealing schedule. p is the prior distribution of the latent variable z_t, and q is the proposed approximation to the posterior distribution of z_t. We parametrize q by conditioning z_t on a_t and o_t and then using a set of parameters to instantiate the mean and co-variance in a Gaussian distribution.

$$q(z_t|a_t, o_t) = \mathcal{N}(g(W^{\theta'}[a_t, o_t] + b^{\theta'}), g(W^{\Sigma'}[a_t, o_t] + b^{\Sigma'})) \tag{11}$$

3.5 Optimization

To balance the KL term and the marginal likelihood term in Equation 10, we change the variable weight λ during training. At the beginning of training, we focus on the loss function of the marginal likelihood term. As training proceeds, we gradually increase the term λ to allow the model to optimize more for the KL term. We borrowed this technique from Bowman et al. [4] and found it effective in our case as well. The schedule we chose was $\lambda = (\tanh((\text{global step} - \alpha)/\beta) + 1)/2$. Global step is the total number of training iterations and α and β are tuned based on datasets.

4 DATASETS

We evaluated VDKT against SDKT on two large real-world language learning datasets collected by Duolingo, the Historical Log dataset (HLD) for vocabulary learning [38] and the Second Language Acquisition Modelling dataset (SLAM) for sentence learning [37]. Both datasets are split such that each training/testing sample consists of a source sequence (a student's historical performance) and a target sequence (the same student's future performance that needs to be predicted). We apply the encoder in Figure 2a to the source sequence in both datasets, the concurrent decoder in Figure 2b to the target sequence in the Duolingo HLD dataset, and the auto-regressive decoder in Figure 2c to the target sequence in the Duolingo SLAM dataset.

4.1 Duolingo Historical Log Dataset (HLD)

The Duolingo HLD dataset[1] contains two weeks of student log data, which is about 13 million student-word session traces. A word session is a data instance in a sequence. Within each session, a student practiced a word multiple times. The aggregated recall rate was provided as the ground truth answer label. The task for this dataset is to predict the recall rates for future student-word sessions. In our experiments, we treated one word session as one question in our model. Thus, the word practiced in a session was q, the aggregated recall rate was a, and related meta information was m. Since there is no dependency between sessions, we used the concurrent decoder to model student learning in this dataset.

4.2 Duolingo Second Language Acquisition Modeling Dataset (SLAM)

The Duolingo SLAM dataset[2] contains about 6000 language learners' first 30 days of learning data on Duolingo. The dataset covers three languages: English, Spanish, and French. In total, students practiced 1,100,000 English exercises, 900,000 Spanish exercises, and 412,000 French exercises. Within each exercise, students practiced multiple words at the same time to form a sentence, i.e., *my ducks do not swim every day*. Duolingo's internal grading algorithm [37] determines if the student's answer to each word is correct or not. Because of the strong dependency between each word, we used the auto-regressive decoder to model each exercise. We formulated each word in an exercise as q and the corresponding answer label as a. All the other information, such as student IDs, exercise types, and morpho-syntactic features, was recorded in m using embedding matrices.

The prediction task for the Duolingo SLAM dataset is to predict binary labels a for words in future exercises. The Duolingo SLAM dataset uses 0 for correct labels and 1 for incorrect labels; the Duolingo HLD dataset uses the opposite labels. To make it consistent across two datasets, we followed the labeling in the Duolingo HLD dataset and reversed the labels in the Duolingo SLAM dataset. We report the results in the same manner throughout the rest of the paper.

Additionally, the train-test splits for both datasets were provided by Duolingo [37, 38]. Our models were trained on the same training set and evaluated on the same testing set as those of prior work to ensure a fair comparison.

5 EVALUATION

We present technical implementation details of our SDKT and VDKT models, as well as quantitative evaluations results on the Duolingo HLD and SLAM datasets.

5.1 Experiment Setup

We implemented our method using TensorFlow 1.7 [1] and trained on an NVIDIA Volta 100 SXM2 GPU.

Dataset	SDKT Params	VDKT Params	Difference
HLD	17.7M	17.8M	+0.5%
SLAM	2.5M	2.7M	+8.0%

Table 1: The total number of parameters in our SDKT and VDKT models on the Duolingo HLD and SLAM datasets. SDKT and VDKT have a comparable number of parameters on both datasets.

