NeTra: A Neuro-Symbolic System to Discover Strategies in Math Learning

Anup Shakya
University of Memphis
ashakya@memphis.edu

Vasile Rus
University of Memphis
vrus@memphis.edu

Stephen Fancsali
Carnegie Learning, Inc.
sfancsali@carnegie learning.com

Steve Ritter
Carnegie Learning, Inc.
sritter@carnegie learning.com

Deepak Venugopal
University of Memphis
dvngopal@memphis.edu

ABSTRACT

Understanding how students with varying capabilities think about problem solving can greatly help in improving personalized education which can have significantly better learning outcomes. Here, we present the details of a system we call NeTra that we developed for discovering strategies that students follow in the context of Math learning. Specifically, we developed this system from large-scale data from MATHia that contains millions of student-tutor interactions. The goal of this system is to provide a visual interface for educators to understand the likely strategy the student will follow for problems that students are yet to attempt. This predictive interface can help educators/tutors to develop interventions that are personalized for students. Underlying the system is a powerful AI model based on Neuro-Symbolic learning that has shown promising results in predicting both strategies and the mastery over concepts used in the strategy.

Keywords

Intelligent Tutoring Systems, Learning Strategies, Neuro-Symbolic AI, Deep Learning

1. INTRODUCTION

In this paper, we describe our system which we call NeTra (Neuro-symbolic system for sTrategy discovery). NeTra was developed from large-scale open source Math learning data from Carnegie Learning's MathIA platform [6]. The data in this case corresponds to interactions between the student and the tutor. The goal of NeTra is to provide teachers/educators an intuitive interface to view *how* a student is likely to solve a problem *before* they have actually attempted to solve that problem.

The system consists of intuitive dashboards that can help teachers analyze the performance of students based on predictions made by an AI model. In particular, the system predicts the strategy (a sequence of steps) [5] the student is likely to follow when solving a problem. Further, while it is important for teachers to know whether a student follows the correct sequence of steps, it is also important to know if a student is likely to be able to apply these steps within the problem context. For example, if the two steps to solve a equation are re-arranging the variables and then making substitutions, the student needs to know that he/she should perform these steps in sequence and also in the context of the problem, the student should be able to apply the concepts that are central to each step correctly to solve the full problem. The system therefore also jointly predicts the mastery over concepts used in the strategy. Since the two are known to be related [7], joint prediction helps us take advantage of their dependency.

The underlying AI model that predicts the strategy is trained with a large dataset consisting of millions of interactions. In particular, we develop a Neuro-Symbolic model [4] where we consider relationships in the data (e.g. different problems that the same student works on are related, different students working on the same problem are related, etc.). The model then learns *embeddings* [1] or vector representations for students and problems such that students and problems with symmetric relationships are close to each other in the embedding space. This allows us to learn clusters of exchangeable students/problems. The strategy model that predicts the likely strategy and the mastery model that predicts correctness of the steps in the strategy are then trained from embeddings sampled from the clusters. Specifically, to train a model that predicts strategies, we use a one-to-many LSTM that outputs a sequence of steps. Technical details about the model can be found in [8]. We also jointly predict if a student can apply the steps in the strategy correctly. Specifically, to estimate mastery over the steps in the strategy, we predict the likelihood of the student correctly solving each step in their first attempt. We use an

attention-based transformer model [10] trained on strategies augmented with positional encodings such that it attends to key components in the strategy to estimate the mastery. A schematic overview of the AI model's architecture is shown in Fig. 1.

1.1 Overview of NeTra

NeTra consists of dashboards developed using the Django web framework. The underlying AI models are trained offline and the training takes approximately 6-8 hours to complete on a 64GB 16-core machine with a NVIDIA GPU. The models are currently trained on two datasets: Bridge-to-Algebra-2008-2009 and the Carnegie Learning 2019-20, both of which are publicly available in PSLC datashop [9, 2]. We next describe three important dashboards in the system.

