Three-Party Interactive Tutoring System for Mastering Machine Learning

Takumi Kojima, Takuya Iida, Daina Teranishi and Masahiro Araki
Kyoto Institute of Technology
{kojima, iida, teranishi}@ii.is.kit.ac.jp, araki@kit.ac.jp

Abstract

Interactive tutoring systems are expected to contribute to self-learning in various areas, such as machine learning. In this study, we develop an interactive tutoring system that performs tutoring in a three-party dialogue form using multiple virtual agents. This would induce lower psychological burden on the learner than one-to-one dialogues. To adapt this system to various levels of users, question-answering functionality is essential. The proposed system generates answers to questions by using a knowledge graph automatically constructed from a textbook.

1 Introduction

Intelligent tutoring systems that can interact with learners are highly useful for subjects with significantly different preliminary knowledge levels of learners, such as machine learning. In this study, we develop a tutoring system in a three-party dialogue form using multiple virtual agents (Figure 1).

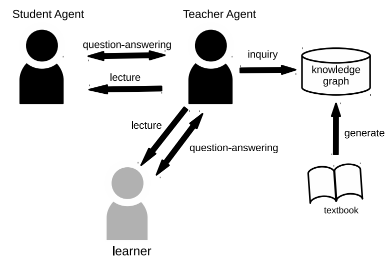

Figure 1: Three-party dialogue tutoring system

In previous works, three-party dialogue systems provided the feeling of easy participation in dialogue situations and avoided dialogue breakdown (Sakamoto et al., 2009), (Sugiyama et al., 2018). Learners can get a feeling of participation in the dialogue even in the phase of talking between agents, and can learn by interactions with the teacher agent. Therefore, this system is expected to reduce the psychological burden for the learner compared with in one-to-one dialogue and to achieve a learning process with moderate tension.

We use two agents assuming teacher and student roles. These two agents play tutoring situations. The teacher agent occasionally prompts the human learner for a response. To adapt this system to various levels of users, question-answering functionality is essential. The proposed system generates an answer to the question by using a knowledge graph automatically constructed from a textbook on machine learning written by one of the authors (Araki, 2018).

2 Related work

Although most of the dialogue systems currently in practical use utilize the two-party dialogue form, some systems achieve a natural dialogue situation via three (or more) party dialogue. Swartout and others (Swartout et al., 2010) implemented a two-agent dialogue system that guides museum visitors. Sakamoto and others (Sakamoto et al., 2009) demonstrated that information presented by three-party dialogue, including the interaction between robots, attracts people' s interest. In this study, a three-party dialogue was adopted to reduce the psychological burden on the learner.

Another essential factor in an intelligent tutoring system is knowledge representation when answering a user's question. The knowledge graph is a promising method that can easily verify the validity. Luan and others (Luan et al., 2018) proposed a method to extract a knowledge graph from abstracts of artificial intelligence related papers in English. Bordes and others (Bordes et al., 2014) proposed a response generation method utilizing subgraph embedding by using Freebase as

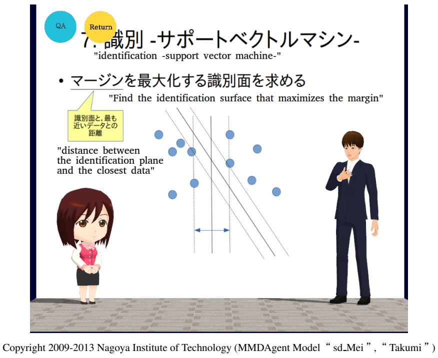

Figure 2: Example of a tutoring situation

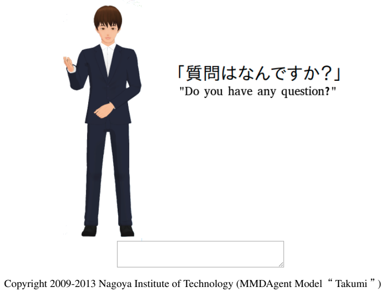

Figure 3: Question-answering screen

a knowledge base. In this study, we generated a knowledge graph to answer a user's question from a textbook on machine learning.

