Knowledge Tracing: Comparing DNNs Versus GPT For Intelligent Tutoring Systems

Seyed Parsa Neshaei, Adam Hazimeh, and Bojan Lazarevski
Under the supervision of the CHILI Lab (Prof. Pierre Dillenbourg and Dr. Richard Lee Davis), EPFL, CH

***Abstract*—Knowledge tracing (KT) and predicting the next answers of students is considered important in designing intelligent tutoring systems. Previous studies have utilized deep neural networks (DNNs) for this task. However, recent research shows large language models (LLMs) may emerge prediction capabilities by prompting. Previous studies have not compared the performance of DNNs with LLMs on KT. In this project, we evaluate the performance of three types of DNNs for recurrent data, against GPT-3.5, on a set of real-world exercises from a database course. We find a lower performance than DNNs, suggesting limited indications of the LLMs' abilities in KT.**

I. Introduction and Related Work

Intelligent Tutoring Systems (ITS) are digital learning environments including computational models for providing intelligent assistance to students [1], [2]. Previous research in learning sciences has shown positive benefits of integrating ITS in educational environments on learning outcomes [3], [4]. Specifically, *personalized learning* is a topic of focus among the learning sciences literature. Education systems across the world are actively trying to implement personalized learning practices, with the aim of providing quality education for all students [5], [6]. Research shows that students with various learning needs will learn better if supported and instructed tailored to their unique needs [7].

As computer-based tools are used in educational environments increasingly more, a research challenge emerges to effectively track how students learn over time through their online interactions with the ITS. This problem is known as Knowledge Tracing (KT), considered an important element of ITS providing personalized learning [8]. In their simple form, KT models are applied to the question-answering part of an ITS, and predict if a given student would answer the next question correct or wrong, given their previous responses. This serves as an indicator of their *learning* over time by interacting with the ITS. Such models can be integrated into real-world ITS for learning provision and selecting the best question to present to the user in their exercise at any given stage, providing a relevant curriculum of materials to maximize learning gains [8], [9], [10].

Previous works have discussed various approaches to implement KT models[1]. Researchers initially used Bayesian inference approaches and Hidden Markov Models [11], [12], [13] to model students' learning over time. With the rise of deep learning and recurrent neural networks (RNNs), researchers have also applied deep neural networks (DNNs) to the task of knowledge tracing. Notably, Piech et al. [14] employed an RNN and a long short-term memory (LSTM) model to predict if the student would answer a given question correctly at each step. They evaluated their approach, called Deep Knowledge Tracing (DKT), on several datasets consisting of math exercises, and outperform the prior models in terms of model accuracy. Other researchers have extended DKT, e.g. by using newer deep learning architectures, such as attention mechanism or Transformers [15], [16], [17].

Recently, large language models (LLMs), such as the GPT family of models, have been increasingly used to support learners in ITS. They have been used to help students in various curricula and topics, including programming [18], biology [19], math [20], or chemistry [21], among others. Previous research suggests LLMs can provide personalized learning experiences [22]. Notably, in their recent work, Mirchandani et al. [23] show how LLMs as *general pattern machines* can learn *patterns* from data from the input prompts, similar to training a DNN. Thus, there is a gap in the literature on applying LLMs to the task of KT and comparing it with DNNs on a real-world dataset to see if embedding LLMs in ITS is better than DNNs.

In this work, we benchmark LLMs against former approaches on data collected from a CS course on database design. We implement RNN, LSTM, and Transformer architectures, and search for the best hyperparameters and window sizes of input questions, comparing them with the performance of GPT-3.5. We implement downsampling and introduce additional features to improve our model. Our results show LLMs can emerge pattern matching capabilities in our task to a very limited extent, but are severly outperformed by deep networks. We shed light on the applicability of LLMs as general pattern machines in education and contribute to the research line of using LLMs in pedagogical scenarios.

