Semi-Supervised Learning for Predicting GMAT Scores

Yi-Fu Wu
Stanford University
yifuwu@stanford.edu

Abstract

We apply a method of applying semi-supervised learning to data from an online test preparation site in order to predict students' GMAT scores. We leverage a large amount of unlabeled data by using a variational autoencoder to extract feature embeddings for each student. We compare these feature embeddings with hand engineered features on a linear regression model and show that the feature embeddings perform comparably. We also show that combining the feature embeddings with the hand engineered features increases the performance of the model with hand engineered features alone.

1 Introduction

Semi-supervised learning leverages patterns found in large amounts of unlabeled data in order to improve prediction performance on a smaller set of labeled data. It has been shown to be an effective technique in domains such as image classification [1], natural language processing [2], and acoustic modeling [3]. In this project, we investigate applying semi-supervised learning to the domain of education, predicting students' GMAT scores based on their interactions with an online test prep site. More specifically, we leverage variational autoencoders (VAEs) [4] to extract feature embeddings from large amounts of unlabeled student interactions and use these embeddings for supervised learning on a smaller set of labeled data which consists of student reported GMAT scores. We compare our approach to standard linear regression that use hand engineered features on only the labeled dataset.

This problem is interesting because we will be able to increase the accuracy of our predictions with only a small amount of labeled data, which is expensive to collect, by leveraging a large amount of unlabeled data. From a practical standpoint, more accurate predictions on students' actual GMAT scores and question difficulty may help the test prep site produce a better experience for their students, potentially even augmenting their service by helping the students learn more effectively.

The input to our algorithm is the dataset of student practice exam data. The output of our algorithm is the predicted GMAT scores of the students.

2 Related Work

Drawing from the success of deep learning in other domains, [5] applies a Recurrent Neural Network (RNN) to model student learning, achieving state of the art results at the time. They use a one hot encoding of the students' questions which we also implement in our initial model. More recently, [6] use a bidirectional long short term memory (bi-LSTM) model to create embeddings for questions and students, which we also experimented with in this project.

Deep learning has also been recently applied more specifically to the domain of grade prediction, which is very relevant to our task of predicting GMAT scores. [7] introduces GritNet, which also uses a bi-LSTM model to make predictions on Udacity data. Their technique outperformed standard logistic regression models, especially in the first few weeks of learning. [8] uses Bayesian deep learning models to estimate student grades in future courses given grades in prior courses.

Our project mostly follows the approach from [9], where VAEs are used to create embeddings in a dataset to predict student learning disabilities. They run experiments using both simple fully connected layers in their network as well as an asymmetric VAE using a convolutional neural network (CNN) in the encoder and LSTM in the decoder. Our project attempts to take these techniques and apply them to the context of GMAT score prediction, although, as we show below, we find that a simpler model performs better for our dataset.

3 Dataset

We use a dataset from TAL Education Group, an online education company that provides online test prep for various exams including the GMAT. The original dataset consists of 8,681 questions with category labels (verbal, quantitative, integrated reasoning); 90,831 students; 16,002,324 student-question interactions consisting of a student, a question, a time the question was attempted and whether or not the student got the question correct; and 458 students labeled with self-reported GMAT scores. Out of the 458 students with self reported GMAT scores, 372 used the test prep system before the date of their reported exam. Out of these 372 students, we consider the 354 students who attempted at least 50 questions. Similarly, out of the 90,831 overall students, we only consider the 19,841 students that have completed at least 50 questions. The overall number of questions that these students attempted is 7808.

4 Methods

4.1 Variational Autoencoders

We use a variational autoencoders (VAE) [4] to create feature embeddings for the students in our dataset. VAEs are generative models that combine variational inference with deep neural networks. The goal is to maximize $P(X)$, the probability of each input X in our training set, assuming each X depends on a latent variable z:

$$P(X) = \int P_\theta(X|z)P(z)dz$$

The latent variable z is what we will use as our student embedding. Here we parameterize P by θ and represent it as a neural network. We model the prior $P(z) = \mathcal{N}(z|0, I)$. The posterior $P_\theta(z|X)$ is not tractable so we approximate it with an inference network $q_\phi(z|x)$:

$$q_\phi(z|x) = \mathcal{N}(z|\mu_\phi(x), diag(\sigma^2_\phi(x)))$$

where $\mu_\phi(x)$ and $\sigma^2_\phi(x)$ are both neural networks parameterized by ϕ. Since maximizing the likelihood directly is also not tractable, we maximize the variational lower bound (derived in [4]):

$$\mathbf{E}_{q_\phi(z|x)}[\log P_\theta(x|z)] - D_{KL}[q_\phi(z|x)||P(z)]$$

The negative of this expression is the loss function in our algorithm. The first term represents the reconstruction loss: given an input X, we sample z using $q_\phi(z|x)$ and then maximize $P_\theta(x|z)$. The second term is a regularization term that pulls $q_\phi(z|x)$ toward the prior $P(z)$.