3 Three-party interactive tutoring system

When implementing the tutoring system, we set three objectives: (1) multi-device support for smartphones, tablets, and PCs, (2) low development cost, and (3) support for users with various knowledge levels.

As a development platform, we use PocketMMDAgent [1] (Figure 2), a multi device version of MMDAgent (Lee et al., 2013) that can build voice interactions with virtual agents.For the question answering stage, we use a web browser as a supplementary window (Figure 3).

3.1 System configuration

The configuration of our proposed system is shown in Figure 4. It is divided into the server and client sides. The server side provides the tutoring content and question answering functionality. It simplifies preparation on the client side and only requires the installation of PocketMMDAgent.

The PocketMMDAgent downloads the tutoring content from the content server, plays the scenar-

[1]PocketMMDAgent: https://mmdagent.lee-lab.org/

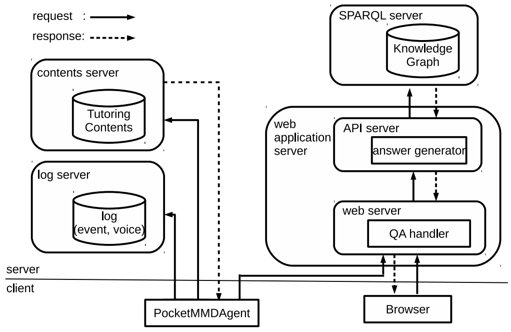

Figure 4: Tutoring system configuration

ios, sends a request to open the question answer page, and sends the log to the log server. In PocketMMDAgent, the utterance content of the system and the user's utterance pattern in the scenario must be described. However, the learner's questions are unpredictable. Therefore, it is difficult to realize the question answering function only on PocketMMDAgent. In this study, we implement the question answering functionality via a web browser. The web browser displays a question answering page in response to the request from the PocketMMDAgent and reproduces an answer generated on the server side for a learner's question.

The web application server is a server for question answering.The web server recognizes the speech input and sends the recognized text to the API server. The API server generates an answer using the SPARQL server that manages the knowledge graph and returns the synthesized speech file for the answer.

The system can update the scenario and collect logs on the server side; thus, we can dynamically update dialogue content and collect a learner's learning history.

3.2 Dialogue scenario

The dialogue scenario of the PocketMMDAgent is described as Finite State Transducer (FST) files. Each state accepts a command and outputs an event. In this study, as elements of dialogue scenario for the tutoring, we set six phases of interaction: (1) lecture by the teacher agent, (2) question answering between the teacher and student agents, (3) question from the teacher agent to the learner, (4) answer from the teacher agent to the learner, (5) selection of chapter, and (6) slide change.

Tutoring dialogue with limited human participation such as our proposed system are appropriate for learners with limited knowledge

Table 1: Example of tutoring dialogue

speaker	utterance sentences
teacher Agt.	The grouping of machine learning based on the training data and output can be expressed like this. First, in the training data, it is roughly grouped depending on whether or not the correct answer is given. We call supervised learning where the training data has the correct answer, unsupervised learning where it does not have, and intermediate learning which is neither of them. Also, each is further divided from the viewpoint of the type of output. What does the output give a type of supervised learning?
student Agt.	The output of classification problem is category type. The output of regression is numerical value.
teacher Agt.	That's right. Moreover, in unsupervised learning. Model estimation models the entire data. Pattern mining extracts useful information from the subset of data. What about intermediate learning? What do you think?
learner	Difference of what is intermediate.
teacher Agt.	I see. Intermediate learning focuses on what is intermediate. If the part of training data is labeled, it is called semi-supervised learning. Intermediate between supervised and unsupervised it is called reinforcement learning. Do you have any question?
learner	Yes.
teacher Agt.	Please, press the blue button to open the QA page. Please, press the yellow button to resume.
learner	(press blue button) What does unsupervised learning learn?
teacher Agt.	Unsupervised learning learns regularity.