II. Methodology

A. Selecting Our Dataset

To make our findings interesting for the research community of learning sciences and educational technologies as a whole, we specifically look for available open-source datasets. Previous works on KT have utilized various dataset collected from ITS or simulated by computers [24], such as ASSISTments [25], EdNet [26], and Simulated-5 [14],

[1] For a comprehensive survey, see [8].

among others. In this project, we specifically looked for an open-source dataset which A) are in the domain of computer science, thus within the scope and the goal of the project for the CHILI lab, B) are not tiny so that the model can generalize well, C) are not very huge and crossing the limited computational resources we had, and D) are generated by interactions with a real-world tutoring systems as opposed to simulated data. We thus picked the relatively recent *Database Exercise* dataset (`DBE-KT22`) [27], collected from an ITS on a database course in Australia [28].

B. Data Pre-processing

After selecting our dataset, we performed an exploratory analysis on it to find how we can leverage it for KT. The `DBE-KT22` dataset contains logs of 1361 students, answering to 212 unique questions in total. Each student, while learning with the ITS, answers a different set of questions, and can answer the same question more than once as well. Figure 1 shows the distribution of the number of questions answered by students[2]. For each student, the questions they have answered, along with if their answer was right or wrong, are timestamped and recorded in the dataset. Each question is also associated with metadata including an instructor-assigned difficulty level (from 1 to 3) and question topic (e.g., *Tuple*, *Foreign Key*, *Data Integrity*, etc.).

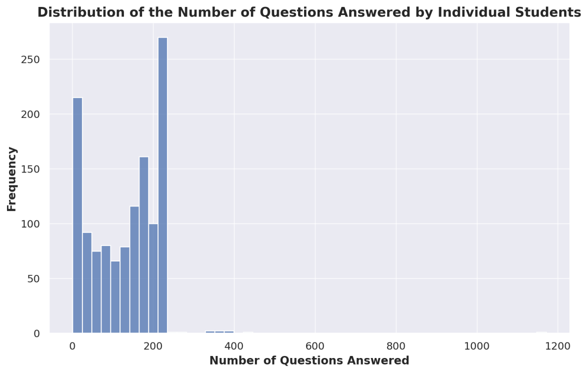

Figure 1. Frequency of number of questions answered by each student.

As also evident from Figure 1, different students have different available input data for obtaining a prediction. For example, while some students have more than 100 questions answered as basis for predicting the next, some have relatively limited input data (e.g. less than 10 questions) for the model to start with. To make our comparison across models and data points fair and be able to incorporate prediction in the middle steps as opposed to only the last question, we splitted the sequences of each student into disjoint sub-sequences of length N. For each sub-sequence of length N, we gave the first $N-1$ elements to the model, and ask it to provide its prediction for the Nth value. We tried different N values of 5, 10, and 20 in our experiments, finding the best number of previous data points to be used to predict the next answer with high accuracy.

We also cleaned our data by removing sub-sequences in which all answers are correct or all are incorrect, which indicate a student already knowing the course material, or not learning anything from the ITS over time, respectively.

As the students using an ITS in general tend to learn the course topic over time, and thus provide fewer incorrect answers as they see new questions, we expected a larger ratio of sub-sequences ending with a *correct* answer, as opposed to *incorrect*. We confirmed this by exploring the sub-sequences, finding 62.81% correct ratio for $N = 5$, 69.96% for $N = 10$, and 71.57% for $N = 20$. As a result, we choose to report *balanced accuracy* [3] instead of normal accuracy for our models, due to the unbalanced nature of our data. We discuss how we try to mitigate the data balance later in the report.

To avoid leaking test data directly or indirectly into our training set, we splitted the data based on students first, and then extracted the subsequences separately in each set. 15% of the data was used for the test set. Among the rest, one third was used for validation and the rest for training[4].

C. Models

We considered four models in total for our comparison, which predict the next possible answer given the previous answers as well as the difficulty level of all questions in the sub-sequence from our dataset.

