Sensors and Materials, Vol. 33, No. 9 (2021) 3101–3123
MYU Tokyo

S & M 2676

Ontology-based Integration of Knowledge Base for Building an Intelligent Searching Chatbot

Hien D. Nguyen,[1,2*] Tuan-Vi Tran,[1,2] Xuan-Thien Pham,[1,2]
Anh T. Huynh,[2,3] and Nhon V. Do[4]

[1]Faculty of Computer Science, University of Information Technology,
Quarter 6, Linh Trung Ward, Thu Duc City, Ho Chi Minh City 700000, Vietnam
[2]Vietnam National University, Ho Chi Minh City (VNU-HCM),
Quarter 6, Linh Trung Ward, Thu Duc City, Ho Chi Minh City 700000, Vietnam
[3]Faculty of Software Engineering, University of Information Technology,
Quarter 6, Linh Trung Ward, Thu Duc City, Ho Chi Minh City 700000, Vietnam
[4]Department of Information Technology, Hong Bang International University,
215 Dien Bien Phu, Binh Thanh District, Ho Chi Minh City 700000, Vietnam

(Received January 11, 2021; accepted June 14, 2021)

Keywords: intelligent chatbot, knowledge base, ontology, e-learning

A chatbot is a useful tool for communicating with users to extract necessary information. An intelligent chatbot requires an effective knowledge base as materials to adequately organize the knowledge domain and process many kinds of inputted queries as natural text. Ontology technology is effective for use in learning technology systems. In this paper, a model that uses ontology technology for relational knowledge to integrate a structure of scripts is presented. This integrated model, called the Rela-Scripts model, is used to organize the knowledge material of a chatbot to search for knowledge in education. Some problems in searching for knowledge by this chatbot are proposed and solved on the basis of the Rela-Scripts model. The proposed method is applied to build an intelligent chatbot for answering questions on contents of the Introduction to Programming course in a university. This chatbot acts as a tutor by communicating with students in Vietnamese and gives explanations that meet the requirements of students. Its instructions are useful for their self-learning to enhance their programming skills.

1. Introduction

Dialog systems, also known as chatbots, play an essential role in the fourth industrial revolution. There are typically two kinds of dialog systems: chit-chat systems for instructing and task-oriented dialog systems.(1,2) A task-oriented dialog system is a chatbot that people can communicate with in natural language via text. An intelligent chatbot has a knowledge base to detect the purpose of users to help them complete specific tasks. The chatbot can also retrieve suitable knowledge from its knowledge base to answer or explain inputted queries.(3)

Intelligent chatbots have applications in many domains. There have been many studies to develop intelligent chatbots as personal assistants, such as Cortana (Microsoft),(4) Google Assistant,(5) Siri (Apple),(6) and Alexa (Amazon).(7) These systems answer a wide range of

*Corresponding author: e-mail: hiennd@uit.edu.vn
https://doi.org/10.18494/SAM.2021.3264

ISSN 0914-4935 © MYU K.K.
https://myukk.org/

questions. They also predict user needs and recommend useful services in response to user requests. Moreover, conversation systems have also been applied to provide information about public services, such as Chatbot 1022 in Danang City (Vietnam),(8) airport chatbots,(9) intelligent chatbots providing a method of solving mathematical problems, such as Mathway,(10) and tutoring chatbots for high-school mathematics.(11,12) Some communication chatbots are designed by using information retrieval (IR)-based models that mine human chats or human–machine chats, such as XiaoIcle(13) and Cleverbot.(14) These chatbot models choose a response by taking the user's query and finding a (tf-idf) similar query in the dataset;(15) after that, they take whatever the response was to that query.

In this paper, a model integrating an ontology for the knowledge of relations and the structure of scripts, called the Rela-Scripts model, is proposed. Ontology technology is an effective way to represent knowledge domains in the real world. The ontology Rela-model is a model for the knowledge of relations.(16) The structure of the Rela-model includes concepts, relations between them, and a set of inference rules of the knowledge domain. The Rela-model is integrated with the structure of scripts, which store common conversations between chatbots and users. The integration is based on the combination of concepts, facts, and internal rules of the Rela-model and scripts. This integrating model is a useful way to organize the knowledge base, which provides knowledge materials for building an intelligent chatbot that can search for knowledge. Some problems in searching for knowledge using this chatbot are proposed: classifying the inputted query, retrieving the knowledge content based on the matching of keywords, and searching for related knowledge to provide the results of a query. These problems are modelized on the basis of the structure of the Rela-Scripts model and the algorithms designed for solving them. The built chatbot is equipped with a knowledge base to increase its search ability based on practical knowledge content. It can communicate as a tutor by supplying required knowledge to users.

