Voelker, Adrian; Thaler, Anna Magdalena; Summerer, Maximilian T.; Schmid, Ute

Large Language Models as Domain-independent Dialogue Component for Intelligent Tutoring Systems – Teaching Concepts of SQL

In:

Schmid, Ute; Leidner, Jochen L.; Kohlhase, Michael; Wolter, Diedrich (Eds.), Proceedings of the Second Work shop on Artificial Intelligence for Artificial Intelligence Education (AI4AI Learning 2024), Bamberg: University of Bamberg Press, p. 33-48. 2025. DOI: 10.20378/irb-107661

Bookpart - Published Version

DOI of the Article: 10.20378/irb-108886

Date of Publication: 07.07.2025

Large Language Models as Domain-independent Dialogue Component for Intelligent Tutoring Systems – Teaching Concepts of SQL[1]

Adrian Voelker, Anna M. Thaler, Maximilian T. Summerer, and Ute Schmid

University of Bamberg, Germany

{adrian.voelker, anna.thaler, ute.schmid}@uni-bamberg.de
maximilian-timon.summerer@stud.uni-bamberg.de

Abstract

With the advance of generative pre-trained language models, new opportunities arise for domain-independent dialogue-based instruction in the context of intelligent tutoring systems (ITS). We propose an approach for combining large language models (LLMs) with domain-specific knowledge graphs to take advantage of the performance of LLMs in natural language processing tasks while still ensuring a faithful knowledge base. We present a prototypical implementation of an ITS for the structured query language SQL to explore the potential of combining LLM and domain specific knowledge graphs for dialogue based didactic intervention. While learning a programming language relies on both declarative knowledge about the language concepts and procedural knowledge for writing program code with respect to a specific task, in this paper we focus on ITS support for acquisition of declarative knowledge. In our implementation LLMs are employed for semantic parsing relating student's questions to the addressed knowledge elements of the domain model as

[1] The work reported in this paper, has been supported by Bund-Länder Förderinitiative "Künstliche Intelligenz in der Hochschulbildung", project "VoLL-KI: Data-based analyses, explanations and aids", project number 16DHBKI091.

well as for feedback formulation based on retrieved information from the knowledge graph. Likewise, knowledge diagnosis is realized by semantic parsing of student answers and mapping them against the LLM output. We conducted a first exploratory evaluation using the LLM Mixtral 8x7B Instruct in combination with our SQL knowledge graph. Exploring different prompting strategies, the best strategy resulted in 90% correct diagnosis of student answers.

Keywords: Intelligent Tutoring Systems, Large Language Models, Knowledge Graphs, SQL, Declarative Knowledge

Introduction

Programming education has been one of the most implemented domains in intelligent tutoring systems (ITS) [11]. Over time many different systems for multiple programming languages [5], such as LISP [2], Python [18], and the Structured Query Language (SQL) [10] were developed. SQL in particular is a widely used language for efficient data retrieval and manipulation in many business processes that is relevant far beyond the field of computer science and therefore of interest to a broad audience from computer science undergraduate students to casual data-oriented users.

Learning a programming language relies on both declarative knowledge about the language concepts and procedural knowledge for writing program code with respect to a specific task. Typically, ITS for programming have a strong focus on procedural knowledge [13]. However, declarative knowledge about programming concepts and their relationships is not only relevant as a knowledge base for the acquisition of programming skills in the given domain, it also substantially enhances the transfer between tasks of different sub-skills (such as programming versus understanding a program) that share the same declarative knowledge base [17]. Therefore, declarative knowledge about concepts and the relationships between them can play a crucial part in supposedly purely procedural tasks. Errors in programming tasks could occur due to existing misconceptions while performing the (procedural) task or due to missing declarative knowledge about the underlying concepts necessary to correctly solve it.

