Imputing Knowledge Tracing Data with Subject-Based Training via LSTM Variational Autoencoders Frameworks

Jia Tracy Shen **Dongwon Lee**

The Pennsylvania State University, USA
{jqs5443, dongwon}@psu.edu

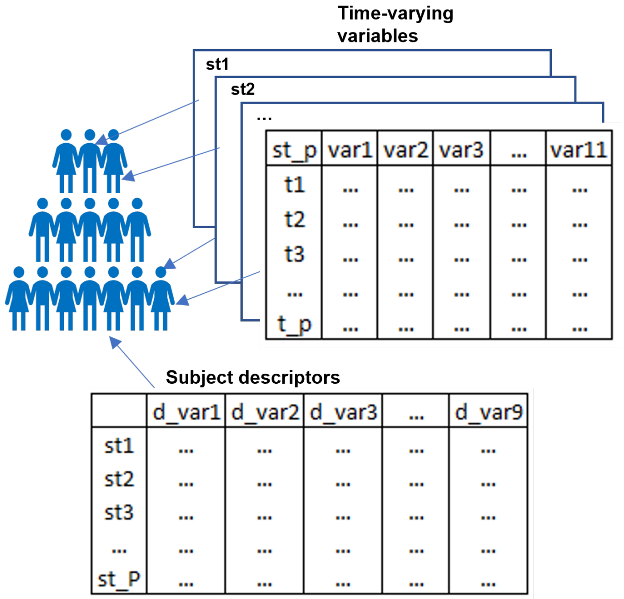
Abstract

The issue of missing data poses a great challenge on boosting performance and application of deep learning models in the *Knowledge Tracing* (KT) problem. However, there has been the lack of understanding on the issue in the literature. In this work, to address this challenge, we adopt a subject-based training method to split and impute data by student IDs instead of row number splitting which we call non-subject based training. The benefit of subject-based training can retain the complete sequence for each student and hence achieve efficient training. Further, we leverage two existing deep generative frameworks, namely variational Autoencoders (VAE) and Longitudinal Variational Autoencoders (LVAE) frameworks and build LSTM kernels into them to form LSTM-VAE and LSTM LVAE (noted as VAE and LVAE for simplicity) models to generate quality data. In LVAE, a Gaussian Process (GP) model is trained to disentangle the correlation between the subject (i.e., student) descriptor information (e.g., age, gender) and the latent space. The paper finally compare the model performance between training the original data and training the data imputed with generated data from non-subject based model VAE-NS and subject-based training models (i.e., VAE and LVAE). We demonstrate that the generated data from LSTM-VAE and LSTM-LVAE can boost the original model performance by about 50%. Moreover, the original model just needs 10% more student data to surpass the original performance if the prediction model is small and 50% more data if the prediction model is large with our proposed frameworks.

1 Introduction

Knowledge tracing (KT) as a student modeling technique has been widely used to predict and trace students' knowledge state during their learning processes. In recent years, with the huge success that deep learning has brought to the field, there are many KT algorithms that can predict individuals' knowledge state to a decent extent. However, the *sparseness* of students' exercise data represented by *missing values* still limits the models' performance and application (Swamy et al. 2018). About half of the existing publications use public data sets (Dai et al. 2021), which can not be available for huge amount due to administration cost. Researchers could opt for other private data sets that however may not even have the sizable volume as the public data sets. Besides, many deep learning algorithms including the state-of-art (SOTA) KT algorithms need huge and diverse amount of training data to obtain decent performances. On the other hand, it is unavoidable to see the missing values in KT data because of two reasons: (i) data is missed completely at random (MCAR) where the probability of missing data is independent on its own value and on other observable values (Roderick J. A. Little 2002). For example, due to COVID, we have many students missing exams; (ii) the data is missed not at random (MNAR), which indicates the reason for a missing value can depend on other variables but also on the value that is missing. For example, if a student performs poorly on the English subject and often miss exams in other subjects, his missed records in English quizzes can be attributed to other known reasons. Moreover, KT data is a type of longitudinal data, all collected repeatedly over time for each *subject* (ie., student). Such data contains both dependent and independent variables. For example, the dependent variables in KT data can comprise time-varying measurements per *subject* (e.g., response correctness, time taken per question), whereas independent variables are time-invariant *subject descriptors* (e.g., grade, gender, gifted or not) (see the illustration in Figure 1). Analyzing such data is challenging as it often includes high-dimensional time [in]variant variables with missing values. Despite that missing data in KT field is ubiquitous and poses challenges on achieving better model results, there are very few studies researching on effective approaches to tackle the missing data issue in KT field. Our work is one of the few studies to address such challenge.

To that end, we suppose a deep generative model such as Variational Autoencoders (VAE) (Kingma and Welling 2019) could effectively generate data for the missing values because of its superiority over other generative models (e.g., Generative Adversarial Networks) in time series data generation (Le, Wang, and Lee 2020; Fährmann et al. 2022). Furthermore, given the challenge arisen from the longitudinal KT data, we make two hypotheses: (i) a training style that can reflect the subject longitudinal nature could entail more effective training; (ii) the information from subject descriptors could potentially represent the latent space better and help improve the quality on the data generation. To vali-



st_p	var1	var2	var3	...	var11
t1	...	...	...	...	...
t2	...	...	...	...	...
t3	...	...	...	...	...
...	...	...	...	...	...
t_p	...	...	...	...	...

	d_var1	d_var2	d_var3	...	d_var9
st1	...	...	...	...	...
st2	...	...	...	...	...
st3	...	...	...	...	...
...	...	...	...	...	...
st_P	...	...	...	...	...