E-learning is a modern method of learning in the era of information technology (IT).(17) The proposed method is applied to build an intelligent chatbot to answer questions on the contents of the Introduction to Programming course in a university. This chatbot acts as a tutor by communicating with students in Vietnamese and gives explanations that meet the requirements of students. Its instructions are useful for their self-learning to enhance their programming skills. This designed chatbot has been tested on IT students in universities, who gave the chatbot positive feedback.

Section 2 presents related works on constructing intelligent chatbots. In Sect. 3, we propose a knowledge model of a chatbot based on integrating the ontology Rela-model and the structure of scripts. This model helps a system communicate with users to retrieve necessary contents from its knowledge base for them. Some problems in querying knowledge using the intelligent chatbot are proposed and solved on the basis of the integrating model in Sect. 4. These problems are classifying queries, searching the knowledge content to match the query's meaning, and recommending some knowledge related to the search content. Section 5 presents the architecture of a question-answering chatbot based on its knowledge base. It is applied to design an intelligent chatbot to support learning in an Introduction to Programming course. This chatbot was also experimented on using IT students in universities. Section 6 summarizes the results of this paper and proposes future works.

2. Related Work

The main goal of an intelligent chatbot is to be a virtual companion to users by providing necessary information in response to their requests.(18) Nowadays, chatbots have become popular in society due to the development of AI technologies, for example, natural language processing,(19) IR,(20) knowledge engineering,(12,16,21) and empathic conversational systems.(13,22) However, these chatbots are not effective in searching for educational knowledge to support studying based on the semantics of queries inputted through a conversation.

XiaoIcle is an empathetic social chatbot that can detect the feelings of humans and respond to user needs dynamically.(13) Its architecture is integrated with an intelligence quotient (IQ) and emotional quotient (EQ), so it has very strong communication skills. However, this bot is designed as a social bot and does not have pedagogical characteristics when searching for knowledge to support students.

The framework of a chatbot for detecting objects from cameras was proposed in Ref. 23. This chatbot can also update a dataset for retraining by transfer learning using more cases taken from surveillance cameras. However, it does not have a knowledge base to solve problems automatically.

In education, chatbots are useful tools for students to make inquiries when applying to university,(24) to tutor students how to solve exercises,(11,12) and to search for knowledge required by students.(25) The educational applications have pedagogical criteria to supply knowledge useful for courses. They require a knowledge base organized on the basis of the course syllabus. To respond to queries on knowledge related to the contents of courses, the chatbot can communicate with students naturally as a tutor or explain lacking knowledge to students. It can retrieve knowledge that matches the requests from users through natural queries. To communicate with users smoothly, the chatbot must be equipped with natural language processing to analyze the meaning of inputted queries,(18,19) from which it can extract suitable contents that match requests from users. Moreover, from the meaning of the queries, the chatbot can recommend further knowledge related to the content found by searching. This related knowledge can provide students with greater understanding about the search content.

There are many models for representing knowledge domains,(26,27) especially knowledge in education.(28) Nonetheless, they are not suitable for organizing a knowledge base of a chatbot because they do not include any component to process the conversation between the user and chatbot. Ontology is an effective method to represent practical knowledge.(29) Moreover, there are also methods of ontology integration that have been applied in knowledge domains related to medicine(30) and education.(31) The semantic sensor network ontology was also used to extract new knowledge through K-means clustering from dynamic IoT data.(32) However, the current integrating methods cannot be used to organize the knowledge obtained from conversations with chatbots.

The intelligent searching chatbot in this study has a knowledge base supporting its conversation as well as the ability to search for knowledge about courses, which is suitable for students. The knowledge base of the proposed chatbot is built on the basis of a model integrating the ontology of relational knowledge and the structure of scripts.

3. Design of Knowledge Base of Intelligent Chatbot

The knowledge base is an important component of an intelligent system. It has to represent the knowledge domain sufficiently. In an intelligent chatbot, the knowledge base needs to have communication scripts to interact with the user naturally. When a query to retrieve knowledge about programming is inputted, it will be analyzed and classified into question forms that can be understood by the chatbot. On this basis, the system will connect to its knowledge base to extract suitable knowledge for the inputted query. Therefore, the most important point is to choose a model to build a knowledge base for the intelligent chatbot. In this section, a knowledge model integrating an ontology for the knowledge of relations and the structure of scripts is presented. This model is useful for organizing the knowledge base of a searching chatbot.

