Automated Assessment Generation in Intelligent Tutoring Systems

László Kovács(✉), László Csépányi-Fürjes, and Ghanim Hussein Ali Ahmed

University of Miskolc, 3515 Miskolc, Egyetemváros, Hungary
{kovacs,ghanim}@iit.uni-miskolc.hu,
Laszlo.Csepanyi-Furjes@uni-miskolc.hu

Abstract. Intelligent Tutoring Systems have a huge potential to improve the efficiency of learning and teaching, especially for personalized adaptive self-learning. The investigation presented in this paper focuses on the implementation of a submodule, the Automated Question Generation for Assessment unit, using a combined rule based and machine learning based methods. The proposed module is based on a proposed background domain ontology, and it works in two phases. A generator automaton is used to construct the answer templates in the first phase. In the next phase, an encoder-decoder neural network translator unit was used to generate the question sentence templates for the selected answer templates. According to the tests with the implemented prototype assessment generator application on the SQL domain, the proposed framework is suitable to generate assessment tasks in a very efficient way.

Keywords: Intelligent tutoring system · Assessment generation · Machine learning

1 Introduction

Intelligent Tutoring System (ITS) is a computer-based system that aims to offer direct and customized instruction or feedback to learners with personalized guidelines based on their cognitive skills [1]. ITS systems can be used to simulate the tutoring process of human teachers, including providing instructions, suggestions, decisions regarding the learning path and assessments usually without requiring intervention from a human teacher. It is widely accepted that ITS has a huge potential to improve the efficiency of learning and teaching, especially for personalized adaptive self-learning. According to [2], intelligent tutoring may be viewed as "an effort to capture in computer technology the capabilities and practices of a human instructor who is expert in both the subject matter and one-to-one tutoring". On the market, we can find many successful ITS applications used by thousands of students, including SQLTutor [3], ALEKS [4], Cognitive Tutor [5] and ASSISTments [6].

As most of the current ITS solutions apply different levels of machine intelligence to control the educational processes, the ITS is considered as a subfield of the area of Artificial Intelligence in Education (AIEd). According to the experts [7], the role of

AI-based educational applications has grown 40% in recent years. Since the appearance of the first AIEd proposal nearly 30 years ago, a wide spectrum of intelligent tools has been developed which can be categorized into three main domains [8]: applications for personal tutors; application for collaborative learning and application for domain knowledge representation (like virtual reality). Regarding the target users, in [9], the following categories are identified: learner-facing, teacher-facing and system-facing intelligent applications. According to [7], the ITS module developments are mainly focused on improving the adaptivity, the assessment processing, student profiling and learning path prediction. In our project works, we were focusing on the teacher-faced applications which is aimed at supporting the teachers with reduction of the workload by automating the key educational activities [7].

ITS applications are designed to behave as intelligent as an expert teacher [10] with intentional application of widely accepted pedagogical strategies [11]. The general architecture of ITS applications involves the following key components to provide the expected level of intelligence [12].

- student model
- teacher model
- domain model
- diagnosis model.

The domain model contains the semantic description of the teaching course content, usually using a semantic network format. Beside the core knowledge model elements, the working of ITS applications is based also on some external components, like fact databases, natural language processing modules, chatbots or web services.

2 Current Issues, Limitations, and Challenges in ITS

The main expectations and properties of an ideal ITS [7] cover the following properties:

- An intelligent tutoring system should be capable of supporting learning
- An intelligent tutoring system would behave as if it genuinely cared about the student's success [13].
- The systems themselves would also learn how to teach [14]
- The systems would study what worked, and when it worked, and they would improve themselves over time [15].

Due to the complexity of the ITS systems, the implemented prototype systems usually cannot meet all the high expectations. Thus, beside the positive evaluations, in the literature, we can find some critical remarks too (see [16]) where the main conclusion of the test experiment was that human tutoring dominates ITS solution both in efficiency and flexibility. As the study in [17] concluded, current AIEd systems can make many mistakes and ITS requires human control. One reason for the weak performance is that the quality of the AI model mainly depends on the quality of the training data, thus the most expensive part in the development of ITS applications is the construction of training data sets having appropriate size and quality.