Figure 1: An Illustration of the Longitudinal (student) Data in KT Field. ‘p’: student p. ‘P’: total # of students.

date hypothesis (i), we develop a subject-based training style where we split and impute data by student IDs to reflect the longitudinal nature of the subjects. The benefit of doing so is to maintain the complete sequence for each student whereas splitting by row number could separate the individual sequence and entail inefficient training. Thus, applying subject-based training on top of VAE framework could potentially address the challenge. To validate hypothesis (ii), we leverage a module from the existing Longitudinal VAE (LVAE) (Ramchandran et al. 2021) framework called additive multi-output Gaussian Process (GP) prior that can extrapolate the correlation between time-invariant subject descriptors and the latent space to enhance the latent variable learning. Given the longitudinal nature of the LVAE framework, a subject-based training can be naturally applied to LVAE to boost data generation quality. Furthermore, we build LSTM kernels to both VAE and LVAE frameworks because LSTMs are good at extrapolating the temporal relationship from multi-variate time series data (Pearlmutter 1989; Giles, Kuhn, and Williams 1994). With the generated data from the proposed frameworks, we will be able to impute them back for retraining and evaluate the effectiveness of the imputed data on boosting the original model performance. Besides, we are also interested to discover how robust our generated data can be on boosting the original model performance, e.g., by only applying a fraction of the generated data.

Thus, this work attempts to make the following contributions:

- Overcoming the issue of the missing KT data, we conduct *subject-based* training on KT data via LSTM-VAE framework
- Leveraging the additive GP prior module from LVAE, we form a LSTM-LVAE framework to showcase the superiority of training additional *subject descriptors* for better latent space representation
- We demonstrate the robustness of only using a fraction of the generated data to boost the original model performance

2 Method

We propose two deep generative frameworks: LSTM-VAE and LSTM-LVAE. The both frameworks use subject-based training. We explain the details as follows.

2.1 Problem setting

According to Ramchandran et al. 2021, let D be the dimensionality of the observed data, P be the number of unique students, n_p be the total number of longitudinal samples from student p, and $N = \sum_{p=1}^{P} n_p$ be the total number of samples. Therefore, the longitudinal samples for student p are denoted as $Y_p = [y_1^p, ..., y_{n_p}^p]^T$, where each sample $y_t^p \in \mathcal{Y}$ and $\mathcal{Y} = \mathbb{R}^D$. The subject descriptors for students are represented as $X_p = [x_1^p, ..., x_{n_p}^p]^T$, where $x_t^p \in \mathcal{X}$ and $\mathcal{X} = \mathbb{R}^Q$, Q be the number of descriptors. The latent space is then denoted as $\mathcal{Z} = \mathbb{R}^L$ and a latent embedding for all N samples as $Z = [z_1, ..., z_N]^T \in \mathbb{R}^{N \times L}$ with L being the number of latent dimensions. To generate data, a joint generative model is then parameterized by $w = \{\psi, \theta\}$ as $p_w(y, z) = p_\psi(y|z)p_\theta(z)$. Therefore, if the latent variable z is known, it will be easy to infer y and hence generate the desired data.

2.2 VAE and LVAE

To infer the latent variable z given y, the posterior distribution is $p_w(z|y) = p_\psi(y|z)p_\theta(z)/p_w(y)$ and is generally intractable due to the marginalization over the latent space $p_w(y) = \int p_\psi(y|z)p_\theta(z)dz$. Therefore, Variational Auto-Encoder (Kingma and Welling 2019) introduced the approximated version posterior, noted as $q_\phi(z|y)$ instead of the true posterior $p_w(z|y)$ and fit the approximate inference model by maximizing the Evidence Lower Bound (ELBO) of the marginal log-likelihood w.r.t. ϕ:

$$\log p_w(Y) \geq \mathcal{L}(\phi, \psi, \theta; Y)$$

$$\triangleq \mathbb{E}_{q_\phi}[\log p_\psi(Y|Z)] - D_{KL}(q_\phi(Z|Y)||p_\theta(Z)) \to \max_{\phi},$$

where $\mathbb{E}_{q_\phi}[\log p_\psi(Y|Z)]$ is a reconstruction error, measuring the difference between the input and the encoded-decoded data. and D_{KL} denotes the Kullback-Leibler Divergence (KLD), measuring the divergence between $q_\phi(Z|Y)$ and $p_\theta(Z)$. In practice, we minimize the negative ELBO: $D_{KL}(q_\phi(Z|Y)||p_\theta(Z) - \mathbb{E}_{q_\phi}[\log p_\psi(Y|Z)]$, where all the parameters are learned simultaneously together: $\mathcal{L}(\phi, \psi, \theta; Y) \to \min_{\phi,\psi,\theta}$.

