SmartPrimer: Online Learning Hint Generation in an Imitation Learning Framework

Dominik Damjakob [*1] **Helgi Hilmarsson** [*1] **William Steenbergen** [*1] **Emma Brunskill** **Sherry Ruan**

Abstract

This research is part of the Stanford SmartPrimer project, and aims to use imitation learning to automate the process of giving hints to children while they try to solve a simple task. The data consists of 19 online conversations between children and a teacher, recorded while the children solve a geometry problem. States are represented by combinations of a child's progress along the question along with their demographic and conversation data. Both regression and classification models were used to predict the expert's action including linear regression, LASSO, logistic regression (with regularization) and random forest. These models were compared with the industry standard of giving hints in increasing order, and are evaluated on how well they imitate the real teacher. It was concluded that many of the methods outperform the baseline, and the conversation data was proven to be valuable for predicting the given hint.

1. Introduction

Intrinsic motivation to learn was found to positively correlate with achievement and IQ while it inversely correlates with anxiety (Gottfried, 1990). Children naturally start their academic life intrinsically motivated to learn, but researchers found that during middle school many children's interest in schooling declines (Anderman & Maehr, 1994). To battle this decline, a lot of research aims to find methods that intrinsically motivate children again during these years. Most of these methods try to create experiences similar to what children already are intrinsically motivated to spend time on (e.g. video games, books or television shows). One commonality under these activities is the fact that they are all based on narratives and storytelling. The Smart Primer project at Stanford University (Sma), builds on this commonality by experimenting with digital environments that use storytelling to teach children new things.

Most methods that use narrative require personalized teaching. Personalized learning has been shown to be one of the most effective methods (Pane et al., 2017), but one downside is that it requires a substantial amount of resources in terms of teachers and mentors. This paper aims to overcome this downside by decreasing the resources needed through the automation of some of those tasks. More precisely, the goal is to select good hints along the learning process of students solving a simple project.

Automatised hint generation is starting to gain ground in industry, with companies like Coursera, Khan Academy and Udacity starting to experiment. The current industry standard is to select hints by increasing the amount of knowledge given with every hint. The most basic hint is given first, and the answer is given as the last hint. This research proposes to improve on this heuristic by using imitation learning to mimic more explicitly what a human teacher would do.

1.1. The experimental design

This paper uses data gathered by an experiment carried out as a part of the Smart Primer project. In this experiment, children aged from 7 to 11 receive the task of solving a simple geometry problem where they have to calculate the volume of a box by going through 6 sequential sub-questions. They are virtually narrated through every problem by a fictional wizard who gives hints, asks questions and engages with the child. The children have the possibility to type messages to the wizard, and the wizard will stimulate them to do so. The messages sent by the child, in addition to the time it takes a child to progress through the answer are the main reasons for the wizard to provide hints.

The wizard is played by a human who can only choose between giving 4 to 8 (depending on the sub-question) different hints per sub-question. Additionally the wizard makes small talk with the children, and asks questions about their understanding. The scope of the research is to only model/imitate the hints the wizard gives and disregard the small talk and other messages. The data includes all messages sent by the child and wizard, timestamps of these messages, and the sub-question that the child was working on at a given time.

The hints are ordered based on the amount of information

[*]Equal contribution [1]Stanford University. Correspondence to: William Steenbergen <wsteen@stanford.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

they give away, with the last hint providing the final answer to the sub-question. The wizard's end goal is to teach the child geometry, not necessarily to make him/her solve the problem (although these are highly correlated). This usually boils down to providing the hint that gives away the least of the solution while still helping the child take the next step in solving the problem.

The data also includes demographic data about the children, such as their age, level in school and gender. The children took a math test before and after solving the problem, and one could define the improvement of a child's understanding by the difference in the math score. There are data available on 19 children.

2. Background and related work

The current state of the art research can be divided in two: The first is student knowledge tracing through problem attempt data (e.g. Cen et al. (2006) and Matsuda et al. (2015)). These methods try to model the student's knowledge and try to get a deep understanding on how a student learns, irrespective of the problem.