Thus, we encourage to consider conceptual knowledge within ITSs for programming. In this paper, we present our current work on the declarative part of an ITS for learning the programming language SQL, which we afterwards combine with procedural knowledge tasks for a flexible switching between the two knowledge types and thus a more accurate student knowledge diagnosis.

One major restriction in the development of ITSs is that the architecture is focused on specific teaching domain and the realized teaching strategies. Typically, ITS require a large number of resources for their development [13]. An often-laborious part of the implementation concerns the domain-specific methods of natural language processing (NLP) for a natural dialogue as form of interaction with the students [6]. The latest developments in large language model (LLM) research promise great potential to move towards a domain-independent method to solve NLP tasks. The development effort for parts of the tutoring and communication module could, therefore, be significantly lowered as a result.

In this paper, we present a prototypical implementation of an ITS for SQL to explore the potential of combining LLM and domain specific knowledge graphs for dialogue based didactic intervention to enable students asking questions while also enabling the ITS with the capability to check and correct given answers. Hereby, we want the LLM to assist in the identification of distinct question-types task and evaluate a given answer. The used LLM is fine-tuned for the first task and prompting techniques are used for the latter. In the following, we first discuss related work, and then describe relevant system components. We show results of a preliminary evaluation exploring different prompting strategies. We conclude with an outlook towards the next steps of extending the implementation and a more comprehensive evaluation.

Related Work

The use of LLMs has been investigated in the context of annotating student actions to corresponding tutoring actions as a way to reduce the resources required for annotations by pedagogic experts [20]. However, the

LLM's helpfulness of feedback in particular has been found to be quantifiably worse compared to human teachers [19]. First attempts to evaluate the effectiveness of LLMs as a sole pedagogical instructor compared to a human teacher have been made [19], but standardized and comprehensive criteria for the evaluation of conversational agents within intelligent teaching are still missing.

Knowledge-based ITS for declarative tasks often use a knowledge graph (KG) as their knowledge base. While a KGs' explicit knowledge representation ensures high precision inference and reasoning [16], LLMs may produce misleading, untrustworthy or incorrect responses, such as hallucinations [7], due to their probabilistic nature [14]. Their output could amplify biases from their training data and they lack in the ability to handle numerical values [14]. Especially when interpretability and explainability is required - which is the case for most ITS - KGs can explicitly represent the structure and relationships between entities [14] and therefore form the basis for the tutoring module's feedback. A sensitive topic regarding the education of students is misinformation. As interacting students are still learning a subject, misinformation might be harder to detect and thus can have a higher negative impact on the learning process compared to experts using LLMs in the domain of expertise. In order to reduce the risk for non-trustworthy responses, we argue that relying on an LLM's parametric knowledge alone is currently not suitable as a knowledge base for ITS.

This might lead to the question of why we want to incorporate LLMs in ITS in the first place. The biggest advantage of pre-trained language models lies in their great performance in the field of language processing, general knowledge and generalizability [16]. Neuro-symbolic methods, therefore, seem to be a promising approach to simplify the development process for ITS. In recent literature were three frameworks for unifying KGs and LLMs defined. KGs can enhance LLMs and could e.g. be used to detect LLMs' hallucinations [14,16] and to improve the reasoning capabilities of a LLM for complex logical problems [15]. LLMs could augment KGs with their natural language processing ability, which we currently view to be most promising for the domain of education. The third framework in their roadmap lies in a full synergy of both methods.

Similarly, Bianchini et al. (2024) argue that to adequately (re)solve complex linguistic tasks, what we herein consider a dialog during a didactic intervention of the ITS or the parsing and contextualizing of questions towards the Expert System, a knowledge graph is essential [3]. We further on share the idea, that solely combination of LLMs with Retrieval Augmented Generation (RAG) on diverse knowledge sources such as a documentation for SQL, is not sufficient for an expert system, since eventually the enriched LLM component leads to hallucinatory output, even though a knowledge base was supplied [3].