In our experiments, we carefully controlled the capacity of each layer and the number of total layers to conduct a fair comparison between SDKT and VDKT. Comparing to SDKT, the only additional parameters VDKT possessed were weights and bias terms in the prior distribution generation: W^{θ}, b^{θ}, W^{Σ}, b^{Σ} and in the approximated posterior distribution generation: $W^{\theta'}$, $b^{\theta'}$, $W^{\Sigma'}$, $b^{\Sigma'}$. Both encoder and decoder LSTMs consisted of 128 hidden units. The vector size of the word embedding layer was 300. All the other embedding layers used a vector size of 64. The latent variables and the weights and the bias terms in the dense layers all contained 128 units. Each dense layer used tanh as the activation function. The dropout rate was 0.2. We used FastText [3] to initialize our word embedding for multiple languages covered in the dataset. Table 1 shows the number of parameters in SDKT and VDKT. We can see that SDKT and VDKT have a comparable number of parameters on both datasets.

In the presented results, an upward arrow indicates that a higher score is better. Similarly, a downward arrow indicates that a lower score is better. The best result for each metric is in bold. We use $*$ to indicate if the predicted results between VDKT and the model being compared have a statistically significant difference ($p < 0.05$). To compute the statistical significance, we followed Settles and

[1] Available at github.com/duolingo/halflife-regression
[2] Available at sharedtask.duolingo.com

Meeder [38] by using a t-test for the Duolingo HLD dataset and used McNemar's test [25] for the Duolingo SLAM dataset.[3]

5.2 Duolingo HLD Dataset Evaluation Results

We followed the settings of Settles and Meeder [38] by evaluating the models using Mean Absolute Error (MAE) and Area Under Curve (AUC). Prior to our work, the Half Life Regression (HLR) model [38] achieved the best (lowest) MAE[4]; the Leitner model [22] achieved the best (highest) AUC. Table 2 shows that our VDKT model outperformed the previous state-of-the-art results on both MAE and AUC simultaneously. The differences between VDK and all the other models were statistically significant, as annotated by $*$. Our baseline model, SDKT, performed close to the state-of-the-art results reported in Settles and Meeder [38] but could not achieve comparable performance to VDKT, the variational version of it. This suggests that randomness could play an important role in predicting student's performance on this dataset.

Since there are two metrics, MAE and AUC, to evaluate the model, we noticed that there was a tradeoff between minimizing MAE and maximizing AUC. As shown in Figure 3, we obtained multiple MAE and AUC result pairs based on parameter tuning and where to stop the training. When deciding which MAE and AUC result pair to report in Table 2, we made a selection by minimizing the sum of the vertical and horizontal distance to the result achieved by the HLR model in Settles and Meeder [38].

	Model	MAE ↓	AUC ↑
Settles & Meeder [38]	*HLR*	**0.128***	0.538
	LR -lex	0.211*	0.514
	Pimsleur	0.445*	0.510
	Leitner	0.235*	**0.542**
Ours	*SDKT*	0.127*	0.540
	VDKT	**0.115**	**0.552**

Table 2: Evaluation results of SDKT and VDKT on the Duolingo HLD dataset compared with previous results in literature [38]. All the models are single models. The best MAE was achieved by the Half Life Regression (HLR) model and the best AUC was achieved by the Leitner system [38]. VDKT outperformed SDKT and previous state-of-the-art results on MAE and AUC simultaneously. The best result for each metric is in bold. * indicates if the predicted results between VDKT and the model being compared have a statistically significant difference (p<0.05).

5.3 Duolingo SLAM Dataset Evaluation Results

Following the norm in the Duolingo SLAM competition [37], we reported F1 and AUC scores and compared model performance on each language. Osika et al. [27] achieved the state-of-the-art F1 and AUC results on this dataset by using an ensemble model combining a Gradient Boosted Decision Tree (GBDT) [11] and a customized LSTM model extending Piech et al. [28]'s DKT architecture. Since we wanted to focus on single model performance and separate out the performance gain due to GBDTs, we performed three types of comparisons: single model, self ensemble, and GBDT ensemble. Self ensemble means models are ensembled with themselves, and GBDT ensemble means models are ensembled with GBDT models, which was what Osika et al. [27] used to achieve their state-of-the-art results. Our SDKT and VDKT models were trained separately on each of the language subsets. Table 3 shows that VDKT outperformed the previous state-of-the-art results reported by Osika et al. [27] in 8 out of 9 comparisons. VDKT also outperformed the non variational version, SDKT, in all the 18 experiments. All the differences between SDKT and VDKT were statistically significant, as shown by * in Table 3.[5]