1.2 Strategy Dashboard

The strategy dashboard is a tool for educators to simulate the performance of a student on any problem. For this, the system prompts the user to select a student s and a problem p. The AI model predicts the strategy that s is likely to follow to solve problem p as a sequence of knowledge components (KCs) [3]. Further, the model also predicts if a step is likely to be solved correctly in the first attempt (called CFA). A screenshot for the strategy dashboard is shown in Fig. 2. The UI design has separate cards providing different views. Under the predicted strategy card, the predicted sequence of KCs is shown. The green boxes in the sequence denote that the step was predicted to be solved correctly in the first attempt (CFA=1) and the red boxes denote that the model predicted that the student makes a mistake (CFA=0) in the corresponding step. The model also estimates the cumulative mastery over KCs in the predicted strategy. To do this, we aggregate the CFA predictions corresponding to a KC over exchangeable problems. This indicates if the student can correctly apply the concept in several related problems which indicates mastery over the concept. Specifically, we cluster problems based on their embeddings and based on this, let $\mathbf{p} = \{p_1 \ \ldots \ p_k\}$ be problems that belong to the same cluster as the current problem p. For each $p' \in \mathbf{p}$, we predict the strategy that the current student s is likely to follow as well as the correctness of the strategy. To estimate mastery over a KC K, we compute the % of cases where the student has correctly applied K in their strategies for problems in $\mathbf{p}$. The estimated mastery over KCs is displayed using the polygon radar chart as shown in Fig. 2. Thus, using this dashboard, it is possible to analyze if a student is i) following the wrong strategy, ii) following the right strategy but lacks the mastery required to solve the problem using this strategy or iii) following the right strategy and also correctly solves the steps in the problem but lacks overall mastery in KCs related to this strategy (which may indicate uncertainty since the student is unable to successfully replicate his/her performance over related problems). Thus, using this information, we envision the design of more personalized interventions for the student.

1.3 Student Dashboard

This dashboard shows detailed student information. From the main strategy dashboard in Fig. 2, the user can select the student information and a new dashboard is displayed as shown in Fig. 3. The details include a graphical view of the predicted mastery over concepts over all attempted problems. Specifically, for every KC that the student has used in at least one strategy, we compute the % of instances in which it is predicted that the student has correctly solved a step using this KC. This allows us to quickly visualize the strengths and weaknesses of a student aggregated over all strategies used by that student. Further, this dashboard also shows a neighborhood of students who are similar to the current student. To do this, we visualize the nearest neighbors of the selected student in the embedding-space. Recall that the embedding tends to place students who are approximately exchangeable (likely to use similar strategies) with the selected student close to each other. This allows a user to navigate through groups of similar students analyzing their performance. In future, we plan to add more specific group analysis to draw attention to performance over groups rather than individuals. Finally, we also show a summary of the cluster to which the student embedding belongs. Specifically, we compute the average success which computes the average steps that are predicted as correct for all students within a specific cluster. Also, we display sections where the average performance was the worst (in terms of average mistakes made over all steps). This allows a user to understand the relative performance of the current student compared to others in his/her symmetry-group. Thus, we can envision a scenario where this can be used for evaluating a student's progress relative to the progress of similar students.

1.4 Problem Dashboard

Analogous to the student dashboard, selecting the problem from the main dashboard shows more details about the specific problem. Fig. 4 shows the screenshot of the problem dashboard. Specifically, we see aggregate statistics for the specific problem. For example, % of steps that were predicted as being correctly solved (CFA=1) aggregated over all students who attempt the specific problem, the number of students attempting the problem, etc. Further, it also shows information from the problem clusters, i.e., clusters learned from embeddings over problems. Analogous to the student clusters, here, we show the same metrics aggregated over problems. This allows us to analyze the performance of students for current problem relative to other problems that are similar to it. In this dashboard, we also show the most common strategies used by the students to solve the problem. To do this, we use an approximate matching method based on positional encodings [10] to check if two strategies are similar to each other. For the current problem, we display the strategies that match closely with a maximum number of other strategies used for the same problem. This allows us to analyze if we need to provide hints to all students to guide them towards more efficient strategies in case the most common strategies are found to be problematic.

1.5 Use Cases

Below are some potential use cases where we envision NeTra to be applicable.

- Interventions based on incorrect strategies for specific students or over specific problems. This can be in the form of better hints that can guide the student, examples, etc.