A hybrid representation of both parametric knowledge (LLMs) and explicit knowledge in the form of KGs, is already being discussed in literature [14,3]. This combination, however, still requires further research and evaluation and more contextualization towards ITS for SQL. Therefore, we first want to explore how LLMs can enrich a knowledge-based tutoring system by simplifying a natural language dialogue and answer validation without relying solely on LLMs and RAG as knowledge base.

System Components

In the following section we highlight the fundamental system components of our ITS. In this paper we want to especially focus on the aforementioned declarative knowledge. Since we consider also procedural knowledge of SQL as a indispensable requirement, we plan our system to be integrable with further components, that are not highlighted in more detail in this paper.

3.1 The Declarative Domain Knowledge Base for SQL

As explicit and declarative knowledge base we employ a semantic network (a KG) on basic SQL concepts based on previous work by Zhou [22]. Zhou identified two types of declarative knowledge within SQL: the natural and the abstract form representation. The natural form includes at least one property about the concept in natural language, e.g. a description about the function of the concept, whereas the abstract form describes the relationship between concepts [22].

Labelled property graphs are highly suitable for representing both forms in one consistent knowledge representation. Each node represents a concept, e.g. the select statement, and the edges between the nodes represent the relationship between concepts, e.g. "has a", "type of", "kind of", "part of" or "set of". Using domain-independent expressions for the edge labels builds the foundation for a domain-independent interaction with the LLM as intermediate component (see Figure 1). We expect that theses generic labels allow using the same inference mechanisms (queries to the graph database within the domain module) and the same prompts for the LLM, even if the entire domain (the graph database) changes. Each node (concept) holds the properties relevant for the comprehension of the concept, e.g. a description of the function, the correct notation, or allowed input (if applicable). These properties can be used for the question generation for a single concept as well as for the comparison between concepts.

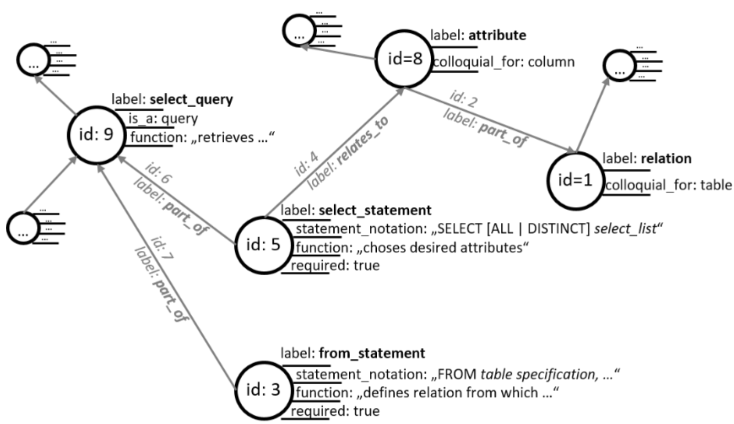

Fig.1. Excerpt and simplified representation of a possible semantic network for SQL knowledge as basis for domain-unspecific inference mechanisms.

3.2 Possible Application Points for LLM

We aim at combining LLMs and KGs as part of an ITS within their scope of expertise. LLMs offer great potential for a nearly effortless built, domain independent and dynamic natural language dialogue, but we do not rely on its parametric, possibly incorrect knowledge. KGs, in contrast, ensure correct inference mechanisms as well as domain-specific and explicit knowledge that provides the base for explainability and feedback formulation. Possible starting points that we see for the integration of LLMs within a simplified version of an ITS architecture are shown in Figure 2. The integration of LLMs into ITS can lead to a further separation of domain-specific modules such as the knowledge representations in the domain or student module from domain-independent natural language processing mechanisms. These could include but are certainly not limited to the extraction of intentions by the student, the answer validation (diagnostic comparison operation between expected and given solution), and the student interaction such as the formulation of feedback or follow-up questions. In the following, we want to outline these mechanisms as approaches for these integration points in the way we are currently implementing them into our system for intelligent SQL tutoring.