1) GPT-3.5: As the default GPT-3.5-Turbo model[5] is not available for local inference, we used the API provided by OpenAI through Azure and made available to us by the CHILI lab. We set a relatively low temperature (0.1) to limit the randomness of the model[6]. As we did not do hyperparameter (e.g. temperature) tuning, and use prompting instead of fine-tuning or training GPT-3.5, we only considered our *test* set when evaluating the GPT-3.5 model. We checked for context length violation of any input prompt and discard each entry exceeding the API's limitation.

For our zero-shot prompt, we used a system prompt (see the Appendix for the full prompt) in which we specifically A) began with *"You are a [profession]"*, as also used in previous works [29], [30], [31], [32], B) described the difficulty metric and the output format, C) explicitly asked

[2] Logarithmic version in Figure 5 in the Appendix.

[3] https://scikit-learn.org/stable/modules/model_evaluation.html#balanced-accuracy-score

[4] We did not use cross-validation due to computational resource limitations.

[5] We did not use the state-of-the-art GPT-4 due to budget constraints.

[6] Due to our limited budget, we leave investigating the effects of changing the temperature for future work. Also, the temperature, as well as not necessarily using seeds for all random initializations, can lead to differences in some or all of the accuracies when rerunning the code.

the model to output only one word[7], and D) asked it to learn the pattern of students, triggering the *general pattern machine* aspect of LLMs. In addition to the system prompt, we provided pairs of *user* (in the format of ”Difficulty: [value]”) and *assistant* (either ”CORRECT” or ”WRONG”) prompts as the input to the API, and asked the API to predict the last *assistant* reply.

2) RNN: Previous works have shown RNNs are useful for modeling time series or ordered data [33] and they are already used for KT as well [14], so we also included them as a baseline to compare its performance on our dataset. We chose a hidden layer size of 50 and tuned the number of layers as a hyperparameter[8] with values 1, 2, and 5.

3) LSTM: LSTMs, similar to RNNs on which their original idea is based, are also useful for time sequences data [34]. Additionally, Piech et al. [14] have employed them for KT, so we also applied them to our dataset. We again chose a hidden layer of size 50 and tuned our hyperparameter (number of layers) among 1, 2, and 5.

4) Transformers: Transformers architecture have shown increasing capabilities in pattern matching [35], and thus, we also applied them to our KT dataset. We used a number of heads equal to 5, a 0.1 dropout rate, and the number of expected features (`d_model` or the embedding dimension) equal to 64. We again tuned the number of layers in the same manner.

For all three deep networks (RNN, LSTM, and Transformers), we searched to find the best accuracy on validation set among the different choices of sub-sequence size (N) and layer size. We also tuned the sub-sequence size for GPT-3.5 to find the best accuracy on the validation set. We then picked the best models to see any possible improvement by A) rebalancing the data by downsampling, and B) also including question topic feature as well as difficulty, and report the results. We used the Adam optimizer with an initial learning rate of 0.001.

III. Results

Figures 2, 3, and 4 show the comparison of the performance of the four models we used, in terms of balanced accuracy on validation set, for different number of layers[9]. We pick the balanced test accuracy of the best epoch to account for fluctuations between our 100 epochs.

A. GPT-3.5

The results from inference using our GPT-3.5 prompt on the test set can be seen in Table I. The best model uses $N = 20$ with a test accuracy of 53.63%. We find a very subtle difference among different N values.

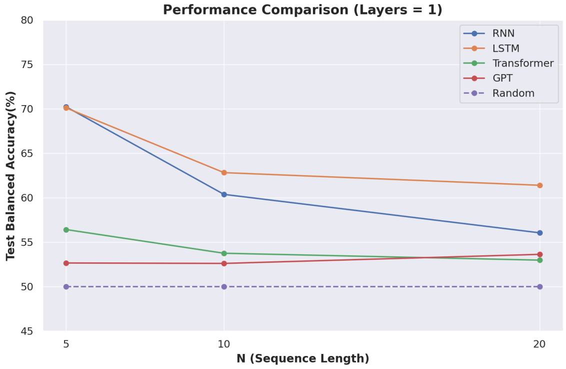

Figure 2. Comparing performance of models with one layer.