According to [18], the main barriers of development and adoption of efficient ITS systems can be summarized into the following points:

- limited size of the background knowledge model
- quality of the background knowledge model
- low adaptivity level
- low flexibility in customization.

Based on the experienced difficulties, some works focus on investigation of the shortcomings providing suggestions on the real application options. In [19], a new naming is proposed instead of ITS; the education community can be provided in the near future only with Intelligent Teaching Assistant (ITA) systems. As [20] points out the teachers require such systems which provide.

- more accurate and wider scope assessment of the students
- decision support in selection of efficient learning path for the different students
- better analytical statistics on the students
- providing more explanatory instructions instead of direct commands.

In [21], the author presents a sharp criticism on the shortcomings of the current ITS solutions stating that we need stupid tutoring systems but intelligent teachers: *"Perhaps we do not in fact need intelligent tutoring systems. Perhaps instead what we need, what we are already developing, is stupid tutoring systems. Tutors that do not, themselves, behave very intelligently. But tutors that are designed intelligently, and that leverage human intelligence.*

In other words, perhaps what we need is stupid tutoring *systems, and intelligent humans."*

These experiments and the outcome of the detailed evaluation in [7] show, that despite the great achievements in AI and in some components of AIEd, much more analysis and development are needed to improve the practical efficiency of ITS applications.

Based on the existing technical difficulties in implementation of high level ITS modules, our investigation focuses on the implementation of sub-module, namely on Automated Question Generation for Assessment (AQGA), using a combined rule based and machine learning based methods. As the proposed method is based on specific knowledge model elements, we also present the supporting knowledge model background of the AQGA unit.

3 Content Model in the Proposed ITS Framework

Considering the functional and data components of current ITS systems, we include the following key components into the proposed ITS system:

- domain module to describe the course topics including the knowledge units and competencies
- student module to model the behavior of the students

- teacher module to model the behavior of the tutor
- training material repository
- assessment management module
- interface module to implement intelligent, human oriented interface to the system
- background supporting units like NLP module or reasoning engine.

This structure can be observed among others in Intelligent Tutoring System Builder (ITSB) [22] or in Cognitive Tutor Authoring Tools (CTAT) [23]. The proposed architecture is presented in Fig. 1.

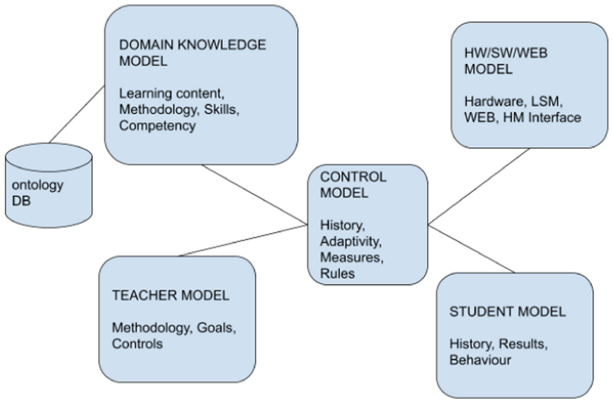

DOMAIN KNOWLEDGE MODEL
Learning content, Methodology, Skills, Competency
HW/SW/WEB MODEL
Hardware, LSM, WEB, HM Interface
ontology DB
CONTROL MODEL
History, Adaptivity, Measures, Rules
TEACHER MODEL
Methodology, Goals, Controls
STUDENT MODEL
History, Results, Behaviour

Fig. 1. Proposed e-tutor architecture.

The goal of the domain model is to provide detailed information about the teaching content including all details needed to perform an automated and adaptive implementation and control in the learning process. In [24], the authors investigated the conceptualization of teacher content knowledge from many aspects including pedagogical viewpoints. The work identifies the following dimensions.

- Content knowledge
- Curriculum knowledge
- Pedagogical content knowledge

The Pedagogical content knowledge covers the integration of content knowledge and pedagogy to perform the teaching using a special form of professional understanding. The work highlights the view that teaching a subject is more than knowing the facts, the teachers should understand the organizing principles and structures.