When facing the longitudinal data, Ramchandran et al. hypothesize z has relationship with both Y and X and formulate the generative model as

$$p_w(Y|X) = \int_Z p_\psi(Y|Z, X)p_\theta(Z|X)dZ$$

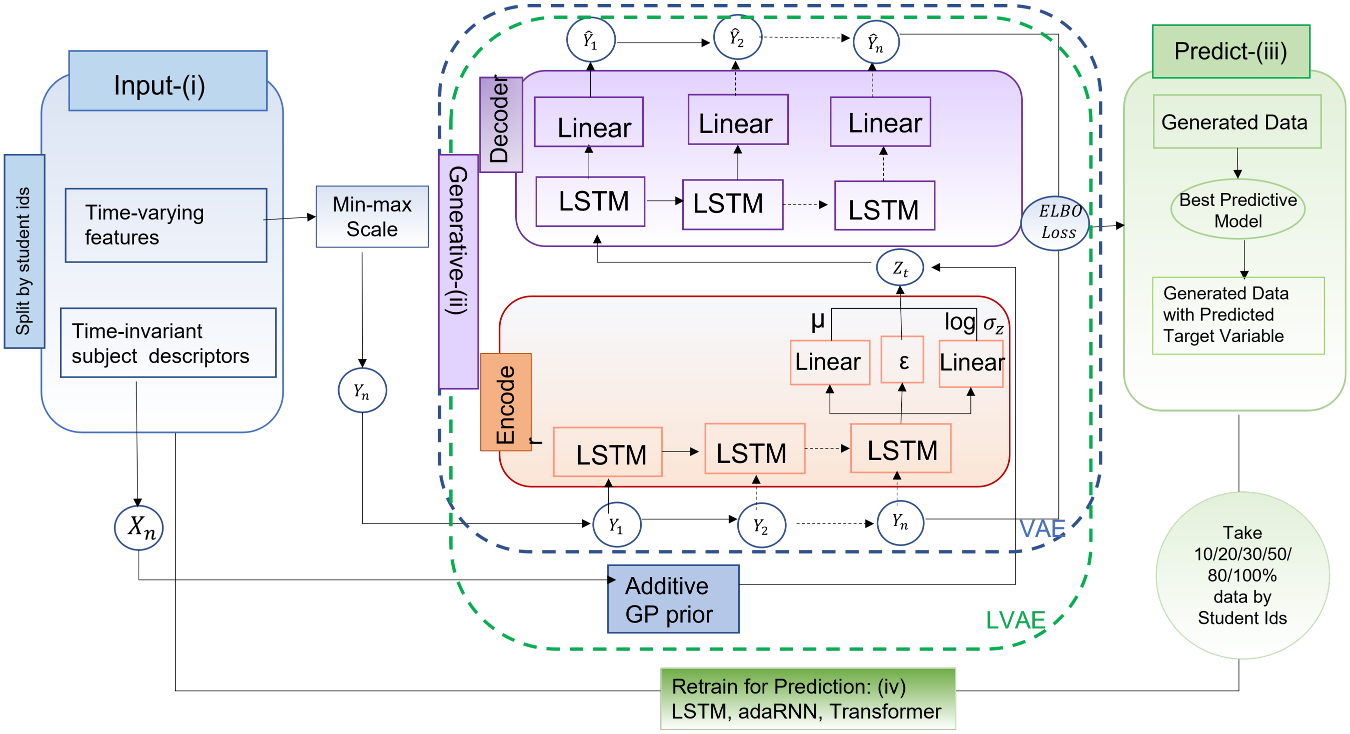


Figure 2: Overview of the methodology proposed in this work.

$$= \int_Z \prod p_\psi(y_n|z_n) p_\theta(Z|X) dZ,$$

where $p_\psi(y_n|z_n)$ is normally distributed probabilistic decoder and $p_\theta(Z|X)$ is defined by the multi-output additive GP prior that regulates the joint structure of Z with descriptors X. The Additive GP is a Gaussian process prior as $f(x) \sim GP(\mu(x), K(x, x'|\theta))$, where $\mu(x) \in \mathbb{R}_{\mathbb{L}}$ is the mean (assumed as 0) and $K(x, x'|\theta)$ is a matrix-valued positive definite cross-covariance function (CCF). Based on the practice of Cheng et al., LVAE constructs the additive GP components with squared exponential CFs (from continuous variables), categorical CFs (from categorical covariates),the interaction CFs (the product of the categorical and squared exponential CFs) and the product of the squared exponential CFs and the binary CFs. Finally, the ELBO function changes to the following after factoring the descriptors X:

$$\log p_w(Y|X) \geq \mathcal{L}(\phi, \psi, \theta; Y, X)$$

$$\triangleq \mathbb{E}_{q_\phi}[\log p_\psi(Y|Z)] - D_{KL}(q_\phi(Z|Y)||p_\theta(Z|X) \rightarrow \max_\phi.$$

LVAE differentiates from VAE in that it hypothesizes there exists a relationship between $\mathcal{X}$ and the latent space $\mathcal{Z}$ and uses an additive multi-output Gaussian Prior to extract that relationship.

2.3 Generative Frameworks

Based on above solutions, two generative frameworks are developed (see in Figure 2). It has 4 phases: (i) input phase that pre-processes data; (ii) generative phase where data gets generated via the generative model framework; (iii) prediction phase where we predict target variable for the generated data; (iv) retraining phase, where we combine the original data and generated data to retrain for donwstream prediction task.