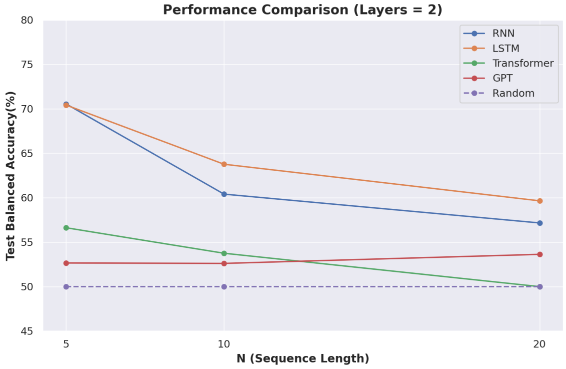

Figure 3. Comparing performance of models with two layers.

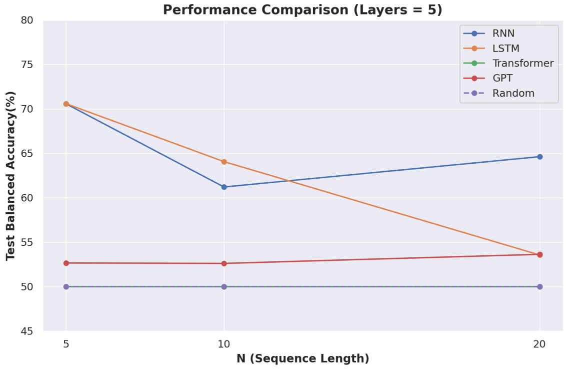

Figure 4. Comparing performance of models with five layers.

Table I
Balanced accuracy with our prompt for GPT-3.5 for different N values

N	Accuracy
5	52.66%
10	52.61%
20	53.63%

[7] In our preliminary tests, we found not explicitly asking the model sometimes leads to additional description provided in the output.

[8] We did not tune the hidden layer size as well, due to computational resource limitations from the lab.

[9] For GPT-3.5, as changing the number of layers is irrelevant, the line is the same among the three figures.

B. Deep Neural Networks

The results of the balanced accuracy on validation set, from our evaluation with RNN, LSTM, and Transformers, can be seen in Tables II, III, and IV respectively. We find the best performing model to be RNN with 5 layers and $N = 5$, leading to a test balanced accuracy of 70.56%.

Table II
Validation set balanced accuracy for different N and number of layers for RNN

N / number of layers	1	2	5
5	69.18%	69.21%	69.36%
10	61.59%	61.18%	61.78%
20	56.59%	57.49%	62.47%

Table III
Validation set balanced accuracy for different N and number of layers for LSTM

N / number of layers	1	2	5
5	69.20%	69.13%	69.15%
10	62.63%	64.32%	62.15%
20	58.85%	60.05%	53.93%

Table IV
Validation set balanced accuracy for different N and number of layers for Transformers

N / number of layers	1	2	5
5	55.91%	55.91%	50.00%
10	53.51%	53.51%	50.00%
20	52.96%	50.00%	50.00%

The results for LSTM and RNN are generally similar to what models in previous works achieved on different datasets and model configuration (e.g. the models from [14]). Surprisingly, Transformers behaved way worse, and closer to the random baseline of 50.00% (considering prior distribution), than the other two models on our downstream task.

Contrary to the deep models, the best performing N for GPT-3.5 is 20, suggesting that longer, more-detailed prompts may lead to a higher accuracy in GPT-3.5 when no training data from our downstream task is given to the model. The GPT-3.5 model scores lower than all deep models, suggesting that the abilities of GPT-3.5 as a *general pattern machine* in our specific task are severely limited.