In [25], the following explicit requirements are set on the content model:

- common content knowledge, teachers need to know the material they teach
- competency to perform domain specific operations
- specialized content knowledge with methodological aspects of the domain
- skills in appropriate survey developments.

Beside pedagogical aspects, another important background for domain modelling is the field of knowledge engineering (KE). In KE, the main approaches to describe the knowledge structure cover among others the semantic graph, the framework structures [26], the semantic and the ontology models [27]. The main benefits of ontology are that this model provides explicit conceptualization providing a rich set of concept construction operators and a broad area for integrity validations.

Based on requirement analysis, our proposed model incorporates the following components:

– K-units refers to knowledge units as a unit of education that defines a course, knowledge topics, domains, key concepts, or teaching units of the learning materials.
– K-slots refer to knowledge slots as an atomic property of the K-units. Every slot has a value domain and concept type.
– R-units refer to rule units as a rule or constraints defined on the k-units and the k-slots. The rule unit is a set of explicit or implicit ontology regulations or principles governing behavior or procedure in a particular activity area.
– T-units refer to task units as an activity related to the k-units and k-slots. T-unit describes an activity to be performed by students.
– M-Units refers to material units as teaching material for the k-units. M-units are teaching materials that are used for learning the k-unit. M-units contain any parts of the educational community or learning material.
– M-Resources refers to material resources.

Regarding the main relationships, the ontology model contains the following elements:

– K-unit taxonomy relationship: it describes the specialization among the K-units
– K-unit component relationship: one K-unit consists of other K-units
– K-competency relationship among the K-units
– KT- assignment: every T-unit task is related to a set of K-units.

The basic building blocks and the key relationship components of the proposed domain model are presented in Fig. 2.

Regarding the completeness of the proposed domain knowledge model, we should first consider the main functional expectations from the viewpoint of the ITS systems. The presented model can cover the information requirements of a large set of functionalities, including:

– identification of the relevant knowledge units together with the related training materials, study aids
– automated navigation among the knowledge units during the learning/teaching process
– automated assessment of the student's knowledge level
– providing hints/suggestion for the students.

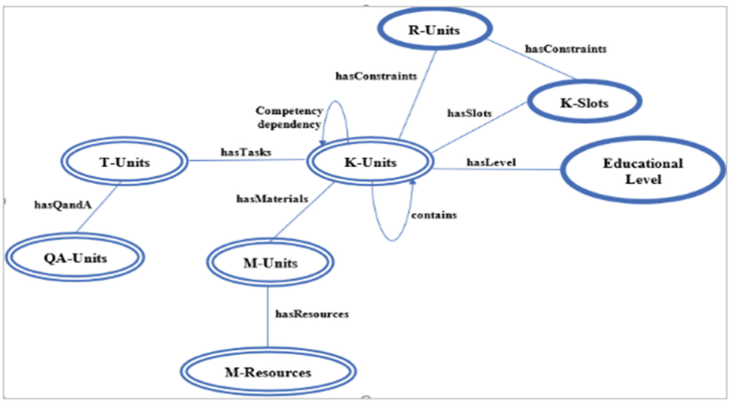


Fig. 2. Schema elements of the domain ontology

On the other hand, the model should be considered as the base framework which can be extended with more sophisticated blocks in the future. Among the possible extensions, we can highlight the following elements:

– compound task structure (current task model uses an atomic approach)
– a more sophisticated grading module for the task units
– direct management of the implicit competency lattice of the tasks

4 Tools for Automated Assessment Generation

According to [7], the automated grading process is the focus area in ITS assessment processing. Due to the large complexity of the assessment module, the existing automated assessment systems require a relatively large human preprocessing and calibration effort. On the other hand, the teachers really need an efficient tool for generation of assessment units, especially in the current epidemic lockdown situation where only the online education mode is possible at each level of the education. Current electronic LMS systems usually do not support automatic assessment generation, thus teachers should devote a lot of time and effort to construction of the assessment units.