1. ***Student Input Translation:*** All queries to the semantic network (labelled property graph) are mediated by controlling components of the ITS. There are approaches that aim to translate natural language directly to a query [14], however, we want to ensure that the inference mechanisms on the KG are considered correct and we – the developer – know the attributes of the semantic network best. Therefore, when a student asks a question in natural language to the ITS, that question is not directly translated to a query (in our case CYPHER within Neo4J [12]), but the LLM is used to translate the student's question to categories of questions (e.g. concerning properties of concepts, or relationships between concepts). By retrieving the essence of a student's question with the help of a LLM, we can formulate our query to the graph database and, therefore, provide higher independent components by separating the dialogue from accessing the KG itself. This is a key point for the transfer of the same methods to other domains as knowledge base.

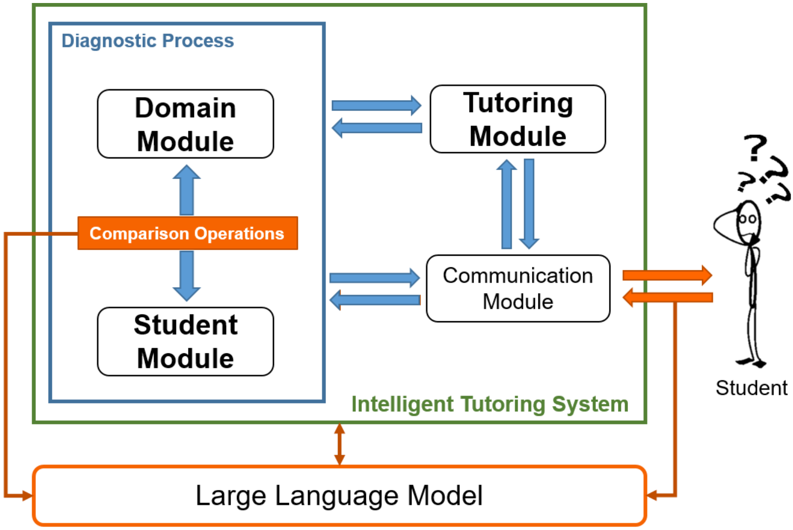

Fig.2. Classic Four-Component Architecture of ITS with possible integration points for LLM marked in orange.

2. ***Answer Validation as Comparison Operation:*** The LLM also offers a flexible way to overcome classical steps of other NLP methods such as the equivalence of specific terms in the sense of grammar or spelling mistakes, or synonyms, but also domain-specifically equivalent terms. In the domain of SQL, for example, students might refer to a relation as table or to a tuple as row in a table. The task of the LLM in this case is to compare the expected answer from the labeled property graph (KG) with the answer given by the student. This way, the lack of domain-specific knowledge of the LLM does not come into play, as the explicit and faithful knowledge from the KG is used for reasoning and serves as expected expert response to compare to the student's reply. The LLM's generated comparison can be used to update the student model accordingly and serves as starting point for analysis when the answers are not found to be completely the same.

3. ***Feedback Formulation:*** Another integral point lies in the formulation of ITS information to natural language for easier understanding by the student. This can be applied for domain-specific explanations based on the KG's data within the domain module, but also didactic strategies within the tutoring module, the LLM is prompted to simply generate natural language feedback from a more restricted and formalized form from within the ITS to present it to the student.

The information from the student model about the current state of comprehension can be used within the prompt to adapt the complexity of the natural language feedback to the student's current diagnosed knowledge.