C. Downsampling

We find our models tend to choose answers mostly as *correct*, possibly due to the imbalanced data. For example, for the best deep model (RNN for 5 layers and $N = 5$), we find a testing F1 score of 80.49% on the *correct* class, but a lower score of 61.99% on the *wrong* class. The situation for best GPT-3.5 run (with $N = 20$) is worse; while we find an F1 score of 82.16% for the *correct* class (way above those of the deep models), we only obtain an F1 score of 19.79% on the *incorrect* class, suggesting that GPT-3.5 tends to respond *correct* significantly more.

To test if we can improve the accuracy of our best models further, we downsample the *correct* class using a random sampling to make it the same size of the *incorrect* class in our training data. As we do not utilize the training set for GPT-3.5 and use prompting instead, we only apply downsampling on the best deep model (RNN, 5 layers, $N = 5$). The results show a very subtle effect on the results (from 70.56% to 70.67%), so we find downsampling is not useful for our best model[10].

D. Question Topics

The dataset we used (DBE-KT22) also comes with the instructor-assigned topics of questions in the data. To further improve the accuracy of our models, we also embed topic IDs in our models (in the prompt for GPT-3.5 as provided in the Appendix, and as input to the best RNN). Interestingly and contrary to our expectations, we achieve a very subtle increase for LSTM. but a decrease for GPT-3.5 scoring even worse than the random baseline (see Table V). This suggests that the topic data is acting as a noise, and difficulty alone is a better indicator of whether the student would answer the database questions in our specific context right or wrong.

Table V
Balanced accuracy of the best-performing models, before and after including question topics.

Model	Test Accuracy Without Topics	Test Accuracy With Topics
LSTM (5 layers, N = 5)	70.56%	71.74%
GPT-3.5 (N = 20)	53.63%	49.57%

IV. Conclusion

In this project, we assessed how LLMs can do knowledge tracing compared to deep models on a recent CS-related dataset, and act as *general pattern machines* in this task. We find the deep models consistently achieve higher accuracies than LLMs, further improved by adding question topic and downsampling. Our results shed light on limitations of applying LLMs as a golden hammer to any machine learning problem without first considering how deep models perform, and help in choosing best models to incorporate in real-world CS ITS.

We call for future researchers to also incorporate more ablations, such as adjusting the temperature of GPT-3.5, more prompting strategies (e.g. few-shot), cross-validation, incorporating more features or rebalancing methods, and using other LLMs (e.g. GPT-4 or LLaMA), which we did not consider due to the lab's resource and budget constraint for our project.

[10] We leave other rebalancing methods for future work.

V. Ethical Risk Assessment

Our relatively low model accuracies causes numerous wrong predictions. To find how our approach can harm the stakeholders, we examined the F1 scores of each class (see the *Downsampling* subsection) and find it is notably lower for the *incorrect* class. This suggests a high tendency to think the user learned the material, although they may have not. This can harm students' learning by, e.g., delivering learning prematurely in Intelligent Tutoring Systems (ITS) to students when they are not ready, or misleading instructors about students' grasp of the course material. This issue is more severe and likely in the less-accurate GPT-based models, making them unsuitable for real-world contexts, but also happens commonly in DNNs.

This risk underscores the need for *confidence* measures alongside binary predictions. Confidence scores, approximated by normalizing the raw outputs of our models, provide a measure of reliability for each prediction, helping people evaluate the model's credibility. By using the *sigmoid* function to normalize our models' outputs (instead of applying a binary threshold of 0.5), we gain an understanding of the trustworthiness of their predictions. For instance, we can see cases where our best DNN model is overconfident (confidence $\geq$ 0.9; incorrect), underconfident (confidence $\leq$ 0.1; correct), and balanced, in its predictions below:

True Label: 1
Prediction: 1
Confidence: 99.991453%
Verdict: Balanced

True Label: 0
Prediction: 0
Confidence: 0.001774%
Verdict: Underconfident

True Label: 0
Prediction: 1
Confidence: 99.971777%
Verdict: Overconfident

It is important to note that providing confidence scores, as a positive step towards understanding when the models are erroneous, is a simple approximation, not a definitive accuracy measure (uncertainty estimation in deep networks is a popular, ongoing research area [36]). Also, this risk is not incorporated into our project, because we only provide the models without the accompanying ITS. We leave the fair integration of the models to the maintainers of ITS using our models.