In our model, we propose a semi-automatic tool supporting the assessment generation. The description of the question-answer QA-unit is part of the domain model and it has the following structure:

TASK task_name.
WITH variable_declaration.
QUESTION question_schema.
ANSWER answer_schema.

where the variable declaration refers to the variables denoting knowledge K-units or K-slots. For example, in the domain on SQL commands, the variables may denote tables or table columns from the background database. The following sample QA unit uses this SQL domain:

TASK insert a new record to the table
WITH t: Table, c: #t.column[SEQ]
QUESTION
Add a new record to the table #t.name (#c.name)
ANSWER
insert into #t.name values (#c.ANY)

The term Table is defined here as a concept having among others name, column, and PK properties. In the generation of the concrete task units, the variables are substituted randomly with some corresponding values from the domain database instance.

Regarding the generation of the QA-units, we have tested an automatic AI-based solution to provide a more efficient method beside the manual QA-unit construction. Our assumption is that the answer section can be generated with some kind of automaton. For example, in the case of SQL domain, an FSA engine can be used to generate valid SQL commands containing variables. In our prototype system, this automaton also has some parameters to control the generation process like the probability of the different SQL commands.

In the next phase, the question section is generated automatically from the answer section. For this processing step, we have tested two options on our SQL domain:

– A: using an automaton approach
– B: machine learning approach

In case A, the automaton was generated manually. The main benefit of this approach is that the automaton will incorporate the knowledge of the human teacher. Regarding the shortcomings, we can mention the following problems:

– it requires significant programming effort
– it is hard to describe complex cases
– high dependency on the teacher knowledge level

In the case of the machine learning approach, we have implemented an encoder-decoder neural network model that transforms the answer sentence into the question sentence The encoder-decoder network structure [28] contains two GRU-RNN layers, one for encoding and the other for decoding steps. The first RNN unit converts the input sequence into a state vector which contains the key hash description of the input sequence. This state vector will be fed into the decoder unit outputting the translated sequence. The most common use of this structure is automated language translation (Figs. 3 and 4).

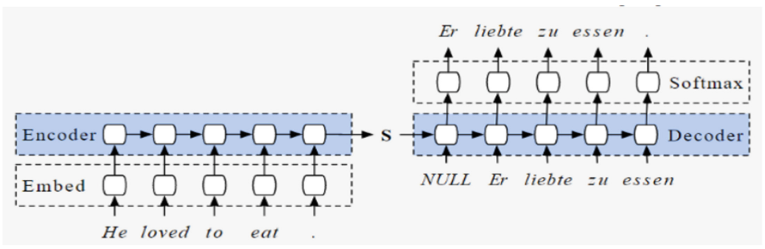
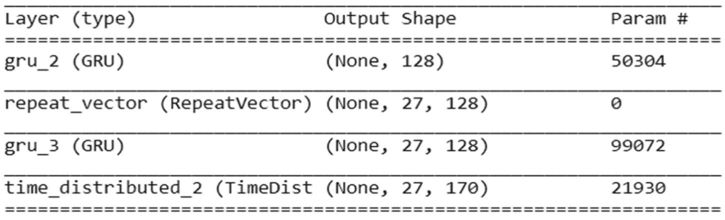


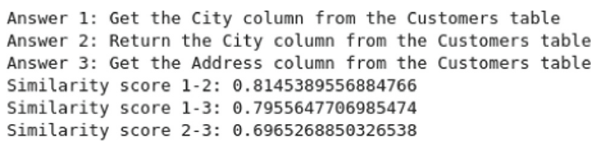
Fig. 3. Encoder-decoder [https://www.guru99.com/seq2seq-model.html]