The more recent approach uses student attempt data to trace solution paths for a particular problem (e.g. Rivers & Koedinger (2017) and Stamper et al. (2013)). These methods often focus on 'solution path construction', in which it models how students can move through a specific problem by defining solution states as an MDP. These solution paths are defined by the guesses and tries that the user makes before ending up at the correct solution.

Stamper et al. (2013) constructs the solution path by modeling the 'states' as different attempts people make, but this has the downside that this only works when the number of states is relatively small. Rivers & Koedinger (2017) goes one step further by allowing for continuous states using a method that is similar to the Value Function Approximation approach often seen in the reinforcement learning field. It still only uses attempt data.

The approach taken in his research falls into the solution path construction and specific problem modeling category. However, where most methods described only use attempt data to define the states, a contribution is made to the existing literature by also using conversation data and demographic data of the user.

3. Descriptive statistics

As mentioned, the data consists of 19 conversations between different children and experts. In total, 98 hints were given during these 19 conversations. There are on average 3.41 hints per conversation and the children average 17.7 words per conversation. The most common words typed by the children can be found in Figure 1. The distribution of hints per sub-question can be found in Figure 2.

In the 19 different conversations, a rough distinction can be made between 2 different types of conversations. 8 out of 19 conversations are very short and contain only little information. The child introduces itself but then stops responding to the teacher. The teacher reminds the child multiple times that he/she is there to provide help, but the child does not respond. The other 11 conversations can be considered more useful, and contain more lengthy interaction between the child and the teacher. See Figure 3 for an example of such a conversation.

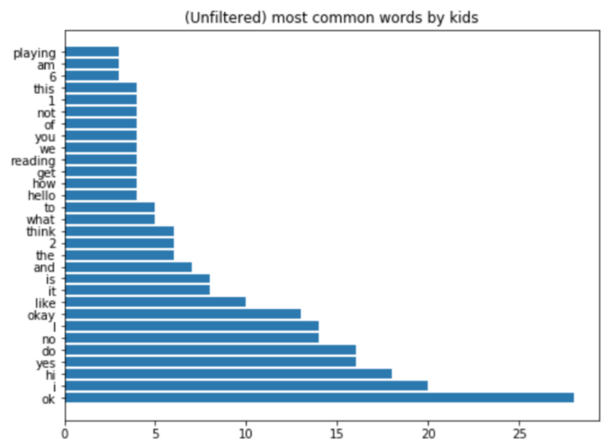

Figure 1. The 30 most common words used by the children.

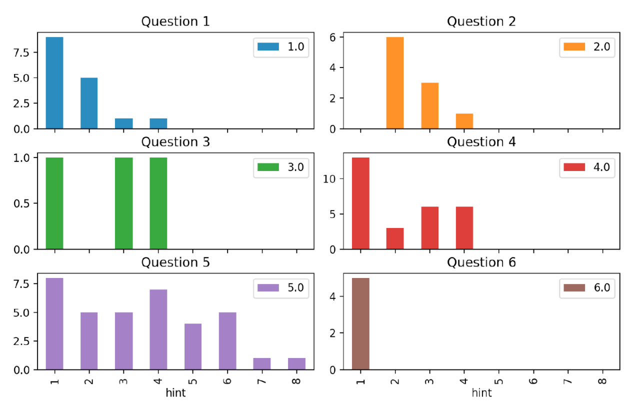

Figure 2. The distribution of hints per sub-question. Note that the y-axes differ per table.

One way to extract meaning from the messages sent by the children is by representing the text as vectors using Google's Word2Vec (Mikolov et al., 2013). The messages sent by the children between given hints are encoded in 300-dimensional vector space, using a word2vec model

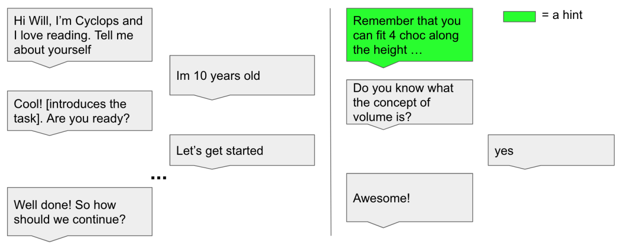

Figure 3. An example of a conversation that is considered more useful.

from Gensim trained on data from Google news (Gen). To visualize the outcome, the 300 dimensions were reduced to only two dimensions using principal component analysis (PCA) and labeled. The result can be seen in Figure 4.