3.1 Ontology for knowledge of relations

The Rela-model is an ontology including concepts, relations between concepts, and inference rules of the knowledge domain.(16) The structure of the ontology Rela-model is suitable for building an intelligent searching system, especially a chatbot for querying course-based knowledge. Moreover, this model must be integrated to fit the conversation process of a chatbot; thus, the structure of the relations between concepts in this ontology also needs to be improved. The improved model is applied to design the knowledge base of the chatbot-based searching system.

Definition 3.1:(16) *Knowledge model of relations*

The Rela-model, which is a knowledge model of relations, is expressed as the following triple:

$$\mathcal{K} = (\mathbf{C}, \mathbf{R}, \mathbf{Rules}),$$

where **C** is the set of concepts about the knowledge of programming in the course, **R** is the set of relations between the concepts in **C**, and **Rules** is the set of inference rules in the knowledge domain.

C – set of concepts

Each concept $c \in \mathbf{C}$ is a class of objects. It has the structure (*Attr*, *Inner-relation*, *Properties*, *Key*), where *Attr* is a set of attributes of a concept, *Inner-relation* is the set of relations between the attributes in *Attr*, *Properties* is the set of properties of the corresponding concept, and *Key* is the set of keywords related to the corresponding concept.

Example 3.1: The knowledge domain for the course of Introduction to Programming has the following concepts:

• ALGORITHMS – the concept of algorithms

Attr = {Name, Chapter, Content}

– Name: The name of an algorithm

– Chapter: The chapter that the algorithm belongs to

– Content: The definition and content of the algorithm

Inner-relation = {*belong to*}

– Name *belong to* Chapter

Properties = {Operators}

Key = set of keywords of the algorithm

• POINTER – the concept of pointers

Attr = {Name, Chapter, Content, Expression, Usage}

– Name: The name of the pointer

– Chapter: The chapter that the pointer belongs to

– Content: The definition and content of the pointer

– Expression: The set of pointers in the lesson

– Usage: How to use of pointer

Inner-relation = {*belong to*}

– Name *belong to* Chapter

Properties = {Static memory allocation, Dynamic memory allocation, New, Delete}

R – set of relations

There are two kinds of relations between concepts in **C**:

$$\mathbf{R} = \mathbf{R}_{\mathbf{hierachy}} \cup \mathbf{R}_{\mathbf{related\_to}}.$$

Here,

• $\mathbf{R}_{\mathbf{hierachy}}$ is a set of hierarchical relations between concepts and can be considered as the inheritance relations. They are relations such as "is_a".

c_i is_a c_j: This means c_i is a sub-concept of concept c_j (c_i, $c_j \in \mathbf{C}$)
and $c_j.Attr \subseteq c_i.Attr$.

Example 3.2:

– "High level programming language" is_a "Programming language"

– "Multi-dimensional array" is_a "Array", "Multi-dimensional array" has all of the attributes of an "Array" such as Name, Usage, etc.

• $R_{\text{related\_to}}$ is a set of knowledge related to the main knowledge. It helps to deduce the related knowledge when the machine operates the proof layer.

c_i related_to c_j: This means c_i has a relationship with c_j (c_i, $c_j \in \mathbf{C}$).

Example 3.3:

– "Program" related_to "Programming language"

– "Statement" related_to {"DataTypes", "Key word", "Typedef"}. This means that

* "Statement" related_to "DataTypes"

* "Statement" related_to "Key word"

* "Statement" related_to "Typedef"

Rules – set of rules

Definition 3.2: *Kinds of facts*

For a knowledge model $\mathcal{K}$ as Rela-model, the following facts hold:

1. $o : c$ ($c \in \mathbf{C}$, o is an object) : o is an object of concept c
2. Determine(o) ($c \in \mathbf{C}$, $o \in \mathrm{I}_c$) : An object o is determined
3. Determine($c.a$) ($c \in \mathbf{C}$, a is an attribute of a component of c)
4. $c_1\ \Theta\ c_2$ ($c_1, c_2 \in \mathbf{C}$, $\Theta \in \mathbf{R}$) : the relation Θ holds between two concepts c_1 and c_2

Each rule $r \in$ **Rules** is an inference rule with the form $u(r) \rightarrow v(r)$, where $u(r)$, $v(r)$ are sets of facts. Facts are concrete statements about "properties of relations", "relations between concepts", and "relations between attributes of concepts".