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
gru_2 (GRU)                  (None, 128)               50304     
_________________________________________________________________
repeat_vector (RepeatVector) (None, 27, 128)           0         
_________________________________________________________________
gru_3 (GRU)                  (None, 27, 128)           99072     
_________________________________________________________________
time_distributed_2 (TimeDist (None, 27, 170)           21930     
=================================================================
```

Fig. 4. Neural network architecture in our test system

Beside the question generation encoding-decoding neural network module, we have developed a domain specific embedding model, too. The NLP embedding models [29] are used to represent the words of a natural language in high dimensional vector space such that words with similar meaning have similar positions in the vector space. The similarity is usually measured with the context similarity of the words, i.e., words occurring in similar contexts should be similar in meaning. The construction of word embeddings is a very expensive operation both in time and computer resources. Also, it requires hundreds of millions of training samples. Thus, usually an existing general embedding model (like the BERT model [30]) is adapted to the specific domain. This general embedding model is called pre-trained model and the domain specific adaptation process is called fine-tuning. During the fine tuning, a domain specific training set is used to update the pre-trained weights so that the model can produce much more accurate results in that certain domain. Two kind of Bert models can be built using a stack of encoder layers. $\text{BERT}_{\text{BASE}}$ consists of 12 encoder layers and produces 768 dimensional vectors and $\text{BERT}_{\text{LARGE}}$ which contains 24 layers and produces 1024 dimensional embedding vectors. The problem with these models is that it is computationally expensive to use them for sentences. For our experiments, we have used a modified variant of the pretrained BERT model called Sentence-BERT [31]. This model is trained to embed whole sentences and to semantically compare them using cosine-similarity. Our aim with this approach was to predict how accurate a free textual answer is. Comparing this experiment with the question generation we used a reverse order here. The QUESTION was an SQL statement (e.g., SELECT City FROM Customers), and the ANSWER was the textual description of that statement. The result of the experiment showed that the cosine similarity of two semantically equal correct answers is higher than the cosine similarity of an incorrect and a correct one, Fig. 5. Though the difference is not significant enough it can give the teacher a hint which answer is incorrect. Our plan is to develop this approach further by fine-tuning the model with the SQL domain.



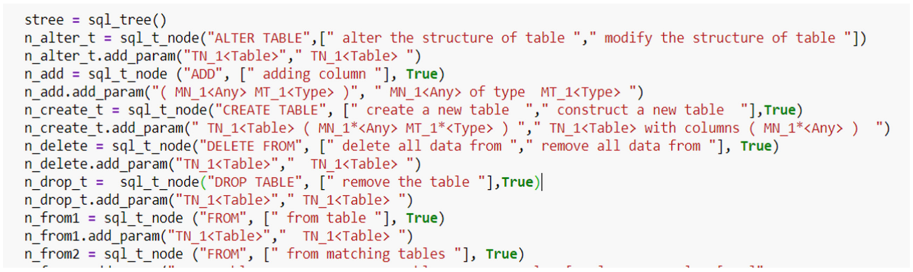
```
Answer 1: Get the City column from the Customers table
Answer 2: Return the City column from the Customers table
Answer 3: Get the Address column from the Customers table
Similarity score 1-2: 0.8145389556884766
Similarity score 1-3: 0.7955647706985474
Similarity score 2-3: 0.6965268850326538
```

Fig. 5. Cosine similarity of two correct and an incorrect answer

Although the presented approach referred to IT domain, this assessment generation model can be used for a wide variate of knowledge domains. In most cases, a conversion module is needed which will convert the answer activity description into a formal, functional format. For example, in elementary mathematics, the calculation procedure can be given with a function having arguments from the given domain. For example, to calculate the area of a rectangle, we can introduce the function *area_rect(r.a, r.b).* After this preprocessing step, we can apply our approach to generate the parameterized question-answer pairs.

5 Test Results for Automated Assessment Generation

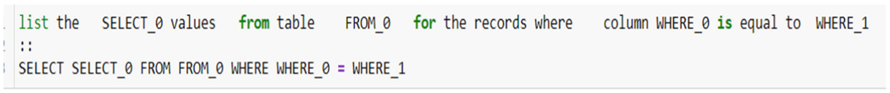
For the generation of valid SQL commands, we have constructed a reduced SQL generator automaton. The automaton was implemented as a graph containing both the SQL syntax and the verbal NLP description (Fig. 6).