Finally, the potential for error in ITS using our models must be clearly communicated to users. This awareness ensures that our models are only used as educational aids, rather than complete predictors of student capabilities. We specifically advise that A) ITS should not rely on our models to severely change educational content, and B) model outputs should always be accompanied with clear description and confidence scores, and discarded when not confident[11].

Appendix

Full System Prompt for GPT-3.5

You are an instructor and want to trace how the student has learned to answer the questions over time. Each time, the user gives you the difficulty of a question (an integer ranging from 1 [easiest] to 3 [hardest] as estimated by the instructor), and you should output a single word: CORRECT if you think the student would answer the question correctly, and WRONG if you think the student would answer the question wrong. Output no other word at all, this is very important. Try to learn the pattern of the student over time and how they improve their knowledge of the course.

Full System Prompt for GPT-3.5 (With Question Topics)

You are an instructor and want to trace how the student has learned to answer the questions over time. Each time, the user gives you the difficulty of a question (an integer ranging from 1 [easiest] to 3 [hardest] as estimated by the instructor) as well as the topic of the question (as an ID indicating the topic, as specified by the instructor), and you should output a single word: CORRECT if you think the student would answer the question correctly, and WRONG if you think the student would answer the question wrong. Output no other word at all, this is very important. Try to learn the pattern of the student over time and how they improve their knowledge of the course.

References

[1] A. C. Graesser, M. W. Conley, and A. Olney, "Intelligent tutoring systems." 2012, publisher: American Psychological Association.

[2] J. R. Anderson, C. F. Boyle, and B. J. Reiser, "Intelligent tutoring systems," *Science (New York, N.Y.)*, vol. 228, no. 4698, pp. 456–462, 1985, publisher: American Association for the Advancement of Science.

[3] J. A. Kulik and J. Fletcher, "Effectiveness of intelligent tutoring systems: a meta-analytic review," *Review of educational research*, vol. 86, no. 1, pp. 42–78, 2016, publisher: Sage Publications Sage CA: Los Angeles, CA.

[11] We provide our *digital ethics canvas* in the root of our GitHub repository with the name `DEC.pdf`. As we did not collect the dataset we used ourselves, we only filled the *software design* page of the canvas, similar to the examples provided on the website (https://www.epfl.ch/education/educational-initiatives/cede/training-and-support/digital-ethics/a-visual-tool-for-assessing-ethical-risks/examples/). Among the risks, we chose the *(non-)maleficence* section to present in details in the current report.

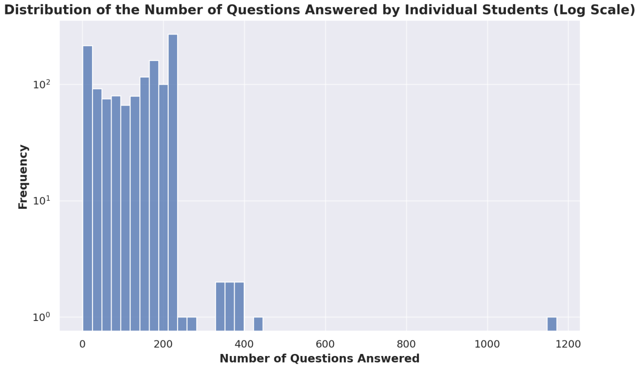

Figure 5. Frequency of number of questions answered by each student in a logarithmic scale.

[4] G. Cheng, "The impact of online automated feedback on students' reflective journal writing in an EFL course," *The Internet and Higher Education*, vol. 34, pp. 18–27, 2017, publisher: Elsevier.

[5] A. Peterson, "Personalizing education at scale: Learning from international system strategies," *The Education Redesign Lab*, pp. 1–135, 2016.