From the figure, after input phase (i), we see that the data gets separated into two sets: (a) time-varying data (noted as $Y = \{y_1, ..., y_n\}$); (b) time-invariant subject descriptors (noted as $X = \{x_1, ..., x_n\}$). The time-varying data y_n goes through a min-max scaler, a typical time series data normalization method (Yu et al. 2021), and enters the LSTM encoder to generate μ and log σ_z for the latent distribution Z_t. The time-invariant subject descriptors X_n on the other hand are only fed into the Additive GP prior module to train for the approximated GP prior with its output merging into the latent space Z_t. Next, the decoder samples on the latent distribution and reconstructs data $\hat{Y}_n$, namely encoded-decoded data, based on the latent features from Z_t. Here, we name the generative framework that only includes the encoder and decoder as LSTM-VAE and the framework that includes encoder, decoder and the additive GP prior module as LSTM-LVAE. We omit LSTM prefix for simplicity. After that, we compare Y_n to $\hat{Y}_n$ for evaluation via ELBO. VAE assigns the equal weight for both reconstruction and KLD errors whereas LVAE assigns a weight to KLD to regularize further. Once we have good generation quality, we generate data on the missing data which has all the subject descriptor information but missing on all the time-varying features.

Before entering phase (iii), we conduct initial prediction on the original data via models that work well with multi-

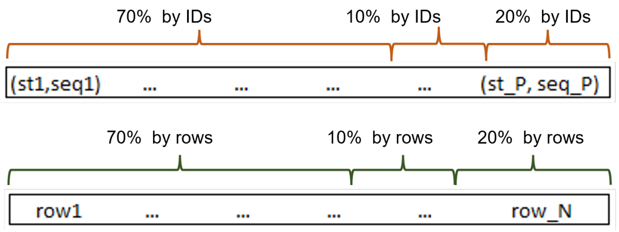

70% by IDs
10% by IDs
20% by IDs
(st1,seq1) ... ... (st_P, seq_P)
70% by rows
10% by rows
20% by rows
row1 ... ... row_N

Figure 3: An Illustration of Split by IDs vs. Row Number

variate time series data: LSTM, adaRNN and Transformer. Similar to LSTM, adaRNN (i.e., adaptive RNN) (Du et al. 2021) is a recurrent neural network but based on the Gated Recurrent Unit that comprises two gates (i.e., reset gate and update gate). It usually trains faster than LSTM and easy to modify and works better if the sequence is not too long. Because some KT data could present non-sequential characteristics, we include the original Transformer model (Vaswani et al. 2017), whose attention mechanism and positional embedding are great for non-sequential data. To evaluate these models, we use Root Mean Square Error (RMSE) as our target variable is continuous (i.e., score rate, the possible score obtained per question divides the total scores obtained per student). After the initial round of prediction is performed, we conduct phase (iii) by selecting the best predicting model to predict the target variable for the generated data from phase (ii). In phase (iv), we impute the fraction of 10/20/30/50/80/100% of the generated data (with target variable) back to the original data and retrain for the downstream predictive task.

2.4 Subject-based Training

Besides the generative frameworks, this work also takes a new training strategy, that is, subject-based training. We refer subject-based training to a style where data are split and imputed back by student IDs instead of row number. We call the training using row-number splitting as non-subject based training. For example, in subject-based setting, 70% of student IDs are extracted as training data and 10% student IDs are extracted as validation data whereas in non-subject based setting, 70% of total rows are extracted as training data and 10% total rows are extracted as validation data) (see the illustration in Figure 3). We see the split points by IDs are not the same as splitting by row number. It indicates there is chance that the sequence of certain students will be cut into two pieces, leaving them into two different sets (e.g, val and test). If we split the data by student IDs, we can impute the generated data back to the original data via IDs and keep the learning sequence relevant and complete for each student. If we opt for row-number splitting, the student's original sequence will be interfered and not be trained appropriately.

3 Experiments

In this section, we carry out two major experiments. The first experiment is to generate the data and impute back to the original data for retraining. It has three steps: (a) generate knowledge tracing data by utilizing VAE and LVAE; (b) predict the target variable for the generated data using the best model obtained from the KT prediction task; (c) merge the generated data with the original data to retrain for model performance. The second experiment is to validate the robustness of imputed data on boosting the original model performance. More specifically, we add a fraction of the generated data in the cadence of 10%, 20%, 30%, 50%, 80%, 100% during the retraining phase to examine the boosting effect.

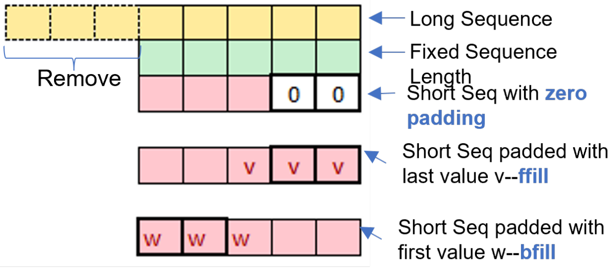

Long Sequence
Fixed Sequence Length
Remove
0
0
Short Seq with zero padding
v
v
v
Short Seq padded with last value v--ffill
w
w
w
Short Seq padded with first value w--bfill

Figure 4: An illustration of KT data sequence aligning process and 3 Padding Strategy.