Translated from Japanese

(Graesser et al., 2017). Learners can participate in tutoring dialogues by question answering with the teacher agent.

These dialogue scenarios do not require complicated state transitions, and it is possible to easily convert the formatted text describing the scenario (phase id, speaker, content) into FST files. An example dialogue is shown in Table 1.

4 Question answering function

Although we limit the dialogue pattern to the simple one as described in the previous section, it is still difficult to prepare multiple scenarios for learners with different levels of preliminary knowledge.Therefore, we add a question answering functionality in our system to follow up the content. It is also desirable to make such question answering functionality as simple as possible. As a knowledge element for answer generation with a low construction cost, we generate the knowledge graph from the textbook and use it to generate the answer (Figure 5).

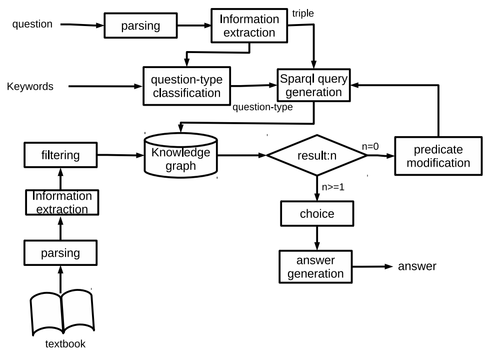

Figure 5: Flow of generating an answer for a question

4.1 Knowledge graph construction

In this study, we use the knowledge graph in RDF format generated by analyzing the textbook and complemented by ontology on the machine learning that was created manually.

The ontology contains a machine learning class that represents machine learning problems, algorithm class, learning method class, model class, parameter class, and data type class (numerical, categorical, and mixed).

To generate the knowledge graph in RDF format, we parse the textbook and extract the candidates for triples from the phrase structure and parallel structure. We use a Japanese morphological analysis system Juman++ (Morita et al., 2015) and a Japanese dependency and case structure analyzer KNP(Kawahara and Kurohashi, 2006). We extract the nominative case as a subject, the verb as a predicate, and other cases as an object in phrase structures; we also extracted nouns in parallel relations by using features obtained by KNP. We filtered triples based on TF-IDF score to exclude common words. The triples are extracted such as ('pattern mining', 'discover', 'regularity') from translated Japanese. The triples extracted by the above method are stored as Notation3 (N3) in the SPARQL server.

4.2 Answer generation

During answer generation, triples are created by parsing the question in the same manner as the triple extraction. The empty elements of the triples are inquired to a knowledge graph, and the answers are generated using the template.The configuration of the answer generation procedure is as Figure 5.

To generate the appropriate answer for the question, we classify the question types by calculating the similarity between the keywords and the question using Word2Vec (Mikolov et al., 2013) and Doc2Vec (Le and Mikolov, 2014). The types are 4W1H (why, when, where, what, how), because the who type is unlikely to occur in the machine learning field. The keywords are set to ”why, when, where, what, how” for interrogative purposes and to ”reason, time, scene, meaning, way” for affirmative sentences.

We send inquiries to the knowledge graph using a SPARQL query generated from the question triple based on the template. At that time, if the number of query results is 0, an inquiry is sent again after replacing the predicate in the triple to the most similar predicate in the knowledge graph as calculated by Word2Vec.If the number of query results exceeds 1, we select one at random.We generate an answer using the query result based on a template that combines the subject, object, predicate, and expression for each question type. We set the expression to (1) reason (why), (2) time (when), (3) scene (where), (4) ϵ (what), (5) method (how) as translated from Japanese.

5 Implementation

To validate our design of the three-party dialogue tutoring system proposed herein, we verify the operation when all servers are deployed on open servers as well as the answer generation operation using the knowledge graph. We use Apache [2] as a content server and a web server, Flask [3] as a web application framework for the API server, and Apache Jena Fuseki [4] as a SPARQL server.