It is clear from Figure 4 that the words used before hint 1 seem to cluster in the middle and from further analysis it was found that most of these sentences include words like 'hi' or 'name', indicating that these are sentences in which the child introduces itself. It must be noted that the figure shows messages from different sub-questions, so sentences before hint 1 are not necessarily the conversation starters. It can be noticed that short words like 'no', 'yes' and 'ok' tend to appear before later hints, possibly explained by the teacher inquiring more information about what the child had difficulties with.

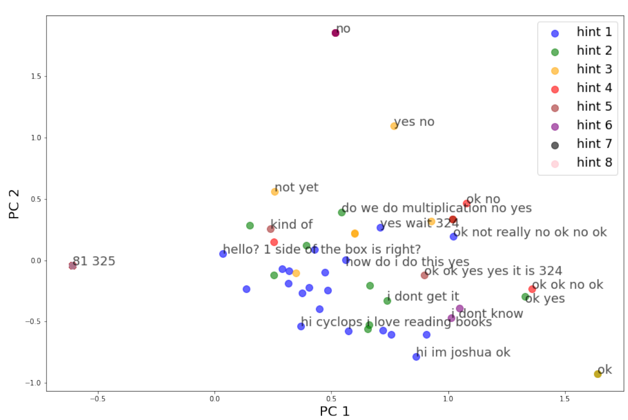

Figure 4. 300 dimensional Word2Vec encoding of the messages sent by the children before the labeled hints. The dimensions were reduced to 2 using PCA. Names are anonymized.

4. Methodology

The method proposed to achieve the goal of maximizing the students learning is imitation learning. That is, it is assume that the wizard is an expert that knows exactly what hint the child needs at a given state, and the model tries to imitate this behavior. The problem can then be framed as a supervised learning problem where the features are some state representation of the child's process and the responses are the expert's actions.

To imitate the expert's actions, the model has to be able to predict which hint to give. For this setting, it is assumed that the model knows when to give a hint and the only problem is to figure out which hint to give. This simulates the real application in which the user can pro-actively ask for a hint, or in which the time frames of when hints should be given are pre-defined. In this approach the model is given the information about when the expert gives hints, and it only considers giving hints at these exact same moments.

4.1. State Representation

The state space representation is encoded using combinations of the following features:

1. The current sub-question the child is solving (Q)
2. Previous hint given/predicted (H)
3. Meta features (M)
   - Demographic variables such as age, gender and grade for a math test
   - Features describing the words said by the child such as number of words used, words including a question and words including a number
4. Words[1] used by the child encoded using a bag of words model (B)
5. Word2vec representation of words[1] used by the child in a 300 dimensional vector space reduced to 4 dimensions by using PCA (W)

Different combinations of these 5 sets of features were used to generate predictions. For example, Q+H is the set of features containing the current sub-question and previous hint that was given. The set of features containing word representations, B and W, are referred to as NLP methods and are never combined.

4.2. Models

One can use both regression models and classification models to predict what hint to give. Regression models assume that there is an ordering in the hints. This is an intuitive assumption in the sense that the hints are naturally ordered containing an increasing level of information. As such, a regression model would predict the amount of information

[1] Only words that were used more than once across all conversations are taken into account

a child needs at a given time. In contrast, the classification models disregard the ordinality of the hints.