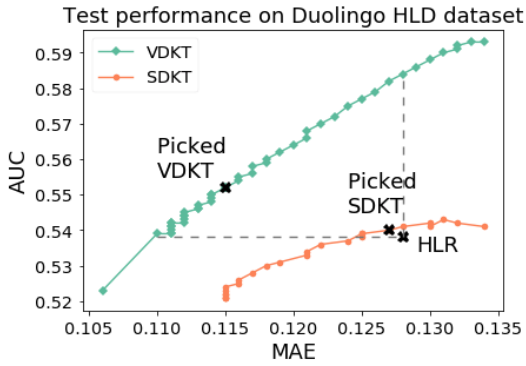

Figure 3: The Pareto Front graph of MAE and AUC evaluation results of HLR, SDKT, and VDKT on the Duolingo HLD dataset. Green and Orange dots represent different MAE and AUC pairs achieved by our SDKT and DKT models, respectively. The dot annotated with "HLR" is the result achieved by the Half Life Regression model in Settles and Meeder [38]. VDKT significantly outperformed SDKT and HLR on MAE and AUC simultaneously. The SDKT and VDKT results reported in Table 2 were selected by minimizing the sum of the vertical and horizontal distance to the result of HLR in the graph.

6 ANALYSIS AND DISCUSSION

In addition to showing modeling performance gains, we present quantitative analysis and qualitative visualizations to illustrate additional advantages of adding latent variables to knowledge tracing modeling.

6.1 Benefits of Probabilistic Representation

Going back to the motivating example discussed in Figure 1 where we presented a simple learning scenario that involves two uses of the word *are*, we explain here how the VDKT model presented in

[3]We used different statistical tests since ground truth answer labels are real numbers in the Duolingo HLD dataset while they are binary labels in the Duolingo SLAM dataset.
[4]The state-of-the-art MAE and AUC results on the Duolingo HLD dataset were achieved in Settles and Meeder [38]. Recently, after we finished this work, we found a newly published method by Zaidi et al. [46] that achieved a lower MAE than Settles and Meeder. However, Zaidi et al. did not report their AUC results. As a result, we still compared our results to the ones in Settles and Meeder.
[5]We only computed statistical significance for results between SDKT and VDKT since the exact predictions in Osika et al. [27] are not available.

		English		French		Spanish	
Type	**Model**	**F1 ↑**	**AUC ↑**	**F1 ↑**	**AUC ↑**	**F1 ↑**	**AUC ↑**
Single Model	*SDKT*	0.531*	0.844	0.542*	0.837	0.490*	0.813
	VDKT	**0.539**	**0.847**	**0.550**	**0.839**	**0.505**	**0.818**
Self Ensemble	*Osika et al. [27]*	–	0.851	–	0.841	–	**0.830**
	SDKT	0.543*	0.850	0.548*	0.843	0.505*	0.820
	VDKT	**0.551**	**0.854**	**0.559**	**0.848**	**0.514**	0.825
GBDT Ensemble	*Osika et al. [27]*	0.561	0.861	0.573	0.857	0.530	0.838
	SDKT	0.559*	0.860	0.564*	0.851	0.513*	0.826
	VDKT	**0.564**	**0.863**	**0.575**	**0.859**	**0.531**	**0.838**

Table 3: Evaluation results of our SDKT and VDKT on the Duolingo SLAM dataset compared with the state-of-the-art results by Osika et al. [27]. VDKT outperformed the state-of-the-art results in 8 out of 9 experiments and SDKT in all the 18 experiments. The best result for each metric is in bold. * indicates if the predicted results between VDKT and the model being compared have a statistically significant difference (p<0.05).