Example 3.4: $\forall\ c_1, c_2 \in \mathbf{C}$:

- r_1: $\{c_1$ related_to $c_2\} \rightarrow \{c_2$ related_to $c_1\}$
- r_2: $\{c_1$ is_a $c_2\} \rightarrow \{c_2.Attr \subseteq c_1.Attr\}$

3.2 Build the scripts for chatbot

In a question-answer system, there are some common conversations. They have a general script that allows conversation with users. According to a survey to discover what students are interested in when they use a tutoring chatbot, there are three main problems for which help is required: definition of a concept, how to apply knowledge in practice, and knowledge related to the search result. In this section, the structure of scripts and the process followed to construct scripts are proposed.

Definition 3.3 *Structure of a script.*

The structure of a script representing a question-answer process includes five elements:

$$(Name, Meaning, Ques, Ans, InnerRul)$$

Here:

- *Name*: The name of the script.
- *Meaning*: The content of the corresponding script.
- *Ques* = $[q_1, ..., q_m]$: The list of questions for the corresponding script.
- *Ans* = $[a_1, ..., a_m]$: The list of answers for the corresponding script, where each a_i is an answer to question $q_i \in Questions$ $(1 \leq i \leq m)$.

These questions in *Ques* and answers in *Ans* will be manually defined on the basis of statements of the corresponding script.

- *InnerRul*: This is a set of deductive rules to support the selection of questions in the script for the chatbot. Each rule $r \in InnerRul$ has the form $u(r) \rightarrow v(r)$, where $u(r)$ is the set of facts and $v(r)$ is the result of selecting a question or an answer.

Example 3.5: Script for searching for the definition of a concept by the intelligent chatbot (see Table 1):

Table 1
Script for searching for the definition of a concept by the intelligent chatbot.

Name	*Meaning*	*Ques*	*Ans*	*InnerRul*
Searching for definition of a concept	Conversation about searching for a pointer	q_1: What is the definition of a pointer?	a_1: A pointer is a variable in C++ that holds the address of another variable. The pointer NULL has a relation to a pointer.	r_1: $\forall\ i$, $1 \le i \le 4$ {q_i.answer = “Yes”} → {**show** a_i} r_2: {q_2.answer = “No”} → {**goto** q_4} r_3: {a_4.answer has a choice} → {**goto** q_1 with that choice}
		q_2: Do you want to practice with some practice questions?	a_2: Write a function that will take a pointer and display the number on the screen. Take a number from the user and print it on the screen using that function.	
		q_3: Do you want to know about anything else?	a_3: Write a program to find the factorial of a number using pointers.	
		q_4: Do you want to search for some other knowledge related to pointers?	a_4: Show knowledge related to pointers.	

The scenarios for searching for information by the intelligent chatbot are as follows:

Scenario 1: Collect information and classify kinds of questions.

Scenario 2: Determine results of the concept or attributes of the main knowledge.

Scenario 3: Return the practice test.

Scenario 4: Recommend related knowledge as a question.

As shown in Fig. 1, when a user begins the conversation, they can greet the chatbot or ask a question. Then, the system returns the answer to the question and asks the user if they want related knowledge. The system also recommends useful definitions for users to understand more about their search contents. To end a communication script, the system can ask questions about practice tests or it can recommend other knowledge related to the content of some query results.

3.3 Integrating knowledge base for an intelligent chatbot

To organize the knowledge base of an intelligent chatbot, in addition to representing the knowledge domain, it is necessary to combine the conversation scripts between the chatbot and user. The knowledge model combines the ontology Rela-model for the knowledge of relations(16) and the structure of scripts described in Def. 3.2.

Definition 3.4: The model integrating the ontology Rela-model and scripts, called the *Rela-Scripts model*, has the structure **(C**, **R**, **Rules) + Scripts**.

Here, the (**C**, **R**, **Rules**) ontology used as the Rela-model represents the knowledge domain about the relations given in Def. 3.1. The **Scripts** component is a set of scripts with the structure described in Def. 3.2, where each element in **Scripts** is a conversation frame for practical searching.