The following regression and classification models are used:

1. Linear regression
2. LASSO with cross-validated lambda selection
3. Logistic regression with 1 norm penalty (Lasso penalty)
4. Random forest

4.3. Performance Evaluation

The performance of the models is measured in two ways, Accuracy and Mean Absolute Error (MAE):

$$Acc = \frac{1}{KT}\sum_{i=1}^{K}\sum_{t=1}^{T_i} I[H_{i,t}^{true} = H_{i,t}^{pred}]$$

$$MAE = \frac{1}{KT}\sum_{i=1}^{K}\sum_{t=1}^{T_i} |H_{i,t}^{true} - H_{i,t}^{pred}|$$

With $H_{i,t}$ being the tth true or predicted hint generated for child i, T_i ($T = \sum_{i=1}^{K} T_i$) being the number of hints given to child i and $K = 19$ the number of children. Accuracy captures how often the model deviates from the expert's policy and the MAE gives an indication of how far the model is off.

The models were evaluated using the average performance score of a custom implementation of leave-one-out cross validation (LOOCV). The implementation withholds one child as a test set, which includes multiple hint predictions, and uses the remaining 18 children to train and tune the predictive model using separate cross validation. If the previous hint given is used as a feature, the previous *predicted* hint is used when evaluating the models. This is done to simulate the real world application as much as possible, to get a representative performance indication.

To get an intuition for how well the tested models are doing, they are compared to the industry standard of giving hints in increasing order, starting with the first one for every sub-question (baseline).

4.4. Validation

To validate that the models would find any strong correlations between given hints and the NLP state features, a data set with strong relationships was simulated. This data set consists of 5 groups of 4 fictional children each. Every group receives a different hint ordering:

- 1→ 2→ 3→ 4 (the Baseline order)
- 2→3→4
- 3→4
- 2→4→3
- 1→3→2

The children all say different sentences and words, but to create a strong correlation between some of the words and the following given hints, the messages sent by the child always included certain words, characteristic for that following hint. Before hint 1, the child always uses the word 'start'. For hint 2, the key words are 'box' and 'chocolate'. Question 3 has 'pencil' and 'understand' and question 4 always includes a number and the word 'think'.

The simulated children use full sentences, and other words could both act as random noise but could also add to the correlation. It is expected that the models find the signal between the chosen words and the given hint and that they perform much better with the NLP data than they do without.

5. Results

5.1. Performance Comparison

The performance of the models over 4 different set of features is summarized in Figures 5 and 6. The exact numbers can be found in Appendix A.

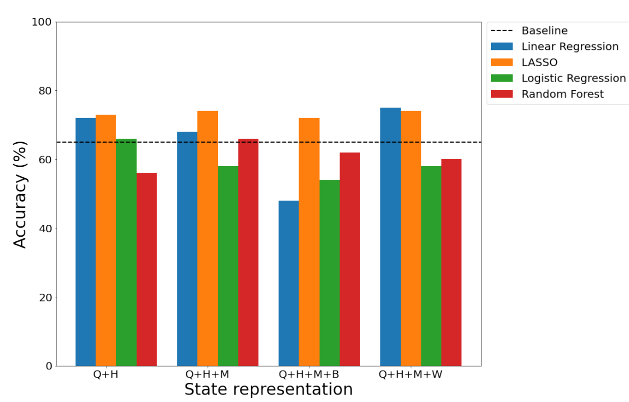

Figure 5. Classification accuracy comparison for the models across different state representations. Q is current sub-question, H is the previous hint given, M is meta features, B and W s are Bag of Words and word2vec representations, respectively. Note that larger values indicate better performance.

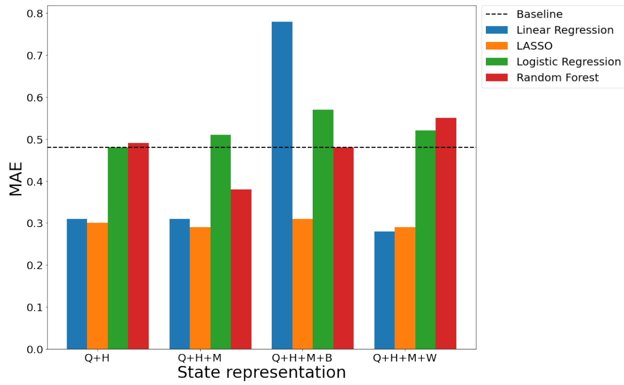

Figure 6. MAE comparison for the models across different state representations. Q is current sub-question, H is the previous hint given, M is meta features, B and W are Bag of Words and word2vec representations respectively. Note that smaller values indicate better performance.