4.2 Semi-supervised Learning

Our semi-supervised learning pipeline is shown in figure 1. It consists of two parts. First, we use the unlabeled student-question interactions as input to a VAE to generate feature embeddings z. The

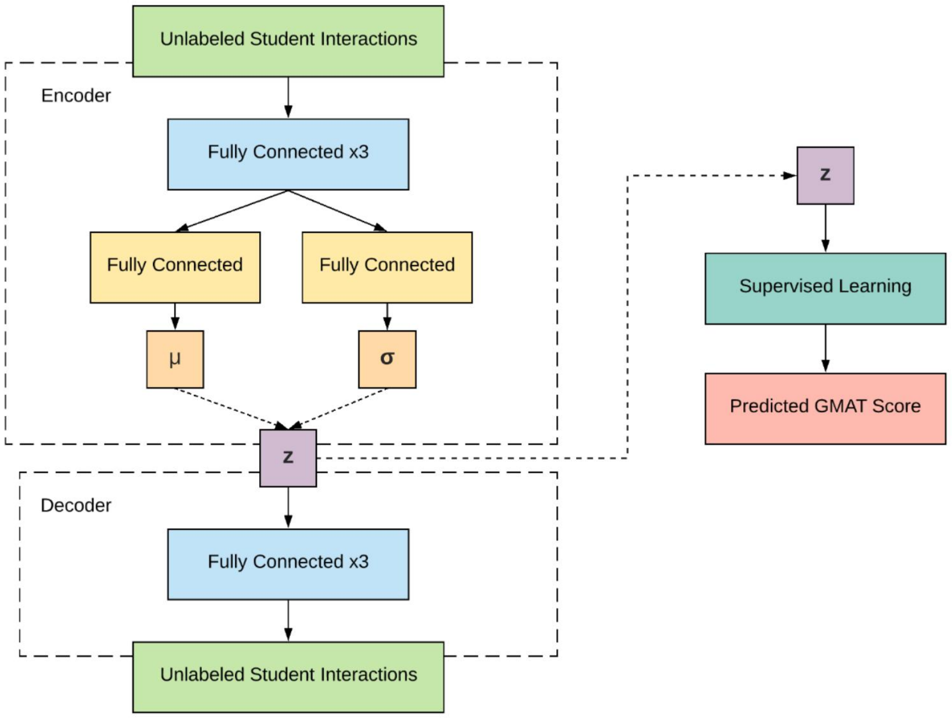

Figure 1: Semi-supervised pipeline. (left) Variational autoencoder using unlabeled student interactions. A series of three fully connected layers are used for both the encoder and the decoder. (right) The latent variable z is used as a feature embedding for supervised learning on the smaller labeled dataset to predict the GMAT score.

VAE uses three fully connected layers as both the encoder $q_\phi(z|x)$ and the decoder $P_\theta(x|z)$. The encoder uses another set of fully connected layers to generate $\mu_\phi(x)$ and $\sigma_\phi(x)$. We use these as parameters to a normal distribution and sample using the reparameterization trick [4] to generate our embeddings z. Next, we use these feature embeddings for supervised learning on our labeled dataset to predict GMAT scores. We run experiments that use the feature embeddings alone to make the predictions as well combining the embeddings with our original hand engineered features.

4.3 Experiment Setup

We originally attempted to model our data similarly to [5] and [9] where each input consists of a sequence of student-question attempts. In [5], each attempt is modeled as a one-hot encoding of the questions and in [9], each attempt consists of all features for the student during their attempt of that question. Since they are trying to predict student disabilities, they use total answer time and the first input response time as the two features for 25 questions, resulting in a 50-dimensional input for each student. For our dataset, we experimented with using a one-hot encoding of the questions resulting in a $T * 7808$ (where 7808 is the number of questions) dimensional input for each student. We later also attempted to use the question text to create a question embedding using InferSent [10] resulting in a $T * 4096$ dimensional input. Unfortunately, running both formulations through our VAE did not generate good feature embeddings – they did not improve our results using the standard supervised learning algorithms, so we did not include them in this write-up.

The above formulation may have unnecessarily forced our VAE to learn the order in which the student attempted each question, which seems less relevant in the context of predicting final GMAT scores. Therefore, we ended up taking a simpler approach of modeling each user as a 7808-dimensional vector where each entry in the vector represents a question. We store 1 if the student's last attempt at

Table 1: Comparison of Single Grasp Approaches on the Cornell Grasping Dataset

Random Seed	**Features**	**RMSE**	**MAE**	R^2
1	Hand engineered	68.5044	52.4286	0.3351
1	VAE	75.4983	56.0000	0.1924
1	VAE + Hand engineered	68.4418	53.2857	0.3363
2	Hand engineered	49.6560	39.4286	0.2993
2	VAE	52.3450	40.2857	0.2214
2	VAE + Hand engineered	49.5263	38.4286	0.3030
3	Hand engineered	54.8895	41.5714	0.4161
3	VAE	68.2352	50.2198	0.2231
3	VAE + Hand engineered	51.6582	39.4286	0.4828
4	Hand engineered	67.1884	46.2857	0.2690
4	VAE	65.9329	47.8571	0.2960
4	VAE + Hand engineered	62.1174	42.7143	0.3751
5	Hand engineered	74.3928	53.4286	0.1935
5	VAE	70.0306	51.2857	0.2853
5	VAE + Hand engineered	62.1174	42.7143	0.3072

the question was correct, 0 if the student's last attempt at the question was incorrect, and a dummy value of -1 if the student did not attempt the question. During training, we ignore the entries the student did not attempt in or loss function.