3.3 Didactic Intervention

The KG explicitly stores data about the concepts and the concepts' relationship and therefore also holds information about their similarities. This constellation can be exploited to create appropriate didactic interventions, e.g. a dialogue following the Socratic instructional strategy as a form of scaffolding [1]. Given the student has already studied (parts of) the subject, when an incorrect answer is given to one concept, the student will be asked similar questions about a similar concept. When knowledge for this similar concept is available, it can serve as a starting point for the student to decode and recall information about the originally disremembered concept. For example, if a student does not recall the description of the aggregate function 'SUM', one possible similar concept for the Socratic questioning might be to ask what 'COUNT' does. This not only enhances memory recall to evoke already acquired but currently not accessible knowledge to the student but also serves as assessment for the student knowledge model. The similarity scores between concepts can be built upon several factors including hierarchical structures such as super- or subconcepts (based on closeness of concepts within the KG structure), but also the concept's properties.

Preliminary Evaluation

We tested and evaluated language model integration in the first two areas described in Sect. 3.2. When users ask questions to retrieve information, a language model is used to match their question to pre-defined question categories:

- **Property question.** (Example: What is %s's %s?)
- **Relation question.** (Example: What is the relation between %s and %s?)
- **Relatives question.** Example: To what node(s) is %s connected via %s relation?)
- **Similarity question.** (Example: What similarities do %s and %s have?)
- **Difference question.** (Example: What are the differences between %s and %s regarding %s?)

Furthermore, when interrogating a user, a language model is used to compare the student answer with the correct solution provided by the KG.

Language Model for Question Category Classification

An attractive application of a language model is as a mediator between user question and information retrieval from the KG. In our ITS structure knowledge is retrieved via predefined question categories, ensuring that no ill-formulated queries are forwarded to the KG. However, determining the most suitable question category for a user question is not trivial. Users can use a variety of formulations for the exact same question. Here, a language model can be used to 'translate' the user's question into one of the pre-defined question categories. Since general language models did not deliver good results in this context, an instance of TinyLlama [21] was fine-tuned on a dataset containing 250 entries with possible user questions and their corresponding question categories. The resulting model proved useful for cases where the user question was either similar to the formulation of the question category itself or similar to the entries in the dataset. However, the more a user question diverged from the expected result, the less frequently the language model delivered the correct match. To address this problem, a different, better performing language model could be used as a basis, a more extensive and more diverse dataset could be used to fine-tune this model. Furthermore, constraining the model to a grammar by utilizing the Backus–Naur form (BNF) has not shown useful results so far. However, this could be further explored.

Language Model for Student Answer Validation.

For the ITS to function effectively, it is crucial to evaluate the correctness of students' answers. While the KG provides only one correct answer, students may express the correct solution using formulations that are completely different from the stored information. This variability makes it difficult for the ITS to determine whether the student answered correctly. To address this issue, using a language model appears promising, as it could grasp the semantic meaning of a user's answer, compare it to the solution provided by the KG, and decide whether or to what degree the student answered correctly. For an initial implementation of this feature, an instance of the Mixtral 8x7B Instruct Model [9,8] was used. A sample dataset was created, comprising ten distinct questions. For each question, the dataset contains three possible user answers: one clearly correct, one clearly incorrect, and one either correct or incorrect but not immediately recognizable. This setup results in a total of 30 entries, with 15 to be classified as correct and 15 to be classified as incorrect (see Table 1 for exemplary questions). Four different prompts were compared: Two longer prompts with examples of the expected output and two prompts containing only a brief instruction of the task.

Remarkably the prompt with a brief instruction using a courtesy form (see Figure 4) was leading to better results than longer and more specific prompts (see Figure 4)). In the first evaluation, the prompt in Figure (4) without examples resulted in an accuracy of 90% on the described dataset. In the future, different prompts, system prompts, and fine-tuned language model could be tested on more extensive datasets, preferably obtained from actual user answers.

Limitations

These implementations still come with several limitations. So far, only a handful of language models have been evaluated, and the data on which they were fine-tuned and evaluated is still limited. Additionally, no data has been collected from actual user-system interactions. Furthermore, it is still unclear whether the use of question categories is the best way to mediate between user questions and the KG.

Table 1. Exemplary test data for evaluating the match between LLM and KG category.