[6] L. Zhang, J. D. Basham, and S. Yang, "Understanding the implementation of personalized learning: A research synthesis," *Educational Research Review*, vol. 31, p. 100339, 2020, publisher: Elsevier.

[7] L. Jones and M. Casey, "Personalized learning: Policy & practice recommendations for meeting the needs of students with disabilities," *National Center for Learning Disabilities. Retrieved from http://www. ncld. org/wp-content/uploads/2016/04/Personalized-Learning. WebReady. pdf*, 2015.

[8] G. Abdelrahman, Q. Wang, and B. Nunes, "Knowledge tracing: A survey," *ACM Computing Surveys*, vol. 55, no. 11, pp. 1–37, 2023, publisher: ACM New York, NY.

[9] A. T. Corbett and J. R. Anderson, "Knowledge tracing: Modeling the acquisition of procedural knowledge," *User modeling and user-adapted interaction*, vol. 4, pp. 253–278, 1994, publisher: Springer.

[10] T. Schodde, K. Bergmann, and S. Kopp, "Adaptive robot language tutoring based on Bayesian knowledge tracing and predictive decision-making," in *Proceedings of the 2017 ACM/IEEE international conference on human-robot interaction*, 2017, pp. 128–136.

[11] R. S. d. Baker, A. T. Corbett, and V. Aleven, "More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing," in *Intelligent tutoring systems: 9th international conference, ITS 2008, montreal, canada, june 23-27, 2008 proceedings 9*. Springer, 2008, pp. 406–415.

[12] M. Villano, "Probabilistic student models: Bayesian belief networks and knowledge space theory," in *Intelligent tutoring systems: Second international conference, ITS'92 montréal, canada, june 10–12 1992 proceedings 2*. Springer, 1992, pp. 491–498.

[13] T. Käser, S. Klingler, A. G. Schwing, and M. Gross, "Dynamic Bayesian networks for student modeling," *IEEE Transactions on Learning Technologies*, vol. 10, no. 4, pp. 450–462, 2017, publisher: IEEE.

[14] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein, "Deep knowledge tracing," *Advances in neural information processing systems*, vol. 28, 2015.

[15] S. Pandey and G. Karypis, "A self-attentive model for knowledge tracing," *arXiv preprint arXiv:1907.06837*, 2019.

[16] S. Pandey and J. Srivastava, "RKT: relation-aware self-attention for knowledge tracing," in *Proceedings of the 29th ACM international conference on information & knowledge management*, 2020, pp. 1205–1214.

[17] A. Ghosh, N. Heffernan, and A. S. Lan, "Context-aware attentive knowledge tracing," in *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 2020, pp. 2330–2339.

[18] C. Cao, "Scaffolding CS1 courses with a large language model-powered intelligent tutoring system," in *Companion proceedings of the 28th international conference on intelligent user interfaces*, 2023, pp. 229–232.

[19] A. I. P. Sanpablo, "Development and evaluation of a diagnostic exam for undergraduate biomedical engineering students using GPT language model-based virtual agents," in *XLVI mexican conference on biomedical engineering: Proceedings of CNIB 2023, november 2–4, 2023, villahermosa tabasco, méxico-volume 1: Signal processing and bioinformatics*, vol. 96. Springer Nature, 2023, p. 128.

[20] M. Zong and B. Krishnamachari, "Solving math word problems concerning systems of equations with gpt-3," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, 2023, pp. 15 972–15 979, issue: 13.

[21] A. D. White, G. M. Hocky, H. A. Gandhi, M. Ansari, S. Cox, G. P. Wellawatte, S. Sasmal, Z. Yang, K. Liu, Y. Singh, and others, "Assessment of chemistry knowledge in large language models that generate code," *Digital Discovery*, vol. 2, no. 2, pp. 368–376, 2023, publisher: Royal Society of Chemistry.