3.1 Data sets

To achieve above, we need to apply our model onto the data sets that have subject descriptors so that we can use LSTM-LVAE model to generate missing data. Unfortunately, the public data sets (e.g., ASSISTment datasets, Junyi, STATICS, EdNet, etc) in KT field do not contain subject descriptor information such as the student's grade level, gifted or not. This is also why the renowned deep learning models such as DKT, DKVMN, NPA, SAINTS (Minn et al. 2018; Zhang et al. 2017; Pandey and Karypis 2019; Shin et al. 2021) are not included in the chapter because most of these models are generated for single variable KT data or take data feature as hyper-parameters. Thus, we use the two private multivariate KT data sets from K12.com platform (an online K-Grade 12 education platform). They are : (1) Grade 10 geometry course (noted as Geom) quiz answering data set with average sequence length of 150 time steps; (2) Grade 11 algebra II (noted as Alg2) quiz answering data set with average sequence length of 150 time steps. Each data set contains 11 temporal features (i.e. sequence number, assessment duration, attempts per question, total attempts, question difficulty, item difficulty, standard difficulty, question reference, item reference, standard id, question type) from July 2017 to June 2019 and 7 subject descriptors that define the student profiles (i.e., school ID, special ED, student id, free reduced lunch, gifted_talented, grade level, score rate). The Geom data set contains 3,265 total students with 412,397 observed instances whereas the Alg2 data has 2,110 total students with 277,548 observed instances.

3.2 Identify Missing Values

In practice, it is hard to identify the missing steps each student has because their learning experience varies. Thus, we develop a regime where we first find all the quiz times of a school where the student is located and then fill up the missing times by comparing to the school's full quiz taking schedule. For example, if school A has 100 quiz times but student A only has 60 records, we fill out the remaining 40 quiz time steps based on the event time variable. This approach is a bit rigorous, assuming all the students are required to test for the same number of quizzes if they are in the same school and skipping any quiz is considered as a missing step. In reality, there might be scenarios where students are allowed to skip, which is complex to study and hence we use this approach as it is straightforward. With that, we are able to retrieve the missing time steps before and after the current temporal steps for all the students. As the students are known, this missing data has all the subject descriptor information.

3.3 Data Processing

To conduct the training for generation, we split the data by 0.5/0.1/0.2/0.2 for train/val/test/generate and 0.7/0.1/0.2 for train/val/test during downstream prediction (see in Table 1). Note that the generation set with a ratio of 0.2 is used to evaluate the quality of generation whereas the generated set we use to impute back to the original data is generated from missing data. Based on the above missing data identification regime, we are able to identify 3,233 out of 3,265 total students who have a total of 799,408 missed instances from the Geom course and 2,057 out of 2,110 students who have a total of 516,884 missed instances from the Alg2 course (see in Table 1). Because all the ratios are applied to both subject and non-subject based training, the generated data from missing values will be imputed back to the original data via IDs in the subject-based training and via row number in the non-subject based training in the splits of train/val/test. Both training styles align data to a fixed sequence length which is due to the model input requirement of 3D dimensions (i.e., batch size * sequence length * number of dimensions). This also aligns with the typical data processing technique for KT model training (Pardos and Heffernan 2011; Lee et al. 2019; Pandey and Karypis 2019). If the actual student learning sequence is longer than the fixed sequence length, we cut the part where it exceeds. If the sequence is shorter than the fixed sequence, we pad it (see in Figure 4). We use three padding strategies to find an optimal model performance: (a) zero paddding; (b) ffill; (c) bfill. Ffill pads forward with the last value 'v' whereas bfill pads backward with the first value 'w' (see in Figure 4). Bfill in practice assumes that a student gets the same quiz result in his missed quiz as his first quiz result whereas ffill assumes that a student gets the same quiz result in his missed quiz as his last quiz result. Zero-padding just simply assumes that a student gets zero in his missing quiz.

3.4 Generation and Imputation

We train three generative models: VAE-NS (non-subject), VAE (subject-based) and LVAE (subject-based) to generate missing data. Since LVAE is only possible to train if we have descriptor information, which relies on student ID information, we do not apply non-subject training for LVAE. After data is generated for all the missing data, we impute back the generated data from VAE-NS by the row-number splits and impute the generated data from VAE and LVAE by ID splits (see in Figure 5). We do not only impute back the generated data to the train set because we believe the data augmentation on all the train, val and test sets will make the model performance harder to improve than we only augment the train set but leave the test set the same.

Table 1: Data Statistics for Geom and Alg2 Data. * is Downstream Task Split

Split Part (Ratio)	Geometry (Geom)		Algebra II (Alg2)	
	# Student	# of Rows	# Student	# of Rows
Train (0.5)	1,633	215,632	1,055	137,409
Validate (0.1)	326	42,259	211	30,652
Test (0.2)	653	82,707	422	60,709
Generation (0.2)	653	71,799	422	48,778
Data Total	3,265	412,397	2,110	277,548
Train* (0.7)	2,286	287,431	1,477	186,187
Validate* (0.1)	326	42,259	211	30,652
Test* (0.2)	653	71,799	422	60,709
Data Total*	3,265	412,397	2,110	277,548
Missing Train (0.7)	2,256	559,586	1,440	361,819
Missing Validate (0.1)	322	79,941	206	51,688
Missing Test (0.2)	645	159,882	411	103,377
Missing Data Total	3,223	799,408	2,057	516,884