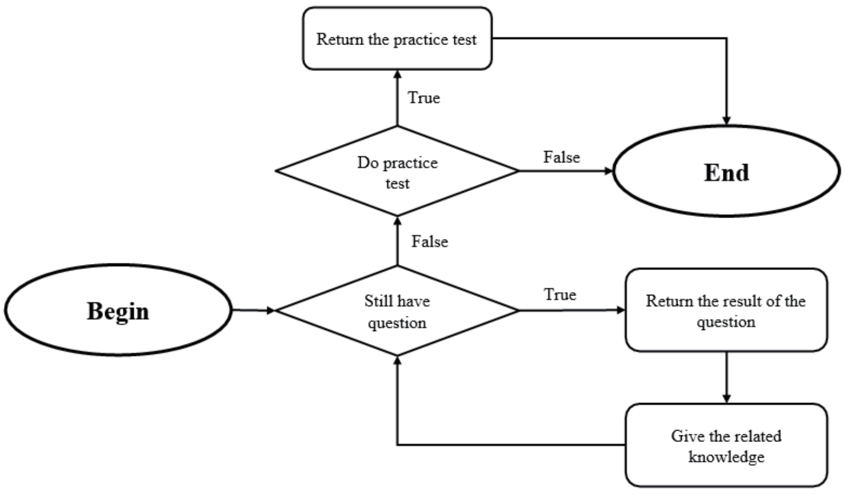


Fig. 1. Construct the script.

This integrated model can organize the knowledge base of an intelligent chatbot for knowledge querying. The integration is based on the following requirements:(31)

- Building the connection between scripts and concepts: Through the meaning of the script, concepts of the knowledge domain related to the corresponding script are extracted on the basis of keyword components of concepts and scripts.
- Adding more facts: There are facts that are related to the content of scripts that need to be connected with the former facts in the Rela-model.
- Adding more rules from the scripts: Some rules appear when integrating the knowledge model. They need to be merged into the integrated model.

Definition 3.5: Let K be a knowledge domain in the integrated ontology Rela-Scripts model or a component of the Rela-Scripts model. For an object o, we define the following:

Component(K) is a set of sub-components of K.

keywords(o) is a set of keywords of o with $o \in k$ and $k \in Com(K)$.

Definition 3.6: Given a knowledge domain $\mathcal{K} = (\mathbf{C}, \mathbf{R}, \mathbf{Rules}) + \mathbf{Scripts}$ as the Rela-Scripts model, a concept $c \in \mathbf{C}$, and a script $s \in \mathbf{Scripts}$, Table 2 presents the integrated structure of **Scripts**.

Table 2
Integrated structure of **Scripts**.

Relations between script and concept	Adding facts to Rela-Scripts model	Adding rules to Rela-Scripts model
Structure of a script: $s := (Name, Meaning, Ques, Ans, InnerRul)$ The script s is *related to* the concept c $\Leftrightarrow keywords(s.Meaning) \cap c.Key \neq \varnothing$, where *keywords*(*text*) is a set of keywords of the *text*.	S1. *Determine a pair comprising a question and answer* Specification: (q_k, a_k) Condition: $q_k \in s.Ques$, $a_k \in s.Ans$, $s \in \mathbf{Scripts}$ S2. *Show a question or an answer* Specification: Show(u) Condition: $u \in s.Ques \cup s.Ans$, $s \in \mathbf{Scripts}$	r is a fully deductive rule. It has the form $u(r) \rightarrow v(r)$ and $\neg u(r) \rightarrow h(r)$, where $u(r)$ is the set of facts and $v(r)$ and $h(r)$ are the results of choosing a question or an answer.

4. Some Problems in Querying Knowledge Using Intelligent Chatbot

When a query or a text is inputted to the chatbot, which may have to retrieve knowledge about programming, the intelligent chatbot will carry out processing to extract some main keywords. After that, these keywords are compared to match with the content in the knowledge base of the search system based on their semantics. The matching is performed by comparison with the structure of the knowledge model's components, especially the concepts and relations. The inference rules of the knowledge model help to deduce more relationships related to the content of the query. Finally, the chatbot shows the retrieval results for the inputted query. The following are some of the main problems in querying knowledge about programming in the search system:

(1) **Problem 1: Classifying the inputted query**. From the query inputted as Vietnamese text, the problem is to extract the main keywords of the query. These words are used by the chatbot to choose a suitable conversation script and retrieve knowledge results from its knowledge base.
(2) **Problem 2: Retrieve the knowledge content based on matching the keywords**. With the set of keywords extracted from the query, a method to compare the similarity between the meaning of the keywords and the content in the knowledge base is proposed. This method determines the content of the required knowledge for the inputted query.
(3) **Problem 3: Searching the related knowledge for the results of a query**. From the query or the text inputted by the user, the system will retrieve the knowledge to answer the query and communicate with the user. The system also detects some knowledge related to the obtained answers.