Section 3 can be applied to this synthesized example. To predict how well students perform, we need to uncover their understanding of the two underlying grammatical rules, and this can only be uncovered at a level of abstraction higher than raw score observations. In Figure 1, we used a two-dimensional latent variable as represented by belief bars to represent the student's understanding of two grammatical rules. In the actual VDKT architecture, we used a high-dimensional latent variable z that can capture substantially richer latent information, including but not limited to two grammatical rules.

Students also practiced the word *are* in the Duolingo SLAM dataset, the real language learning dataset. We present a real student's learning trajectory of the word *are* in the Duolingo SLAM dataset and visualize the advantage of VDKT over SDKT when predicting this student's performance in Figure 4. This student practiced six consecutive exercises containing the word *are* with three underlying linguistic rules: progressive forms, expletive constructions, and linking verbs. As Figure 4 shows, SDKT and VDKT performed equally well at predicting the student's performance on *are2*, *are3*, and *are4* when the associated rule was expletive constructions. However, the deterministic SDKT failed to capture the student's knowledge representation when the linguistic rule changed to progressive forms or linking verbs. VDKT, on the other hand, was still able to represent the knowledge state through probabilistic distributions and predicted the student's performance on *are1* and *are6* correctly.

```
are1: are you walking on the street (progressive form)
are2: these are books (expletive construction)
are3: these are cats (expletive construction)
are4: those are cups (expletive construction)
are5: are you in china (linking verb)
are6: the french are like them (linking verb)
```

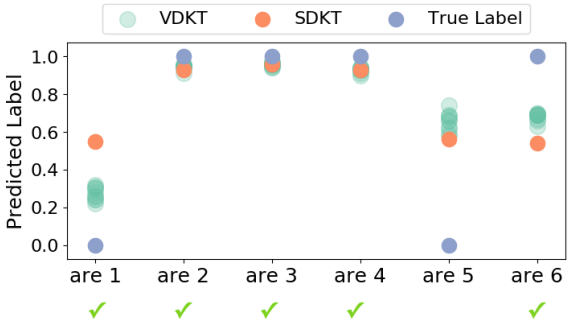

Figure 4: The student hKgzLBVC practiced the word *are* through six consecutive exercises covering three linguistic rules. Ground truth results and prediction results using SDKT and VDKT are shown in blue, orange, and green. VDKT results were randomly sampled ten times. ✓ indicates that the averaged VDKT result performs better than or as good as SDKT.

6.2 Gains of VDKT vs. Stochasticity in Data

We then proceed to examine students' stochastic behaviors in language learning. Although we cannot formally characterize how stochastic a dataset is, we approximate the slipping and the guessing events as follows: we define a student's incorrect answer to a word as a *slip* if the same student's immediately preceding and following answers to the same word are correct; likewise, a *guess* occurs if a student answers a word correctly while the immediately preceding and following answers are incorrect. The percentages of slipping and guessing events in the Duolingo SLAM test dataset for three languages are shown in Table 4. The distributions were similar for all the three languages: students were much more likely to make slips than guesses in language learning. We also computed the gains of VDKT over SDKT on F1 and AUC in Table 4. As can be seen, the general trend is that the more stochastic events a dataset has, which are characterized by percentages of guess and slip behaviors, the better performance VDKT achieves compared to SDKT.

Language	Size	F1 Gain	AUC Gain	Slip	Guess
English	1.05M	+1.47%	+0.51%	2.77%	0.60%
French	921K	+2.01%	+0.69%	3.01%	0.79%
Spanish	412K	+1.68%	+0.61%	2.74%	0.67%

Table 4: The performance gains of VDKT over SDKT on F1 and AUC versus the percentages of slips and guesses in three Duolingo SLAM sub-datasets. Size indicates the number of exercises in the combined train and test dataset for each of the three languages. The general trend is that higher stochasticity as characterized by percentages of slips and guesses leads to higher performance gains of VDKT compared to SDKT.