Tutor question	LLM match to KG	Student answer	classification
What is the relation between equal and comparison predicate?	TYPE_OF	type of	correct
To what node(s) is equal connected via TYPE_OF relation?	comparison predicate	to comparison predicate	correct
To what node(s) is relation connected via HAS_A relation?	relation name, cardinality, degree, domain, PK, FK	to relation name, degree, cardinality, PK, FK, and domain	correct
To what node(s) is equal connected via TYPE_OF relation?	comparison predicate	it is a type of aggregate function	incorrect
To what node(s) is equal connected via TYPE_OF relation?	comparison predicate	select	incorrect
...	...	...	...

“Please decide whether the student answer is correct, based on the correct solution.”

Fig.3. Best performing prompt with the Mixtral 8x7B Instruct Model

Your task is to determine whether a student answer is correct in relation to the solution and the question.

####
Here are some examples:

Question : To what node(s) is avg connected via TYPE_OF relation ?;
Correct Solution: aggregate function;
Student answer: average is a type of aggregate function;
Classification: correct

Question: What is the relation between where and group by?;
Correct Solution: where–>COMES_BEFORE–>group by ;
Student answer: group by comes before where;
Classification: incorrect

Question: What is the relation between where and group by?;
Correct Solution: where–COMES_BEFORE–>group by;
Student answer: where comes before group_by ;
Classification: correct

Question: What is the description of avg ?;
Correct Solution: Calculates the average value of a numeric column;
Student answer: avg calculates the average of the columns ;
Classification: correct

Question: What is the description of avg ?;
Correct Solution: Calculates the average value of a numeric column;
Student answer: avg tells me more about the average time a query takes to execute;
Classification: incorrect

Question: What is the relation between sum and aggregate function?;
Correct Solution: sum–TYPE_OF–>aggregate function ;
Student Answer: sum is a type of aggregate function;
Classification: correct

Question: What is the relation between sum and aggregate function?;
Correct Solution: sum–TYPE_OF–>aggregate function ;
Student Answer: sum is equivalent to aggregate function;
Classification: incorrect

####

Fig.4. Less performing prompt with the Mixtral 8x7B Instruct Model

Discussion and Further Work

In this paper we show that LLMs can be combined with labelled property graphs to enable teaching of declarative knowledge for programming languages such as SQL. However, our current work builds the basis for domain-independent methods that can be transferred to other labelled property graphs as knowledge base given the same edge labels are used. Exploiting LLMs for usually very laborious method development for NLP tasks enhances the separation of domain specific knowledge and the student interaction in natural language.

Since the shown preliminary evaluations show promising results, we are currently assessing the performance for the answer validation by comparing different open-source models, and experiment with prompt engineering methods for more accurate and stable results. Furthermore, use-case specific evaluations should be conducted on how hallucinations of the LLM are more suppressed and hindered in this approach than without using the KG in the inference process. Adapting LLMs to the method of Automatic Short Answer Grading and the respective development pipeline [4] to adequately evaluate student responses in our ITS sounds promising for future research on answer validation in our context.

In the future we aim to combine this declarative knowledge base with procedural knowledge by mapping the concepts of our semantic network to specific tasks that require those underlying concepts. This should allow a flexible switch between testing for applied skills in SQL on specific databases and the underlying declarative concepts as prerequisite or supplement to the required skills for the task. Exploring the possibility to incorporate LLM into the tutoring module e.g. to select didactic strategies, seems to be a promising new integration point. However, to our knowledge, LLMs have not yet shown results comparable to suitable strategies proposed by human teachers, yet.

References

[1] Alshaikh, Z. et al.: Experiments with a socratic intelligent tutoring system for source code understanding. In: The Thirty-Third International Florida Artificial Intelligence Research Society Conference (FLAIRS-32). (2020).