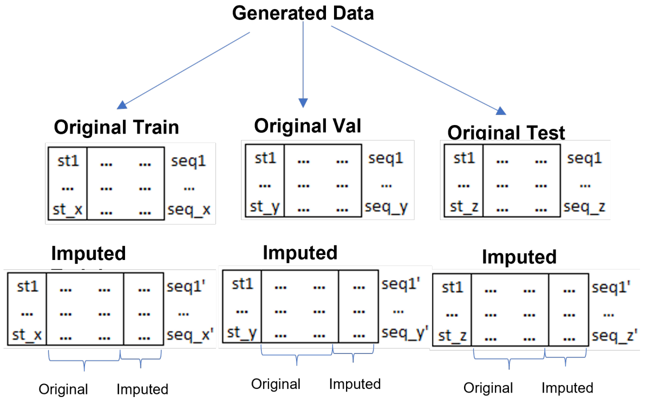


Figure 5: An Illustration of the Data Imputation Process.

3.5 Downstream Prediction

There are two rounds of downstream predictions. The initial round is conducted on the original data using the three padding strategies to find the best performance model so that we can use it to predict the target variable for the generate data. The second round is a retraining round where we impute back the generated data using the best padding strategy. The second round has two parts: (i) we conduct the retraining on the combined data that contains all the generated data and the original data by IDs (for VAE, LVAE) and by row number (for VAE-NS); (ii) we conduct retraining on the combined data with a fraction (i.e., 10/20/30/50/80/100%) of the generated data and the original data only by IDs (for VAE, LVAE) because VAE-NS does not show salient improvement with the data it generates.

Table 2: Average RMSE by Padding Strategy, Models and Data sets. The boldface represents the best performance.

Avg. RMSE	Geometry (Geom)			Algebra II (Alg2)		
	adaRNN	LSTM	Transformer	adaRNN	LSTM	Transformer
Bfill	0.50734	0.47665	0.48613	0.52034	**0.48967**	0.49463
Ffill	0.51946	**0.47664**	0.49713	0.51895	0.48995	0.49632
Zero	**0.48160**	0.47702	**0.40208**	**0.48860**	0.49173	**0.45138**

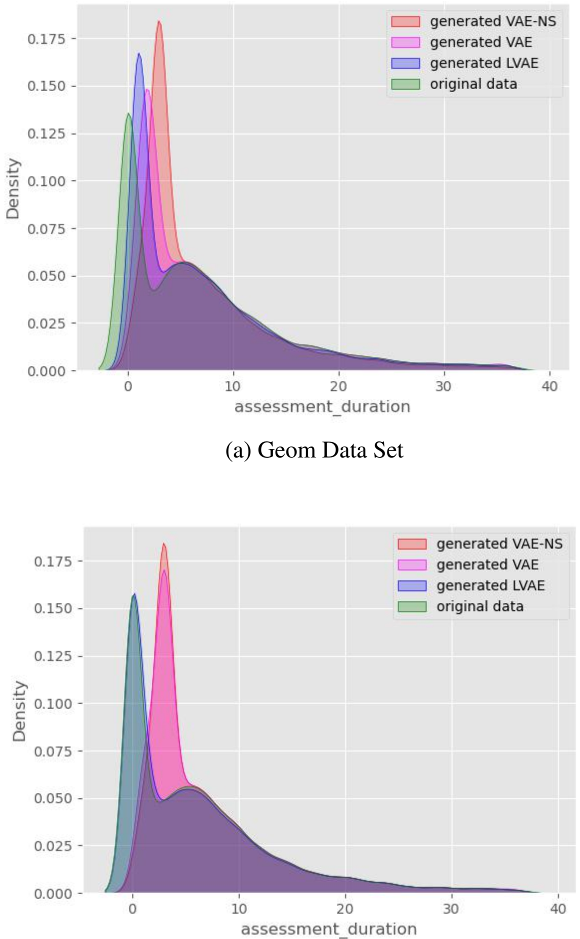
Table 3: Average RMSE by Generative Models for Prediction Tasks. The bold face represents the best performance.

Avg. RMSE	Geometry (Geom)			Algebra II (Alg2)		
	adaRNN	LSTM	Transformer	adaRNN	LSTM	Transformer
Original	0.48160	0.47664	0.40208	0.48860	0.49173	0.45138
VAE-NS	0.58251	0.48030	0.37090	0.49388	0.48989	0.34071
VAE	0.26570	0.26559	0.32902	0.30304	**0.27293**	**0.35260**
LVAE	**0.26226**	**0.26185**	**0.28913**	**0.29470**	0.27326	0.35911

4 Results

4.1 Evaluating the Quality of the Generated Data

We exhaustively train three generative models (i.e., VAE-NS, VAE and LVAE) until its loss stops improving with different sets of hyper-parameter tuning to reach the best result. We observe that VAE-NS model is hard to converge and stops early with final loss of around 13.4349 for Alg2 data set and 0.6282 for geom data set. VAE is able to decrease loss to 0.2652 for Alg2 data set and 0.2661 for geom data set. LVAE however can decrease loss to 0.2291 for alg2 data and 0.1863 for geom data set with the latent dimension as 64 and hidden dimension as 128. Figure 6 selects the 'assessment_duration' feature to compare the data distribution between original data and generated data by VAE-NS, VAE and LVAE. We can tell that the Geom generated data for the feature 'assessment_duration' from VAE-NS sways the farthest from the original data whereas the generated data from VAE and LVAE are closer to the original data distribution with LVAE slightly better. The same case applies to the Alg2 data. The plot also shows that generally both VAE and LVAE can reconstruct data closely to the original data distribution, indicating that we are safe to use such generated data for downstream prediction tasks.