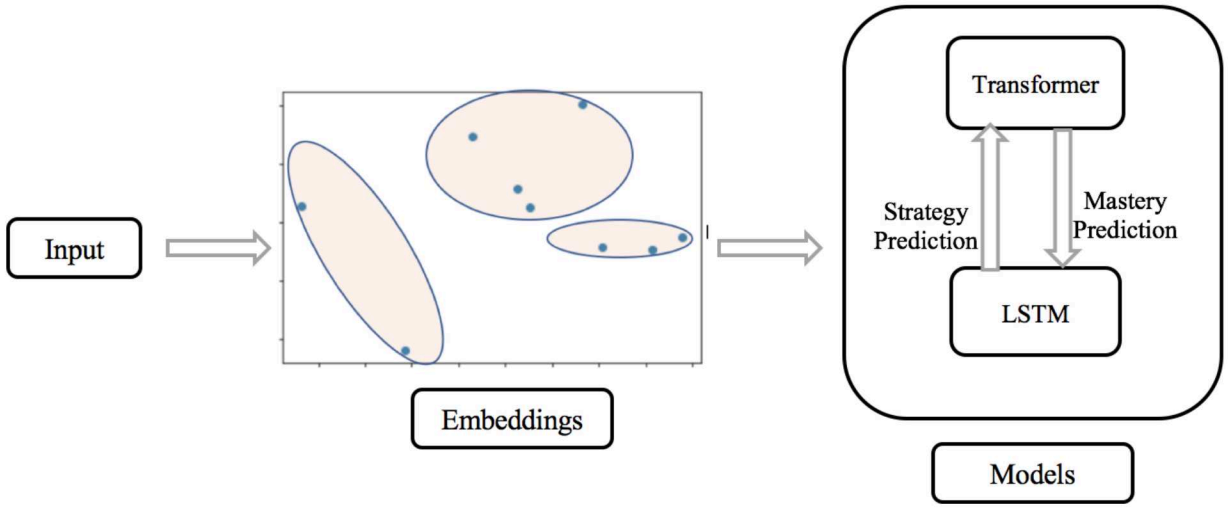

Figure 1: Overview of the learning model.

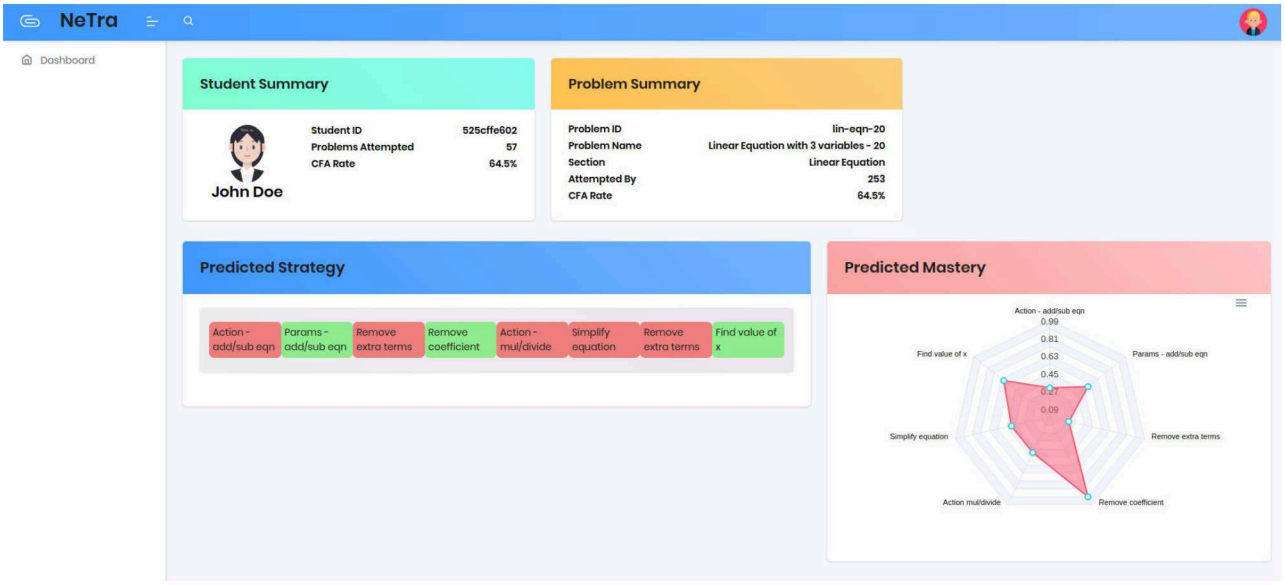

Figure 2: Demo of the Strategy dashboard presenting the predicted strategy and mastery over different concepts.