Similar to [9], we found from our initial training attempts that the KL divergence term in our loss function went to zero and we ended up learning the prior for our encoder $q_\phi(z|x) = P(z)$. This was also observed in [11] where a warm-up phase of gradually turning on the KL divergence term is introduced. We follow a similar approach and modify our objective:

$$\mathbf{E}_{q_\phi(z|x)}[\log P_\theta(x|z)] - \beta D_{KL}[q_\phi(z|x)||P(z)]$$

where β is linearly increased from 0 to 1 during the first 100 epochs of training.

We also found that there was quite a bit of variation in our supervised learning results when we changed the random seed used to generate our training / test set split for our small labeled dataset, so we ran experiments with a few different random seeds and report all of the results below. Additionally, in the interest of time, we only used linear regression as our supervised learning algorithm. We also ran experiments using decision trees, SVM regression, and gradient boosting with similar results, but these algorithms were more time consuming to tune since they were more sensitive to changes in hyperparameters.

We calculate the root mean square error (RMSE), mean absolute error (MAE), and R^2 for each run of our experiments. For the purposes of evaluating how good a model is, we use R^2. For each random seed, we run a set of three experiments: (1) hand engineered features (total questions attempted, total practice exams attempted, average score in terms of number of questions the student got correct, average score for the last 100 exercises, whether or not the student purchased the product, the average score improvement between the first half of questions the student attempted and the second half) (2) the VAE generated feature embeddings (3) a concatenation of hand engineered features and VAE generated features.

5 Results

The results of our experiments are shown in Table 1. Our overall R^2 scores are low, indicating that both the hand engineered features and the VAE generated features may not be the best model for predicting GMAT scores in the dataset. Our VAE generated features generally performs comparably to the hand engineered features, although there is a bit of variance. However, combining the hand engineered features with the VAE features seems to increase the R^2 across the board. This indicates that we can indeed leverage the large amount of unlabeled data in this dataset to improve the performance of our supervised learning model. Although most of the improvements of combining the features are modest, the random seed 4 and random seed 5 runs show that significant improvements in R^2 is possible by leveraging the VAE generated data.

6 Conclusion and Future Work

In this project, we explored a method to improve prediction performance of GMAT scores by leveraging a large amount of unlabeled student interaction data. We used a variational autoencoder to extract feature embeddings and used these feature embeddings as input to a linear regression model. Although the R^2 of our experiments were not very high, they do seem to indicate that we can improve the prediction performance of a linear model using hand engineered features by introducing VAE generated features.

Future work on this project could explore using the extracted feature embeddings on other regression algorithms such as decision trees, SVM regression, or gradient boosting methods. It would also be interesting to explore different VAE architectures such as ones that include convolution or recurrent layers.

7 Acknowledgements

The scope and direction of this project was suggested by Sherry Ruan, a PhD student at Stanford, whose research group is working on related studies with the same dataset. In addition to proposing the idea for the project, Sherry also suggested papers with related work and helped answer questions and provide guidance along the way. Sherry and her collaborator Liwei Jiang also implemented the baseline code for the prediction task using the hand engineered features, which we used as a starting point for this project.

References

[1] Diederik P. Kingma, Danilo Jimenez Rezende, Shakir Mohamed, and Max Welling. Semi-supervised learning with deep generative models. In *NIPS*, 2014.

[2] Joseph P. Turian, Lev-Arie Ratinov, and Yoshua Bengio. Word representations: A simple and general method for semi-supervised learning. In *ACL*, 2010.

[3] Hank Liao, Erik McDermott, and Andrew W. Senior. Large scale deep neural network acoustic modeling with semi-supervised training data for youtube video transcription. *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 368–373, 2013.

[4] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2014.

[5] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J. Guibas, and Jascha Sohl-Dickstein. Deep knowledge tracing. In *NIPS*, 2015.

[6] Yu Su, Qingwen Liu, Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Chris H. Q. Ding, Si Wei, and Guoping Hu. Exercise-enhanced sequential modeling for student performance prediction. In *AAAI*, 2018.

[7] Byung-Hak Kim, Ethan Vizitei, and Varun Ganapathi. Gritnet: Student performance prediction with deep learning. *CoRR*, abs/1804.07405, 2018.

[8] Qian Hu and Huzefa Rangwala. Reliable deep grade prediction with uncertainty estimation. *CoRR*, abs/1902.10213, 2019.

[9] Severin Klingler, Rafael Wampfler, Tanja Käser, Barbara Solenthaler, and Markus H. Gross. Efficient feature embeddings for student classification with variational auto-encoders. In *EDM*, 2017.

[10] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[11] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. In *NIPS*, 2016.