[2] Anderson, J.R., Skwarecki, E.: The automated tutoring of introductory computer programming. Commun. ACM. 29, 9, 842–849 (1986). https://doi.org/10.1145/6592.6593.

[3] Bianchini, F. et al.: Enhancing Complex Linguistic Tasks Resolution Through Fine-Tuning LLMs, RAG and Knowledge Graphs (Short Paper). In: Almeida, J.P.A. et al. (eds.) Advanced Information Systems Engineering Workshops. pp. 147–155 Springer Nature Switzerland, Cham (2024).

[4] Burrows, S. et al.: The Eras and Trends of Automatic Short Answer Grading. International Journal of Artificial Intelligence in Education. 25, 1, 60–117 (2015). https://doi.org/10.1007/s40593-014-0026-8.

[5] Crow, T. et al.: Intelligent tutoring systems for programming education: a systematic review. In: Proceedings of the 20th Australasian Computing Education Conference. pp. 53–62 Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3160489.3160492.

[6]. Graesser, A.C. et al.: AutoTutor. In: Applied natural language processing: Identification, investigation and resolution. pp. 169–187 IGI Global (2012).

[7] Ji, Z. et al.: Survey of Hallucination in Natural Language Generation. ACM Computing Surveys. 55, 12, 1–38 (2023). https://doi.org/10.1145/3571730.

[8] Jiang, A.Q. et al.: Mistral 7B, https://arxiv.org/abs/2310.06825, (2023).

[9] MistralAI: Mixtral 8x7B Instruct, https://huggingface.co/mistralai/Mixtral-8x7B-Instruct-v0.1, (2023).

[10. Mitrovic, A.: An Intelligent SQL Tutor on the Web. International Journal of Artificial Intelligence in Education. 13, 2–4, 173–197 (2003).

[11] Mousavinasab, E. et al.: Intelligent tutoring systems: a systematic review of characteristics, applications, and evaluation methods. Interactive Learning Environments. 29, 1, 142–163 (2021). https://doi.org/10.1080/10494820.2018.1558257.

[12] Neo4J Inc.: Neo4J, neo4j.com, (2024).

[13] Nkambou, R. et al. eds: Advances in Intelligent Tutoring Systems. Springer Berlin Heidelberg, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13451-7.

[14] Pan, J.Z. et al.: Large Language Models and Knowledge Graphs: Opportunities and Challenges, http://arxiv.org/abs/2308.06374, (2023).

[15] Pan, L. et al.: Logic-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning, http://arxiv.org/abs/2305.12295, (2023).

[16] Pan, S. et al.: Unifying Large Language Models and Knowledge Graphs: A Roadmap, http://arxiv.org/abs/2306.08302, (2023). https://doi.org/10.48550/arXiv.2306.08302.

[17] Pennington, N. et al.: Transfer of Training Between Cognitive Subskills: Is Knowledge Use Specific? Cognitive Psychology. 28, 2, 175–224 (1995). https://doi.org/10.1006/cogp.1995.1005.

[18] Rivers, K., 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). https://doi.org/10.1007/s40593-015-0070-z.

[19] Tack, A., Piech, C.: The AI teacher test: Measuring the pedagogical ability of blender and GPT-3 in educational dialogues. arXiv preprint arXiv:2205.07540. (2022).

[20] Vujinović, A. et al.: Using ChatGPT to Annotate a Dataset: A Case Study in Intelligent Tutoring Systems, https://www.techrxiv.org/articles/preprint/Using_ChatGPT_to_Annotate_a_Dataset_A_Case_Study_in_Intelligent_Tutoring_Systems/23617551/1, (2023). https://doi.org/10.36227/techrxiv.23617551.v1.

[21] Zhang, P.: TinyLlama, https://huggingface.co/TinyLlama/TinyLlama-1.1B-Chat-v1.0, (2023).

[22] Zhou, G.: Towards designing a knowledge-based tutoring system: SQL-tutor as an example. New Jersey Institute of Technology (1996).