```
stree = sql_tree()
n_alter_t = sql_t_node("ALTER TABLE",[" alter the structure of table "," modify the structure of table "])
n_alter_t.add_param("TN_1<Table>"," TN_1<Table> ")
n_add = sql_t_node ("ADD", [" adding column "], True)
n_add.add_param("( MN_1<Any> MT_1<Type> )", " MN_1<Any> of type  MT_1<Type> ")
n_create_t = sql_t_node("CREATE TABLE", [" create a new table  "," construct a new table  "],True)
n_create_t.add_param(" TN_1<Table> ( MN_1*<Any> MT_1*<Type> ) "," TN_1<Table> with columns ( MN_1*<Any> )  ")
n_delete = sql_t_node("DELETE FROM", [" delete all data from "," remove all data from "], True)
n_delete.add_param("TN_1<Table>","  TN_1<Table> ")
n_drop_t =  sql_t_node("DROP TABLE", [" remove the table "],True)
n_drop_t.add_param("TN_1<Table>"," TN_1<Table> ")
n_from1 = sql_t_node ("FROM", [" from table "], True)
n_from1.add_param("TN_1<Table>","  TN_1<Table> ")
n_from2 = sql_t_node ("FROM", [" from matching tables "], True)
```

Fig. 6. A sample definition for generator automata

The next figure (Fig. 7) shows an example for an output item pair:



```
list the   SELECT_0 values   from table   FROM_0   for the records where   column WHERE_0 is equal to  WHERE_1
::
SELECT SELECT_0 FROM FROM_0 WHERE WHERE_0 = WHERE_1
```

Fig. 7. An example for generated sentence template pairs

The training set contains both valid exercises collected from the internet, like W3CSchool [https://www.w3schools.com/], and synthetic examples shown in the previous step. Using this hybrid training set generation approach, we could construct a larger size training set with low cost.

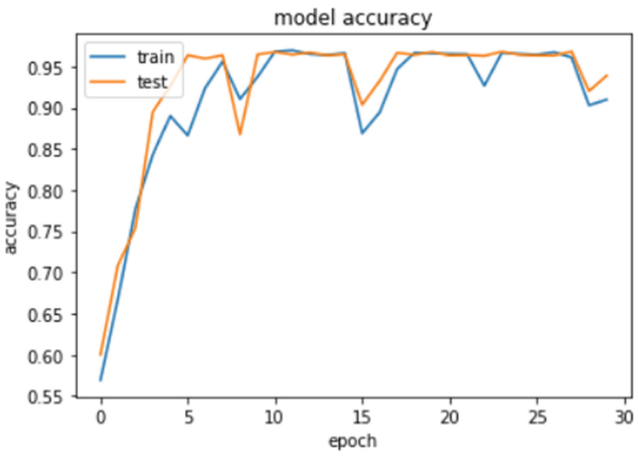
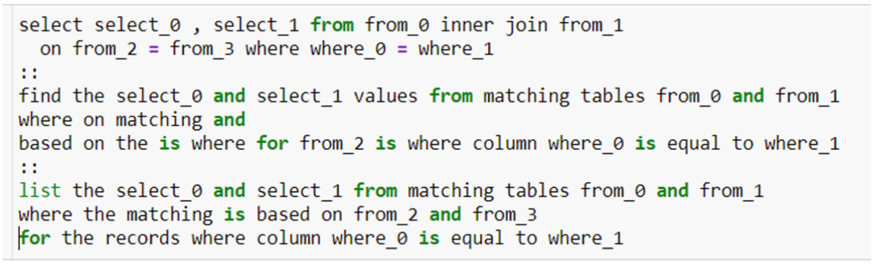


Fig. 8. Accuracy curves for the translation training

Using N = 1000 training size and E = 100 epochs, the validation accuracy reached an acceptable 94.7% value. With a larger dataset (N = 10000), accuracy level reached the 97.7% value. The accuracy values for both train and test are shown in Fig. 8.

The example in Fig. 9, shows a complex syntax, as we can see the prediction from the smaller dataset (N = 1000) was incorrect, especially at near the end of the sentence. In the case of a large dataset (N = 10000), the prediction is correct.

As our experiences show, the encoder-decoder model was able to learn at a high accuracy level also the complex SQL commands generating longer sentences.