It is clear that the linear regression models outperform both the baseline and the classification models in terms of both classification accuracy and MAE. The highest accuracy is reached by the linear regression model using the Q+H+M+W state representation, achieving 75% accuracy, clearly beating the baseline accuracy of 65%, the same model also attains the best MAE of 0.3, outperforming the baseline MAE of 0.48. In contrast to the regression models, the classification models tend to perform similarly or worse than the baseline for both metrics. Comparing the state representations, it seems as if adding the bag-of-words representation mainly decreases the performance of the models except for the robust LASSO (which in that case happens to shrink the coefficients of the added words to 0). However, the W2V representation does seam to give a small improvement, especially for the the linear models.

5.2. Contrasting Models

For illustrative purposes, the paths of 2 different models are visualized in in Figures 7-10. The trajectories are plotted for questions 2 and 5 along with trajectories of both the expert and the baseline. The best performing model, linear regression for the Q+H+M+W state representation, is plotted in Figures 7 and 8 and trajectories of the non-linear classifier random forest is depicted in Figures 9 and 10 for the same set of features.

In general, the trajectories of the linear models tend to be quite similar to the baseline. Interestingly, the paths in question 2 show how the model skips hint number 1 and gives hint number 2 straight away, as opposed to the baseline which always start by giving hint number 1. Further, it does not always simply increment its hint prediction by 1 as can be seen for both of the questions.

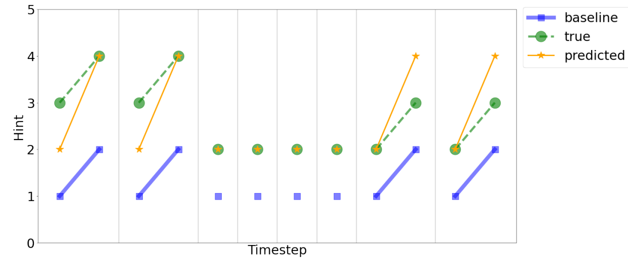

Figure 7. Trajectories of the linear regression using the Q+H+M+W state representation for question 2. Vertical lines separate trajectories of different students.

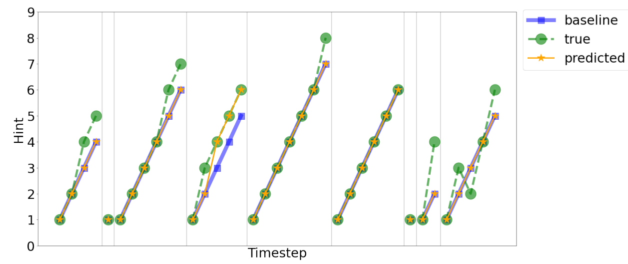

Figure 8. Trajectories of the linear regression using the Q+H+M+W state representation for question 5. Vertical lines separate trajectories of different students.

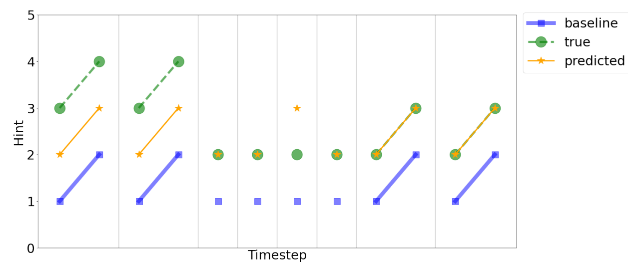

Figure 9. Trajectories of the random forest using the Q+H+M+W state representation for question 2. Vertical lines separate trajectories of different students.