To generate a knowledge graph, we use 466 sentences from chapters 1, 3, 4, and 5 of the chosen textbook on machine learning (Araki, 2018). We also utilize some self-made questions as test data to generate answers.

5.1 Inspection

In our system, we confirmed that the dialogue scenario and the question answering function work with all the servers of Figure 4 placed on the public server by using the tools mentioned above.

During the generation of the knowledge graph, 401 cases were obtained from the text in the textbook. Moreover, examples of the result of question answering obtained during answer generation are shown in Table 2.

Table 2: Examples of answer generation

objective	question	answer
subject	What does the class classify?	Boundaries separate classes.
predicate	How does model estimation perform for parameters?	Model estimation is a method of estimating parameters

Translated from Japanese

If the obtained query cannot be searched, the answer that mean it is not known is returned.

5.2 Consideration

Our system can be operated on all servers as public servers. Therefore, users can utilize this system with various devices that are compatible with the PocketMMDAgent and the web browser.

Moreover, the generation of a knowledge graph extracted from the textbook is considered effective, as we generated approximately 1 triple per sentence. Our system could generate simple answers by using a knowledge graph generated from the textbook, the analysis of which enabled the question answering function, and this tutoring system can be developed at a low cost.

6 Conclusion and future work

To assist users who study various subjects, we developed a three-party dialogue tutoring system using PocketMMDAgent. The system possesses question answering functionality based on a knowledge graph. In the prototype implementation, it is verified that the proposed system operates on the public server for the multi-device requirement and can provide answers for questions having a specific structure. In future works, we will investigate the influence on the psychological burden in the subject experiment.

Acknowledgments

We would like to express our gratitude to Dr. Akinobu Lee for his advice on PocketMMDAgent. Funding from the TIS Inc. is gratefully acknowledged. This study is also partly funded by JSPS research grant 18K11569.

[2]Apache: https://httpd.apache.org/

[3]Flask: http://flask.pocoo.org/

[4]Apache Jena Fuseki: https://jena.apache.org/documentation/fuseki2/

References

Masahiro Araki. 2018. *Introduction to machine learning to begin with free software (2nd Ed.) (Japanese)*. Morikita Publishing Co., Ltd.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014. Question answering with subgraph embeddings. *In Proc. EMNLP*, pages 615–620.

Arthur Graesser, Carol Forsyth, and Blair Lehman. 2017. Two heads may be better than one: Learning from computer agents in conversational trialogues. *Teachers College Record*, 119(030302).

Daisuke Kawahara and Sadao Kurohashi. 2006. A fully-lexicalized probabilistic model for japanese syntactic and case structure analysis. In *Proc. HLT-NAACL*, HLT-NAACL '06, pages 176–183.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *ICML*, pages 1188–1196.

Akinobu Lee et al. 2013. Mmdagent—a fully open-source toolkit for voice interaction systems. In *IEEE-ICASSP*, pages 8382–8385. IEEE.

Yi Luan et al. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. *In Proc. EMNLP*, page 32193232.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Hajime Morita et al. 2015. Morphological analysis for unsegmented languages using recurrent neural network language model. In *Proc. 2015 EMNLP*, pages 2292–2297.

Daisuke Sakamoto et al. 2009. Humanoid robots as a broadcasting communication medium in open public spaces. *IJSR*, 1(2):157–169.

Hiroaki Sugiyama, Toyomi Meguro, Yuichiro Yoshikawa, and Junji Yamato. 2018. Avoiding breakdown of conversational dialogue through inter-robot coordination. In *Proc. the 17th AAMAS*, pages 2256–2258. AAMAS.

William Swartout et al. 2010. Ada and grace: Toward realistic and engaging virtual museum guides. In *Proc. the 10th IVA*, pages 286–300.