4.1 Classifying the inputted query

The collective dataset includes a set of user intents and facilitates analysis of out-of-scope (OOS) queries: queries that users may reasonably make but fall outside of the scope of the system-supported intents.(33) The OOS queries are issues that are not expected but are unavoidable in dialog systems because most users are not fully aware of the ability of the system, which is limited to specific intents. The processing of OOS queries is necessary to avoid incorrect responses and find ways to update the set of intent classes.

4.1.1 Data collection

There were 932 queries collected for the dataset: 802 in-scope queries covering six intent classes and 130 OOS queries. Table 3 classifies the training queries and the results for training them in the collected dataset.

In-scope data were collected from the University of Information Technology (UIT) forum and from IT students studying the Introduction to Programming course at UIT. The intents were grouped on the basis of the goals of the chatbot, which were determined before. OOS data were utilized when a student made a mistake in creating data, i.e., the query did not actually belong to

Table 3
Classification of queries.

Class	Meaning	Number of training queries	Number of tested queries	Example (in Vietnamese)	English translation of example
Greeting	Sentences for greeting	10	5	Chào bạn	Hi, you
Bye	Sentences for ending communication	11	5	Hẹn gặp lại	See you again
Concept	Require definition or attributes of a concept	274	106	Tôi muốn biết về hàm	I want to know about a function
Setting	Explain how to do something	100	65	Làm thế nào để code hàm?	How do I do functional programming?
Comparison	Compare two concepts	245	97	Sự khác nhau giữa kiểu “số nguyên” và “số thực” là gì?	What is the difference between “int” and “float”?
Related knowledge	Require related knowledge	20	10	Còn gì nữa không?	What is something else?
OOS		100	30	Hôm nay trường mấy giờ mở cửa?	What time does school open today?
Total		760	318		

one of the six intent classes. OOS data were also collected from online platforms such as Quora and freeCodeCamp.

4.1.2 Methodology

In this section, we discuss the intent classification stage. Each approach combines a sentence representation and a classification algorithm. The classification of the intent determines the purpose of a user.

The utterance $s \in S$ of a dialog is given, where S is the utterance space. The intent classes are $C = \{c_1, c_2, ..., c_M\}$ and the training set is $D = \{d_i, c_i\}_{i=1}^{N} (M < N)$, $(s, c) \in S \times C$. Each utterance s belongs to only a label $c \in C$. For example:

(s, c) = {“Sự khác biệt giữa mảng số nguyên và số thực là gì?”, “so sánh”}

(Meaning: (s, c) ={“What is the difference between int and float?”, “comparison”})

The Subword Semantic Hashing method is used to provide the features input to classify an intent.(34) This method has better performance than most natural language understanding (NLU) services and other platforms, such as Botfuel, Dialogflow, Luis, Watson, Rasa, Recast, and Snips. This method extracts part of the words from a sentence to generate SemHash tokens as features, then it uses these vectorized features for training and testing (see Algorithm 4.1).

For example, the input utterance T = “*Tôi muốn biết về con trỏ*” (I want to know about pointers) is split into the list of words t_i = [“*Tôi*”, “*muốn*”, “*biết*”, “*về*”, “*con*”, “*trỏ*”]. Each word is parsed into a pre-hashing function $\mathcal{H}(t_i)$ to generate sub-token t_i (j is the index of the sub-token). $\mathcal{H}(t_i)$ adds token “#” to the head and tail of each word and then extracts sub-tokens t_i^j (trigrams) from it, such as $\mathcal{H}$(*muốn*) = [*#mu*, *muố*, *uốn*, *ốn#*]. $\mathcal{H}(t_i)$ is applied as a vector to the entire dataset for generating sub-tokens. It is used to extract features and is considered as a hashing function for an inputted query.