For the random forest model, Figure 9 shows that it is also able to skip the first hint in question 2. For Question 5, the model shows some very strong non-linearity, even decreasing its hint prediction in several instances. These swings appeared for both of the classifiers across all state representations, exaggerated when including the NLP features.

5.3. Validation

The results for the validation data set (found in Appendix B) show that the NLP features indeed add value to the predictions, as was hypothesized. With accuracy rates above 90%,

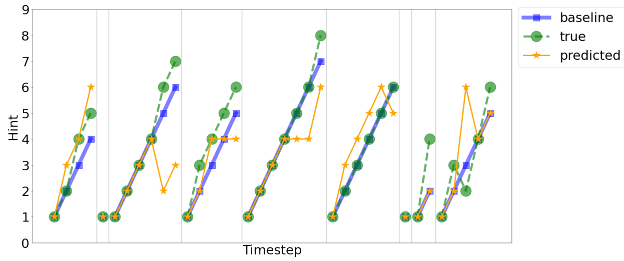

Figure 10. Trajectories of the random forest using the Q+H+M+W state representation for question 2. Vertical lines separate trajectories of different students.

the models clearly capture most of the essential information and predict the hints very well. Further, the classification accuracy of the classifiers using the bag-of-words features peaks at 98%, indicating that the NLP methodology is sufficiently able to use the information provided by the chat data. Additionally, though the decreasing paths of some of the hints seemingly violates their ordinal property, this does not greatly diminish the performance of the regression framework due to the inclusion of NLP characteristics. With the validation data not being collected in the same fashion as real-world data, the results should not be interpreted as an indication that our models perform with a very high accuracy. However, these results show that if there is a strong correlation between the words used by the children and the hint given, the models would be able to pick it up and perform better.

6. Conclusion and Discussion

In general, the above results show that using Imitation Learning show very promising results for automating the process of giving hints and thereby increasing the utility gained in online learning environments and improving learning experience. These results are consistent over different models, where each of them outperform the current industry standard methods represented by the baseline. This holds especially when considering the MAE, as it may be considered less of a risk to deploy these models compared to the baseline.

Figures 5 and 6 suggest that a simple encoding of the state space, such as just using the current sub-question and previous hint given, is already enough for simple models to outperform the baseline. This holds especially for the linear models as they proved that this simple set of features is enough to capture the ordinal structure of the hints. The Figures also suggest that adding information such as demographic data can add some predictive power. This is what one could expect since in general, children of different age and mathematical capability would not need the same hints.

Despite the small amount of conversation data, the feature representation of the best performing model still includes the word2vec representation. This is in compliance with the intuition that using the context from the students should improve predictive performance of their need for instructions. This should motivate further research for using NLP for automatic hint generation. With larger data sets and evolving NLP literature, the process might possibly be completely automated.

In order to implement these automated hint generation models, it is necessary to investigate the performance of automated hints in a time-based framework. The above conclusions assume the timing of the hints to be known, only predicting the exact hint to be provided. This allows for use the real world only if the moments that a hint should be generated are pre-defined. Instead, one could also predict both the time and value of the hint in a two step model. This brings along two challenges. First, it is not immediately clear how to evaluate such a two step model. The question rises about how to weigh between the performance in giving correct hints and the performance in giving them at the right time. Secondly, the sparsity of the data (there is a continues time space but finite number of hints) introduces a problem. It is not trivial to discretize the time in such a way that there are enough opportunities to give hints at the same time as the expert, but limit the sparsity.

The positive results of using Imitation Learning in combination with conversation data are promising but will always be limited by the assumption that the teacher giving hints makes no mistakes. This motivates multiple areas for further research and improvement. For example, it is possible to train a reinforcement learning system optimizing for a reward based on the child's improvements indicated by a difference between the students knowledge of the material (e.g. pre and post geometry test scores) instead of solely imitating the expert. This approach more applicable to the inverse or batch reinforcement learning literature and as such methods are more data-exhaustive, they were not considered for this research. For a larger data set, however, they can provide helpful insights into the automatic hint generation process.