```
select select_0 , select_1 from from_0 inner join from_1
  on from_2 = from_3 where where_0 = where_1
::
find the select_0 and select_1 values from matching tables from_0 and from_1
where on matching and
based on the is where for from_2 is where column where_0 is equal to where_1
::
list the select_0 and select_1 from matching tables from_0 and from_1
where the matching is based on from_2 and from_3
for the records where column where_0 is equal to where_1
```

Fig. 9. Examples for the generated assessment templates

6 Conclusion

The automatic assessment management module is one of the main components in ITS systems. In this paper, we presented an automated method to generate assessment question and answer pairs related to a knowledge domain. The proposed module relates to a background domain ontology, where the domain schema is based on the following proposed building blocks: K-units (concepts), K-slots (properties), R-units (constraint rules), M-units (study materials) and T-units (tasks given by question-answer pairs), K-unit taxonomy relationship, K-unit containment relationship and K-competency relationship. The question-and-answer pairs are defined with a template sentence containing placeholders with reference to ontology elements. The proposed method uses a generator automaton to construct the answer templates in the first phase. In the next phase, an encoder-decoder neural network translator unit was used to generate the question sentence templates for the selected answer templates. The paper presents a domain specific embedding model, too. The proposed embedding neural network can be used to compare the semantic similarity of free-text answer sentences. According to the tests with the implemented prototype assessment generator application on the SQL domain, the proposed framework is suitable to generate assessment tasks in a very efficient way.

In the future research, the main goal is to investigate some related open problems like the application of AIEd technology in construction of the generator automaton or the development of efficient crawler engines to collect related examples from the Internet.

References

1. Nkambou, R., Mizoguchi, R., Bourdeau, J. (ed.): Advances in intelligent tutoring systems. Springer Science & Business Media (2010). https://doi.org/10.1007/978-3-642-14363-2
2. Sottilare, R., et al.: Design Recommendations for Intelligent Tutoring Systems. U.S. Army Research Laboratory, Florida (2015)
3. Mitrovic, A.: An intelligent SQL tutor on the web. Int. J. Artif. Intell. Educ. **13**(2–4), 173–197 (2003)
4. Craig, S.D., et al.: The impact of a technology-based mathematics after-school program using ALEKS on student's knowledge and behaviors. Comput. Educ. **68**, 495–504 (2013)
5. Pane, J.F., Griffin, B.A., McCaffrey, D.F., Karam, R.: Effectiveness of cognitive tutor algebra I at scale. Educ. Eval. Policy Anal. **36**(2), 127–144 (2014). https://doi.org/10.3102/0162373713507480
6. Koedinger, K.R., McLaughlin, E.A., Heffernan, N.T.: A quasi-experimental evaluation of an on-line formative assessment and tutoring system. J. Educ. Comput. Res. **43**(4), 489–510 (2010). https://doi.org/10.2190/EC.43.4.d
7. Zawacki-Richter, O., Marín, V.I., Bond, M., Gouverneur, F.: Systematic review of research on artificial intelligence applications in higher education – where are the educators? Int. J. Educ. Technol. Higher Educ. **16**(1), 1–27 (2019). https://doi.org/10.1186/s41239-019-0171-0
8. Luckin, R., et al.: Intelligence unleashed: An argument for AI in education (2016)
9. Baker, T., Smith L., Anissa N.: Educ-AI-tion rebooted? Exploring the future of artificial intelligence in schools and colleges (2020)