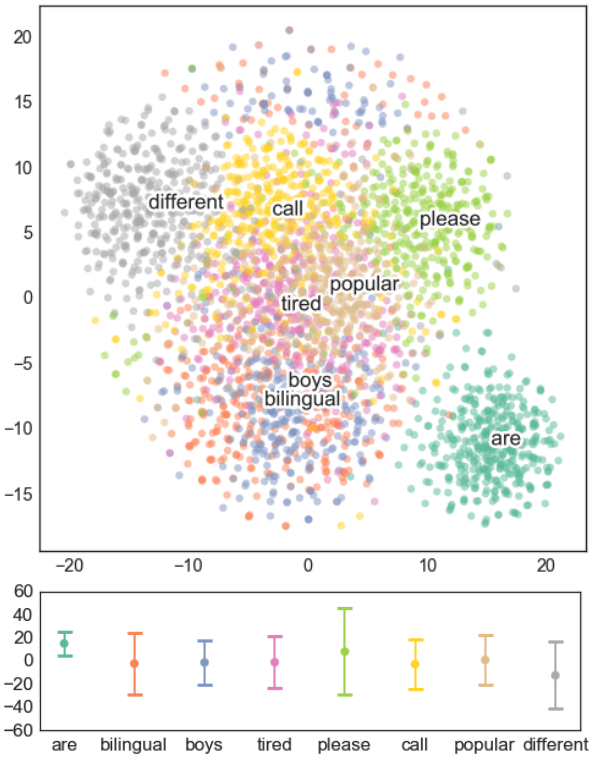

Figure 5: Visualization of latent variables generated from different words studied by the same student projected to 1D and 2D spaces.

6.3 Latent Space Projection

To understand the variation the latent variables capture, we visualized the contributions of words and students to the latent variables.

For vocabulary visualization, we randomly selected a user in the Duolingo SLAM dataset and sampled 400 latent variables from the distributions generated from each of the eight random words this student practiced. We then projected these latent variables into 1D and 2D spaces using t-SNE [24]. Figure 5 shows that simple and short words such as *are* have smaller variances, indicating that students have lower chances of making a guess or having a slip probably because they can memorize these words better. In contrast, words that are relatively more complicated and longer, e.g., *please* and *different*, have higher variances. This indicates that students might have higher chances of exhibiting stochastic behaviors when learning these words.

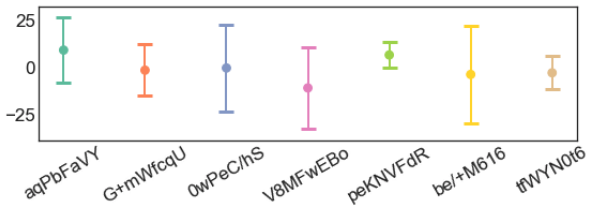

Figure 6: Visualization of latent variables generated from different students studying the same word projected to a 1D space. The X-axis lists student IDs.

The projected latent variables generated from different students studying the same word *are* are visualized in Figure 6. The model produces different variances for these students despite all learning the same word, which indicates that some students are more likely to have uncertain performance than others. This seems reasonable since student performance varies based on personal differences even when they study the same materials.

6.4 Error Analysis

Figure 7 shows the performance of SDKT and VDKT on a specific exercise in the SLAM dataset. We find that the predictions of SDKT stay conservatively in the middle while the predictions from VDKT are more spread out on the predictions, especially for words that have wrong labels. We analyzed the error rates of both SDKT and VDKT in Figure 8 for the Duolingo SLAM dataset. A comparison of the confusion matrices show that the class where VDKT outperformed SDKT the most is the wrong class, with a 4% difference.

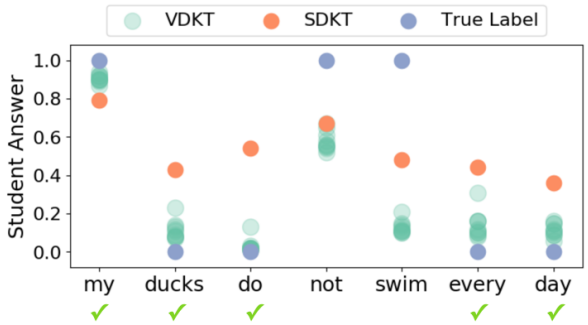

Figure 7: Visualization of predicted student answers using VDKT and SDKT versus ground truth labels. Ground truth results and prediction results using SDKT and VDKT are shown in blue, orange, and green. VDKT results were randomly sampled ten times. ✓ indicates that the averaged VDKT result performs better than or as good as SDKT.