The SmartPrimer project aims to help children keep their intrinsic motivation in learning by using narrative. This research contributes by exploring promising methods of decreasing the labor cost that narrative based learning induces. It was found that using data that is generated during the narrative has potential for automatic hint generation, which in turn is a big step towards making narrative-based learning reality.

References

Gensim. https://radimrehurek.com/gensim/auto_examples/tutorials/

run_word2vec.html#sphx-glr-auto-examples-tutorials-run-word2vec-py. Accessed: 2020-03-18.

Smart primer. https://hci.stanford.edu/research/smartprimer/projects/smartprimer.html. Accessed: 2020-02-25.

Anderman, E. M. and Maehr, M. L. Motivation and schooling in the middle grades. *Review of educational Research*, 64(2):287–309, 1994.

Cen, H., Koedinger, K., and Junker, B. Learning factors analysis–a general method for cognitive model evaluation and improvement. In *International Conference on Intelligent Tutoring Systems*, pp. 164–175. Springer, 2006.

Gottfried, A. E. Academic intrinsic motivation in young elementary school children. *Journal of Educational psychology*, 82(3):525, 1990.

Matsuda, N., Cohen, W. W., and Koedinger, K. R. Teaching the teacher: tutoring simstudent leads to more effective cognitive tutor authoring. *International Journal of Artificial Intelligence in Education*, 25(1):1–34, 2015.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Pane, J. F., Steiner, E. D., Baird, M. D., Hamilton, L. S., and Pane, J. D. How does personalized learning affect student achievement? 2017.

Rivers, K. and Koedinger, K. R. Data-driven hint generation in vast solution spaces: a self-improving python programming tutor. *International Journal of Artificial Intelligence in Education*, 27(1):37–64, 2017.

Stamper, J., Eagle, M., Barnes, T., and Croy, M. Experimental evaluation of automatic hint generation for a logic tutor. *International Journal of Artificial Intelligence in Education*, 22(1-2):3–17, 2013.

A. Complete results

Model \Features	**Q+H**		**Q+H+M**		**Q+H+M+B**		**Q+H+M+W**	
	ACC	**MAE**	**ACC**	**MAE**	**ACC**	**MAE**	**ACC**	**MAE**
Baseline	65%	0.48	65%	0.48	65%	0.48	65%	0.48
Linear regression	72%	0.31	72%	0.31	47%	0.78	75%	0.28
LASSO	73%	0.30	74%	0.29	72%	0.31	74%	0.29
Logistic regression L1	66%	0.48	58%	0.51	54%	0.57	58%	0.52
Random forest	56%	0.49	66%	0.38	62%	0.48	60%	0.55

Table 1. The results for the data set for different models and features/state representations. Q is previous question, H is the previous hint, M is the meta data, B is the Bag of Words representation and W is the word2vec representation.

B. Results validation experiment

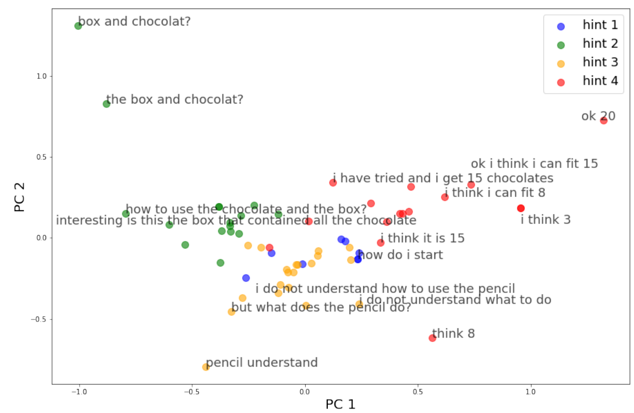

Figure 11. 300 dimensional Word2Vec encoding of the messages sent by the children in the validation data setbefore the labeled hints. The dimensions were reduced to 2 using PCA.