(a) Geom Data Set

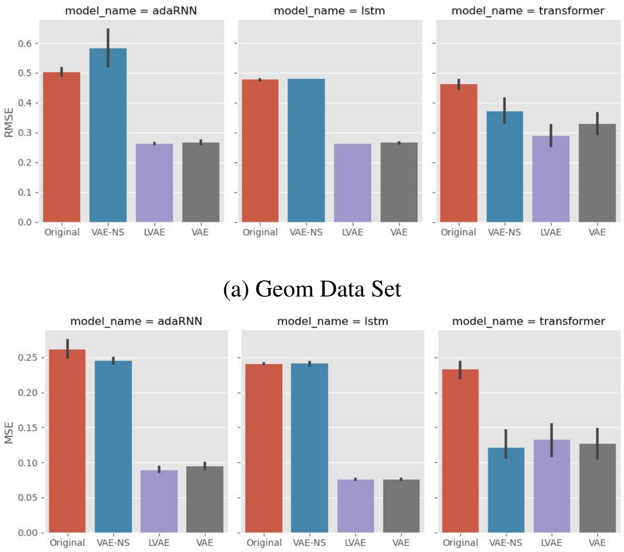
(b) Alg2 Data Set

Figure 6: 'Assessment Duration' Feature Distribution Comparison Between Original Data and Generated Data

4.2 Evaluating the Effectiveness of the Imputed Data

Before we impute the generated data, we conduct the first round of downstream prediction via three models (i.e., LSTM, adaRNN and Transformer) by three padding strategies (i.e., bfill, ffill and zero padding) to select the best model performance as the baseline original data model performance. We run 10 random seeds for each model, padding strategy and data set. Table 2 shows the detailed average RMSE for each model by padding schemes. We see that adaRNN and Transformer model obtain the best performances when padded with zero whereas LSTM model obtains its best performance via ffill padding for Geom data and bfill for Alg2 data. We then use the best performing model to predict target variable for the generated data and impute back the generated data (with the target variable) to the original data set for retraining.

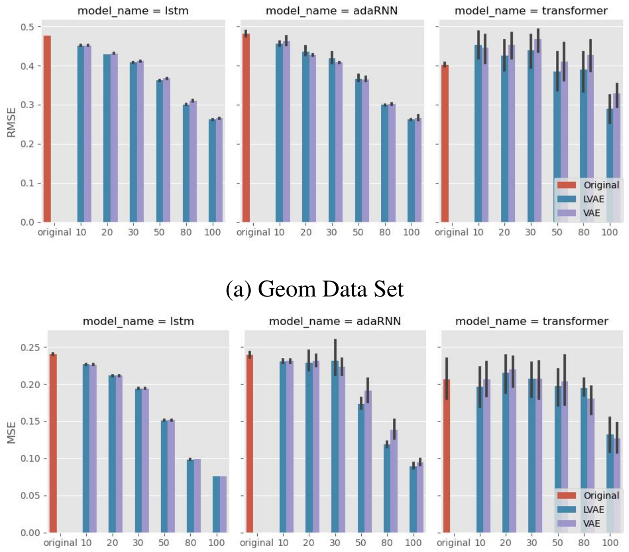
We observe the retrained model performance surpasses the original model performance by big margins (see in Table 3). From the table on column 1 under Geom data, we observe the retrained adaRNN model performance using the generated data from VAE is 0.26570, almost about 50% lower than the original model performance of 0.48160 in RMSE. Oppositely, the retrained model performance using the generated data from VAE-NS has RMSE of 0.58251, which is higher than the original model RMSE. This might indicate the generated data from non-subject based training perturbs the original data and creates negative gain. Further, we notice the model performance of using generated data from LVAE is even slightly better than VAE with a lower average RMSE of 0.26226. This phenomenon is present across the three models for Geom data. For Alg2 data, we also observe superior performance from both VAE and LVAE. However, the retrained model using generated data from VAE seems to perform slightly better than the one with LVAE generated data. Figure 7 visually presents the sharp drop of the average



(a) Geom Data Set

(b) Alg2 Data Set

Figure 7: Average Retraining RMSE Using Generated Data From Different Models. The error bar shows the min. and max. of the 10 random seeds.



(a) Geom Data Set

(b) Alg2 Data Set

Figure 8: Average RMSE by % of Imputed Data vs. Original Data.

RMSE after imputing the generated data from both VAE and LVAE models.

4.3 Evaluating the Robustness of the Imputed Data

Once we learn that imputed data can boost original model performance to a significant extent, we further experiment to validate the robustness of the imputed data. More specifically, we impute the number of students in the fraction of 10%, 20%, 30%, 50%, 80%, 100% back to their original train/val/test sets. The choice of percentage increments in number of students are arbitrary but all the students are linked back via their IDs to the original train/val/test sets. It is designed this way so that it is harder for the retrained model to outperform the original model as the number of students in the train/val/test set are still the same but with longer sequences. Figure 8 showcases the effectiveness of adding different fractions of students to boost the original model performance. For LSTM and adaRNN model, we observe that the model performance starts to boost after imputing only 10% of student IDs back. As the percentage gets higher, we see higher boosting. For Transformer model, it starts to boost after imputing 50% of student IDs back. This confirms a known fact that large models such as Transformer model needs more data to boost its performance. In general, imputing data based on the subjects can boost the model to a great extent.