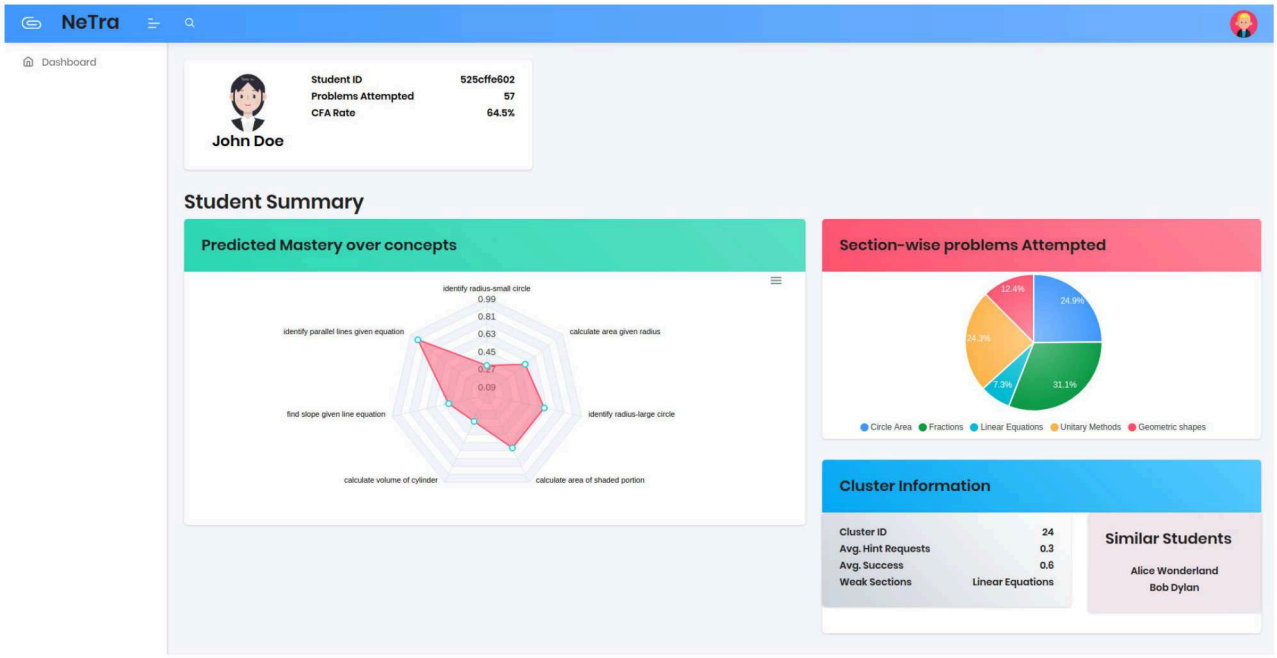

Figure 3: Demo of the Student Dashboard displaying details about the student.