Model \Features	**Q+H**		**Q+H+B**		**Q+H+W**	
	ACC	**MAE**	**ACC**	**MAE**	**ACC**	**MAE**
Baseline	40%	0.80	40%	0.80	40%	0.80
Linear regression	40%	0.6	82%	0.18	52%	0.50
LASSO	40%	0.6	95%	0.05	52%	0.47
Logistic regression L1	27%	0.93	98%	0.02	87%	0.13
Random forest	13%	1.00	98%	0.03	90%	0.18

Table 2. The results for the validation data set for different models and features/state representations. Q is previous question, H is the previous hint, M is the meta data, B is the Bag of Words representation and W is the word2vec representation.

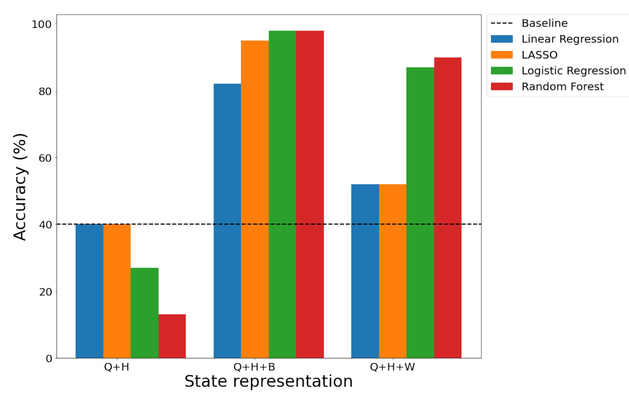

Figure 12. Classification accuracy comparison for the models across different state representations. Q is current sub-question, H is the previous hint given, B and W s are Bag of Words and word2vec representations, respectively. Note that larger values indicate better performance.

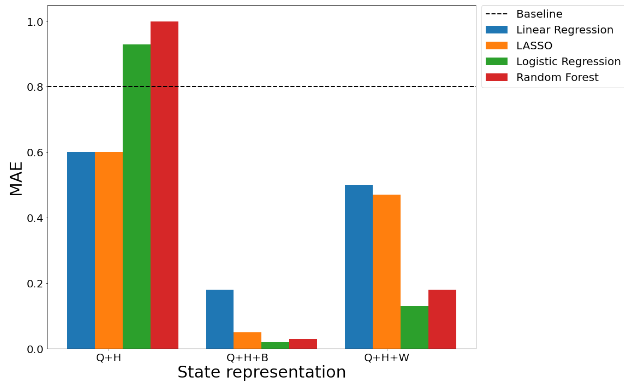

Figure 13. MAE comparison for the models across different state representations. Q is current sub-question, H is the previous hint given, B and W are Bag of Words and word2vec representations respectively. Note that smaller values indicate better performance.

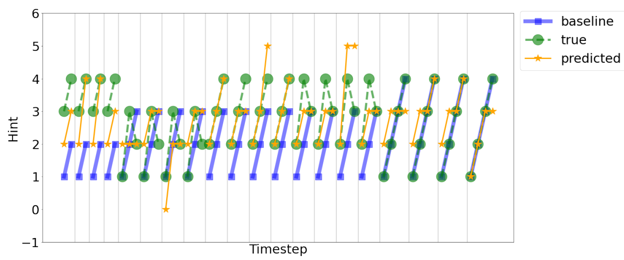

Figure 14. Trajectories of the linear regression with word2vec (Q+H+W model) for Question 2. Vertical lines indicate data for new students

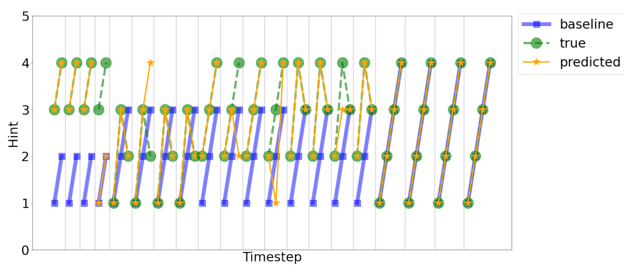

Figure 15. Trajectories of the random forest with word2vec (Q+H+W model) for Question 2. Vertical lines indicate data for new students