5 Summary

In conclusion, to augment missing data in KT field, we first identified missing values by school testing schedules and then we train two deep generative models (i.e., VAE and LVAE) to generate quality data in the subject-based setting for imputation. With the imputed data, we are able to boost the original model by almost 50% in average RMSE. In addition, we validate the robustness of the imputed data and observe that only 10% of students data are needed to boost the original model performance for small to medium models such as LSTM and adaRNN and 50% of students data are needed to boost large models such as Transformer. In future, we plan to test the effectiveness of training using the varying length, instead of fixed length, on the model performance.

6 Acknowledgments

The work was mainly supported by NSF awards (1940076). Meanwhile, K12/Stride does not release student-level data to external research, but does use such data to continually improve the learning experience for students served in K12-powered schools. The authors are grateful to K12 for allowing its internal researchers to align research questions of general academic interest and publish some of those learnings.

References

Cheng, L.; Ramchandran, S.; Vatanen, T.; Lietzén, N.; Lahesmaa, R.; Vehtari, A.; and Lähdesmäki, H. 2019. An additive Gaussian process regression model for interpretable non-parametric analysis of longitudinal data. *Nature Communications*.

Dai, M.; Hung, J.-L.; Du, X.; Tang, H.; and Li, H. 2021. Knowledge Tracing: A view of Available Technologies. *Journal of Educational Technology Development and Exchange*, 14(2).

Du, Y.; Wang, J.; Feng, W.; Pan, S.; Qin, T.; Xu, R.; and Wang, C. 2021. AdaRNN: Adaptive Learning and Forecasting for Time Series. In *Proc. Int' Conf. Information and Knowledge Management*. ISBN 9781450384469.

Fährmann, D.; Damer, N.; Kirchbuchner, F.; Kuijper, A.; Choras, M.; and Pawlicki, M. 2022. Lightweight Long Short-Term Memory Variational Auto-Encoder for Multivariate Time Series Anomaly Detection in Industrial Control Systems. *Sensors*.

Giles, C. L.; Kuhn, G. M.; and Williams, R. J. 1994. Neural Networks: Theory and Applications. In *IEEE Transactions on Neural Networks*, volume 45, 89–90. IEEE.

Kingma, D. P.; and Welling, M. 2019. *An introduction to variational autoencoders*, volume 12. ISBN 9781680835502.

Le, T.; Wang, S.; and Lee, D. 2020. GRACE: Generating Concise and Informative Contrastive Sample to Explain Neural Network Model's Prediction. In *KDD*. ISBN 9781450379984.

Lee, Y.; Choi, Y.; Cho, J.; Fabbri, A. R.; Loh, H.; Hwang, C.; Lee, Y.; Kim, S.-W.; and Radev, D. 2019. Creating A Neural Pedagogical Agent by Jointly Learning to Review and Assess. In *arXiv preprint arXiv:1906.10910v2*.

Minn, S.; Yu, Y.; Desmarais, M. C.; Zhu, F.; and Vie, A. 2018. Deep Knowledge Tracing and Dynamic Student Classification for Knowledge Tracing. In *IEEE International Conference on Data Mining*.

Pandey, S.; and Karypis, G. 2019. A Self-Attentive model for Knowledge Tracing. In *Proc. Int'l Conf. on Educational Data Mining*.

Pardos, Z. A.; and Heffernan, N. T. 2011. KT-IDEM: Introducing Item Difficulty to the Knowledge Tracing Model. In *International Conference on User Modeling, Adaption and Personalization*.

Pearlmutter, B. A. 1989. Learning state space trajectories in recurrent neural networks. In *International JointConference on Neural Network*, 365–372.

Ramchandran, S.; Tikhonov, G.; Kujanpää, K.; Koskinen, M.; and Lähdesmäki, H. 2021. Longitudinal Variational Autoencoder. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 130.

Roderick J. A. Little, D. B. R. 2002. *Statistical Analysis with Missing Data*.

Shin, D.; Shim, Y.; Yu, H.; Lee, S.; Kim, B.; and Choi, Y. 2021. SAINT+: Integrating Temporal Features for EdNet Correctness Prediction. In *Proc. Conf. Learning Analytics and Knowledge*. ISBN 978-1-4503-8935-8.

Swamy, V.; Guo, A.; Lau, S.; Wu, W.; Wu, M.; Pardos, Z.; and Culler, D. 2018. *Deep knowledge tracing for free-form student code progression*, volume 10948 LNAI. Springer International Publishing. ISBN 9783319938455.

Vaswani, A.; Brain, G.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, ; and Polosukhin, I. 2017. Attention Is All You Need. In *Proc.Conf. Neural Information Processing Systems*.

Yu, M.; Xu, F.; Hu, W.; Sun, J.; and Cervone, G. 2021. Using Long Short-Term Memory (LSTM) and Internet of Things (IoT) for localized surface temperature forecasting in an urban environment.

Zhang, J.; Shi, X.; King, I.; and Yeung, D.-Y. 2017. Dynamic Key-Value Memory Networks for Knowledge Tracing. In *Int'l World Wide Web Conference Committee*. ISBN 9781450349130.