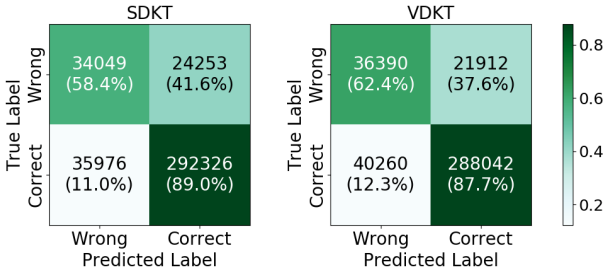

Figure 8: Confusion matrix of SDKT and VDKT on the SLAM English dataset.

6.5 Limitations and Future Work

Despite the promising results of modeling latent variables in learning, our work only studied one way of applying latent variational modeling to knowledge tracing, which is incorporating latent variables into a decoder. There might be other types of latent information in students' learning trajectories that need to be captured through different types of latent variable modeling, e.g., adding variations to the encoding phase [4] or to the LSTM states [6]. Future work can be developed under these directions to achieve even stronger modeling performance.

7 CONCLUSION

In this work, we examined the challenge of representing students' stochastic behaviors, which had long been overlooked in the deep knowledge tracing community. We addressed it by proposing Variational Deep Knowledge Tracing (VDKT), a latent variable deep knowledge tracing model that can model stochastic learning events. We evaluated VDKT on two large real-world language learning datasets and demonstrated its advantages over a deterministic baseline deep knowledge tracing model. To our knowledge, we are the first to systematically model latent variables in deep knowledge tracing, and we hope this work opens doors to future studies in the knowledge tracing field.

ACKNOWLEDGMENTS

We would like to thank Jia Li, Emma Brunskill, Chris Piech, Dan Jurafsky, and the Stanford NLP Group for providing valuable feedback throughout different stages of this work. We would also like to thank the reviewers for their thoughtful comments and revision suggestions, which helped us produce a better paper.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org.

[2] John R Anderson, C Franklin Boyle, Albert T Corbett, and Matthew W Lewis. 1990. Cognitive modeling and intelligent tutoring. *Artificial intelligence* 42, 1 (1990), 7–49.

[3] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics* 5 (2017), 135–146.

[4] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning* (Berlin, Germany). Association for Computational Linguistics, 10–21.

[5] Kris Cao and Stephen Clark. 2017. Latent Variable Dialogue Models and their Diversity. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Association for Computational Linguistics, Valencia, Spain, 182–187.

[6] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. 2015. A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*. 2980–2988.

[7] Albert T Corbett and John R Anderson. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction* 4, 4 (1994), 253–278.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B* 39, 1 (1977), 1–38.

[9] Jiachen Du, Wenjie Li, Yulan He, Ruifeng Xu, Lidong Bing, and Xuan Wang. 2018. Variational Autoregressive Decoder for Neural Response Generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 3154–3163.

[10] Christopher M Ely. 1989. Tolerance of ambiguity and use of second language strategies. *Foreign Language Annals* 22, 5 (1989), 437–445.

[11] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[12] Zoubin Ghahramani. 2001. *An Introduction to Hidden Markov Models and Bayesian Networks*. World Scientific Publishing Co., Inc., USA, 9–42.

[13] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

[14] Andrew Heathcote, Scott Brown, and Douglas JK Mewhort. 2000. The power law repealed: The case for an exponential law of practice. *Psychonomic bulletin & review* 7, 2 (2000), 185–207.

[15] Geoffrey E Hinton and Richard S Zemel. 1994. Autoencoders, minimum description length, and Helmholtz free energy. *Advances in neural information processing systems* 6 (1994), 3–10.

[16] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[17] Masahiro Kaneko, Tomoyuki Kajiwara, and Mamoru Komachi. 2018. TMU System for SLAM-2018. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, New Orleans, Louisiana, 365–369.

[18] Mohammad Khajah, Robert V Lindsey, and Michael C Mozer. 2016. How deep is knowledge tracing? *Educational Data Mining* (2016).

[19] Mohammad Khajah, Rowan Wing, Robert Lindsey, and Michael Mozer. 2014. Integrating latent-factor and knowledge-tracing models to predict individual differences in learning. In *Educational Data Mining 2014*.

[20] Diederik P Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*.