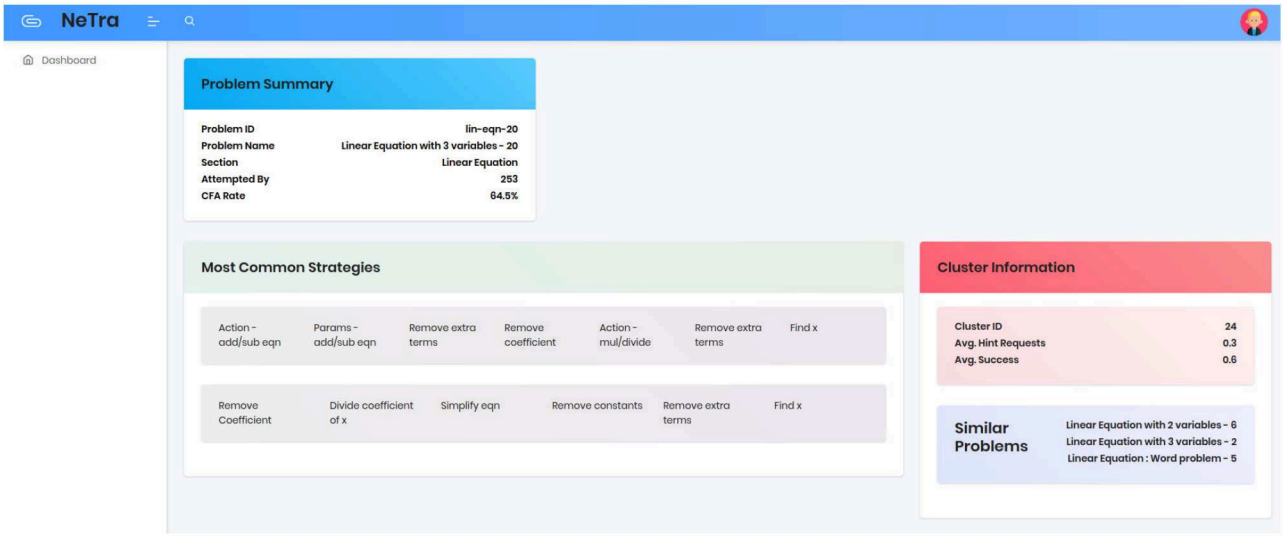

Figure 4: Demo of the Problem Dashboard showing details about the problem.

- Personalized problem selection where we select exchangeable problems. This allows a student to demonstrate mastery by applying the same concepts in related problems but with different contexts. This can help also reduce uncertainty over whether the student has truly understood a concept.

- Clusters can provide aggregate information over groups and this can be analyzed to ensure fairness over diverse groups. For instance, do all the groups have similar performance or is there a high degree of variance between groups. A more detailed dashboard that can help analyze fairness is something that we hope to add to our tool in future.

- The correlation between strategies and mastery over concepts can be analyzed and the progression of students can be monitored. For example, as the mastery over concepts increases, do the strategies become more advanced, if not, cues that help in transitioning to more effective strategies can be provided.

2. FUTURE WORK

There are several new features that will be added to `NeTra`. Specifically, we will add more in-depth cluster summarization to simultaneously visualize information from several clusters which can help in higher-level analysis. Next, the current dashboards are not real-time in the sense that they do not update the strategy as a student is solving a problem. The dynamic updates can help with real-time interventions by Intelligent Tutoring Systems. Also, an interface for updating the model has not yet been developed. Currently, the trained model is loaded at the start and used to make all predictions. We will add options to load different trained models for predictions. An explanation interface will be added to improve interpretability of the system. Specifically, note that since our model uses attention-based transformers, we can identify key parts of the strategy used by the model for predictions. Visualizing this can help explain how the model is understanding the strategy thus making the system more transparent. Finally, the current dashboards are focused on predictions and not on training. The use of Neuro-Symbolic AI can help users interact with the system by stating assumptions or knowledge about the data which can be incorporated into training the model. A future goal is add this interface where different constraints related to fairness, validity, etc. can be added during training and the results can be verified based on predictions by the system.

References

[1] M. M. Islam, S. Sarkhel, and D. Venugopal. On lifted inference using neural embeddings. In *AAAI Conference on Artificial Intelligence*, pages 7916–7923, 2019.

[2] K. R. Koedinger, R. S. Baker, K. Cunningham, A. Skogsholm, B. Leber, and J. Stamper. A data repository for the edm community: The pslc datashop. *Handbook of educational data mining*, 43:43–56, 2010.

[3] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cogn. Sci.*, 36(5):757–798, 2012.

[4] L. C. Lamb, A. S. d'Avila Garcez, M. Gori, M. O. R. Prates, P. H. C. Avelar, and M. Y. Vardi. Graph neural networks meet neural-symbolic computing: A survey and perspective. In *IJCAI*, pages 4877–4884, 2020.

[5] S. Ritter, R. Baker, V. Rus, and G. Biswas. Identifying strategies in student problem solving. *Design Recommendations for Intelligent Tutoring Systems*, 7:59–70, 2019.

[6] S. Ritter, R. Carlson, M. Sandbothe, and S. E. Fancsali. Carnegie learning's adaptive learning products. *Educational Data Mining*, 2015:8th, 2015.

[7] S. Ritter, M. Yudelson, S. E. Fancsali, and S. R. Berman. How mastery learning works at scale. In *L@S*, pages 71–79. ACM, 2016.

[8] A. Shakya, V. Rus, and D. Venugopal. Student strategy prediction using a neuro-symbolic approach. In *EDM*, 2021.

[9] J. C. Stamper, K. R. Koedinger, R. S. J. de Baker, A. Skogsholm, B. Leber, S. Demi, S. Yu, and D. Spencer. Datashop: A data repository and analysis service for the learning science community (interactive event). In *AIED*, volume 6738 of *Lecture Notes in Computer Science*, page 628. Springer, 2011.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.