[22] X. Su, T. Wambsganss, R. Rietsche, S. P. Neshaei, and T. Käser, "Reviewriter: AI-generated instructions for peer review writing," in *Proceedings of the 18th workshop on innovative use of NLP for building educational applications (BEA 2023)*, 2023, pp. 57–71.

[23] S. Mirchandani, F. Xia, P. Florence, B. Ichter, D. Driess, M. G. Arenas, K. Rao, D. Sadigh, and A. Zeng, "Large language models as general pattern machines," *arXiv preprint arXiv:2307.04721*, 2023.

[24] G. Casalino, L. Grilli, P. Limone, D. Santoro, D. Schicchi, and others, “Deep learning for knowledge tracing in learning analytics: an overview.” *TeleXbe*, 2021.

[25] L. Razzaq, M. Feng, G. Nuzzo-Jones, N. T. Heffernan, K. Koedinger, B. Junker, S. Ritter, A. Knight, C. Aniszczyk, S. Choksey, and others, “The Assistment project: Blending assessment and assisting,” in *Proceedings of the 12th annual conference on artificial intelligence in education*, 2005, pp. 555–562.

[26] Y. Choi, Y. Lee, D. Shin, J. Cho, S. Park, S. Lee, J. Baek, C. Bae, B. Kim, and J. Heo, “Ednet: A large-scale hierarchical dataset in education,” in *Artificial intelligence in education: 21st international conference, AIED 2020, ifrane, morocco, july 6–10, 2020, proceedings, part II 21*. Springer, 2020, pp. 69–73.

[27] G. Abdelrahman, S. Abdelfattah, Q. Wang, and Y. Lin, “Database exercises for knowledge tracing (DBE-KT22),” 2022, tex.entrytype: data. [Online]. Available: http://dx.doi.org/10.26193/6DZWOH

[28] ——, “DBE-KT22: A knowledge tracing dataset based on online student evaluation,” *arXiv preprint arXiv:2208.12651*, 2022.

[29] Y. Bao, K. P. Yu, Y. Zhang, S. Storks, I. Bar-Yossef, A. De La Iglesia, M. Su, X. L. Zheng, and J. Chai, “Can foundation models watch, talk and guide you step by step to make a cake?” *arXiv preprint arXiv:2311.00738*, 2023.

[30] E. R. Mollick and L. Mollick, “Using AI to implement effective teaching strategies in classrooms: Five strategies, including prompts,” *Including Prompts (March 17, 2023)*, 2023.

[31] A. G. Møller, J. A. Dalsgaard, A. Pera, and L. M. Aiello, “Is a prompt and a few samples all you need? Using GPT-4 for data augmentation in low-resource classification tasks,” *arXiv preprint arXiv:2304.13861*, 2023.

[32] S. J. Krol, M. T. Llano, and J. McCormack, “Towards the generation of musical explanations with GPT-3,” in *International conference on computational intelligence in music, sound, art and design (part of EvoStar)*. Springer, 2022, pp. 131–147.

[33] L. R. Medsker and L. Jain, “Recurrent neural networks,” *Design and Applications*, vol. 5, no. 64-67, p. 2, 2001.

[34] S. Siami-Namini, N. Tavakoli, and A. S. Namin, “The performance of LSTM and BiLSTM in forecasting time series,” in *2019 IEEE International conference on big data (Big Data)*. IEEE, 2019, pp. 3285–3292.

[35] S. Ahmed, I. E. Nielsen, A. Tripathi, S. Siddiqui, R. P. Ramachandran, and G. Rasool, “Transformers in time-series analysis: A tutorial,” *Circuits, Systems, and Signal Processing*, vol. 42, no. 12, pp. 7433–7466, 2023, publisher: Springer.

[36] J. Gawlikowski, C. R. N. Tassi, M. Ali, J. Lee, M. Humt, J. Feng, A. Kruspe, R. Triebel, P. Jung, R. Roscher, and others, “A survey of uncertainty in deep neural networks,” *Artificial Intelligence Review*, vol. 56, no. Suppl 1, pp. 1513–1589, 2023, publisher: Springer.