[21] Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics* 22, 1 (1951), 79–86.

[22] Sebastian Leitner. 1995. *So lernt man lernen: angewandte Lernpsychologie–ein Weg zum Erfolg*. Herder.

[23] Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354* (2016).

[24] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[25] Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika* 12, 2 (1947), 153–157.

[26] Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural Variational Inference for Text Processing. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48* (New York, NY, USA) *(ICML'16)*. JMLR.org, 1727–1736.

[27] Anton Osika, Susanna Nilsson, Andrii Sydorchuk, Faruk Sahin, and Anders Huss. 2018. Second Language Acquisition Modeling: An Ensemble Approach. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, New Orleans, Louisiana, 217–222.

[28] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. 2015. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*. 505–513.

[29] Paul Pimsleur. 1967. A memory schedule. *The Modern Language Journal* 51, 2 (1967), 73–75.

[30] Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. 2016. Variational Autoencoder for Deep Learning of Images, Labels and Captions. In *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (Eds.), Vol. 29. Curran Associates, Inc., 2352–2360.

[31] Adithya Renduchintala, Philipp Koehn, and Jason Eisner. 2017. Knowledge Tracing in Sequential Learning of Inflected Vocabulary. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*. 238–247.

[32] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31st International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 32)*, Eric P. Xing and Tony Jebara (Eds.). PMLR, Bejing, China, 1278–1286.

[33] Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. 2018. A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, Stockholmsmässan, Stockholm Sweden, 4364–4373.

[34] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning representations by back-propagating errors. *Nature* 323, 6088 (1986), 533–536.

[35] Jürgen Schmidhuber. 2015. Deep learning in neural networks: An overview. *Neural networks* 61 (2015), 85–117.

[36] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. A Hierarchical Latent Variable Encoder-decoder Model for Generating Dialogues. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence* (San Francisco, California, USA) *(AAAI'17)*. AAAI Press, 3295–3301.

[37] B. Settles, C. Brust, E. Gustafson, M. Hagiwara, and N. Madnani. 2018. Second Language Acquisition Modeling. In *Proceedings of the NAACL-HLT Workshop on Innovative Use of NLP for Building Educational Applications (BEA)*. ACL.

[38] B. Settles and B. Meeder. 2016. A Trainable Spaced Repetition Model for Language Learning. In *Proceedings of the Association for Computational Linguistics (ACL)*. ACL, 1848–1858.

[39] Yu Su, Qingwen Liu, Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Chris HQ Ding, Si Wei, and Guoping Hu. 2018. Exercise-Enhanced Sequential Modeling for Student Performance Prediction. In *AAAI*.

[40] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[41] TensorFlow. 2021. Word embeddings. https://www.tensorflow.org/tutorials/text/word_embeddings. Accessed: 2021-01-27.

[42] Lisa Wang, Angela Sy, Larry Liu, and Chris Piech. 2017. Deep Knowledge Tracing On Programming Exercises. In *Proceedings of the Fourth (2017) ACM Conference on Learning @ Scale* (Cambridge, Massachusetts, USA) *(L@S '17)*. ACM, New York, NY, USA, 201–204.

[43] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR* abs/1609.08144 (2016).

[44] Xiaolu Xiong, Siyuan Zhao, Eric Van Inwegen, and Joseph Beck. 2016. Going Deeper with Deep Knowledge Tracing. In *EDM*. 545–550.

[45] Shuyao Xu, Jin Chen, and Long Qin. 2018. CLUF: a Neural Model for Second Language Acquisition Modeling. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*. 374–380.

[46] Ahmed Zaidi, Andrew Caines, Russell Moore, Paula Buttery, and Andrew Rice. 2020. Adaptive Forgetting Curves for Spaced Repetition Language Learning. In *Artificial Intelligence in Education*, Ig Ibert Bittencourt, Mutlu Cukurova, Kasia Muldner, Rose Luckin, and Eva Millán (Eds.). Springer International Publishing, Cham, 358–363.

[47] Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. 2017. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 765–774.

[48] Liang Zhang, Xiaolu Xiong, Siyuan Zhao, Anthony Botelho, and Neil T Heffernan. 2017. Incorporating rich features into deep knowledge tracing. In *Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale*. ACM, 169–172.