10. Graesser, A.C., Conley, M.W., Olney, A.: Intelligent tutoring systems. In: Harris, K.R., Graham, S., Urdan, T., Bus, A.G., Sonya Major, H., Swanson, L. (eds.) APA educational psychology handbook, vol. 3: Application to learning and teaching., pp. 451–473. American Psychological Association, Washington (2012). https://doi.org/10.1037/13275-018

11. Merrill, D.C., Reiser, B.J., Michael Ranney, J., Trafton, G.: Effective tutoring techniques: A comparison of human tutors and intelligent tutoring systems. J. Learn. Sci. **2**(3), 277–305 (1992). https://doi.org/10.1207/s15327809jls0203_2
12. Huang, J., Chen, Z.: The research and design of web-based intelligent tutoring system. Int. J. Multimedia Ubiquitous Eng. **11**(6), 337–348 (2016). https://doi.org/10.14257/ijmue.2016.11.6.30
13. Self, J.A.: Student models: what use are they? In: Ercoli, P., Lewis, R. (eds.) Artificial Intelligence Tools in Education, pp. 73–86 (1988)
14. O'Shea, T.: A self-improving quadratic tutor. In: Sleeman, D.H., Brown, J.S. (eds.) Intelligent Tutoring Systems, pp. 283–307. Academic Press (1982)
15. Beck, J., Woolf, B., Beal, C.R.: ADVISOR: A machine learning architecture for intelligent tutor construction. AAAI/IAAI **2000**(552–557), 1–2 (2000)
16. Cooper, E.: Tutoring center effectiveness: The effect of drop-in tutoring. J. Coll. Read. Learn. **40**(2), 21–34 (2010). https://doi.org/10.1080/10790195.2010.10850328
17. Prinsloo, P.: Fleeing from Frankenstein's monster and meeting Kafka on the way: Algorithmic decision-making in higher education. E-Learn. Digital Media **14**(3), 138–163 (2017). https://doi.org/10.1177/2042753017731355
18. Trausan-Matu, S., Boyer, K.E., Crosby, M., Panourgia, K. (eds.): ITS 2014. LNCS, vol. 8474. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07221-0
19. Yacef, K.: Intelligent teaching assistant systems. In: International Conference on Computers in Education. Proceedings IEEE (2002)
20. Holstein, K., McLaren, B.M., Aleven, V.: Intelligent tutors as teachers' aides: exploring teacher needs for real-time analytics in blended classrooms. In: Proceedings of the 7th International Learning Analytics, Knowledge Conference (2017)
21. Baker, R.S.: Stupid tutoring systems, intelligent humans. Int. J. Artif. Intell. Educ. **26**(2), 600–614 (2016). https://doi.org/10.1007/s40593-016-0105-0
22. Naser, S.S.A.: ITSB: an intelligent tutoring system authoring tool. J. Sci. Eng. Res. **3**(5), 63–71 (2016)
23. Feng, C., Guo, G., Jin, S., Zhou, C.: Authoring tools for easy creation of adaptive tutoring. In: Spring, BHCI Capstone Project (2018)
24. Shulman, L.S.: Ways of seeing, ways of knowing: Ways of teaching, ways of learning about teaching. J. Curric. Stud. **23**(5), 393–395 (1991). https://doi.org/10.1080/0022027910230501
25. Loewenberg Ball, D., Thames, M.H., Phelps, G.: Content knowledge for teaching: What makes it special. J. Teach. Educ. **59**(5), 389–407 (2008)
26. Tsang, V., Stevenson, S.: A graph-theoretic framework for semantic distance. Comput. Linguistics **36**(1), 31–69 (2010). https://doi.org/10.1162/coli.2010.36.1.36101
27. Maedche, A., Staab, S.: Ontology learning for the semantic web. IEEE Intell. Syst. **16**(2), 72–79 (2001). https://doi.org/10.1109/5254.920602
28. Mastan, I., Shanmuganathan, R.: Multi-level encoder-decoder architectures for image restoration. In: (CVPRW) IEEE (2019)
29. Hao, B., Henghui, Z., Ioannis, P.: Enhancing clinical BERT embedding using a biomedical knowledge base. In: Proceedings of the 28th International Conference on Computational Linguistics (2020)
30. Devlin, J., Chang, M.V., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805. ArXiv: 1810.04805 (2018)
31. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. arXiv:1908.10084. ArXiv: 1908.10084 (2019)