Algorithm 4.1
Subword Semantic Hashing.

```
queries ← user utterances;
sub-tokens ← { };
examples ← [ ];
for each query $q \in$ queries do
{
  example ← [ ];
  tokens ← split $q$ into words;
  for each token $t$ in tokens do:
  {
    $t$ ← cat("#", $t$, "#");
    for $j$ from 0 to length($t$) − 2 do:
    {
        sub-tokens := sub-tokens $\cup$ {$t[j{:}j + 2]$};
        example := [op(example), $t[j{:}j + 2]$];
    }
  }
  examples.append(example)
}
return (sub-tokens, examples)
```

The stage of query processing represents the text in the form of fixed-length numerical vectors for consideration as an input to the classifiers. Subword Semantic Hashing(34) is a bag of *n*-gram SemHash tokens that form a matrix with rows and columns representing the queries and the SemHash tokens in the dataset of queries.

The following classifiers are implemented: ridge classifier, passive aggressive classifier, random forest classifier, linear support vector classifier (SVC), stochastic gradient descent classifier (SGD), and nearest centroid. The ridge classifier, passive aggressive classifier, linear SVC, and nearest centroid classifier are implemented with default parameters. A grid search is used to find the best hyperparameters for the random forest classifier and SGD by performing fivefold cross-validation on the training set. Finally, the model with the best average validation score is applied in the testing set. The chosen model belongs to the contents of the knowledge domain and the training set.

4.2 Searching for knowledge content based on matching keywords

After extracting the intents and keywords of the utterance, the system matches the keywords with the content of the knowledge for defining and comparing texts based on the Rela-model in Sect. 3. The technique for matching by the search engine is shown in Fig. 2.

As illustrated in Fig. 2, the processing text for classifying the inputted query was presented in Sect. 4.1. This process solves problem 1 of extracting main keywords or key phrases of a query. They are used to retrieve the knowledge content in problem 2. These words are materials that are compared to choose the conversation script of the chatbot. From the results of the conversation, the chatbot can retrieve suitable knowledge content for the query from its knowledge base as the ontology.

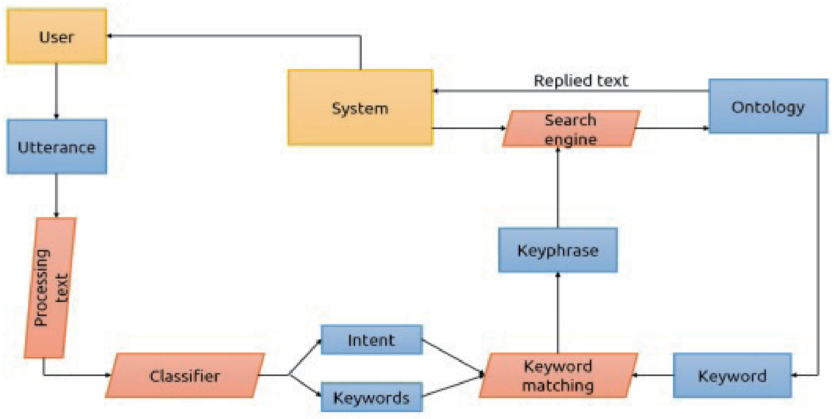


Fig. 2. (Color online) Technique for matching by the search engine.

The dictionary is the set of keywords in the knowledge domain of the course. In the first step, the dictionary is established from the knowledge base of the search system. It also contains question words that help to classify queries.

Example 4.1: Some words in the dictionary of the Introduction to Programming course.

- Some individuals in the dictionary: Array, Algorithm, Define, Variable, Usage.
- The synonyms of a keyword in the dictionary: “How to use” is equivalent to “Usage”, and “What is” is equivalent to “Define”.

Next, the extracted keywords of the utterance are compared with the dictionary to create a set of key phrases. The search engine uses these key phrases to retrieve the knowledge of programming from the ontology. Moreover, from the retrieval results, the system also searches for related knowledge based on the relations of these results and the inference rules of the knowledge base. Finally, all results are returned in the form of a document for each kind of intent.

The search for the knowledge content returns a set of knowledge based on the meaning of an inputted query. The system determines the meaning of this query from its extracted keywords. This search is performed by determining keywords of the query with similar meanings and using stored knowledge. There are two problems in searching for the knowledge content based on the matching of keywords.(17)

Problem 4.1: Given a knowledge domain $\mathcal{K}$ as an integrated ontology Rela-Scripts model and a query q, determine a set of keywords W that are the words most similar to the meaning of query q.

$$\text{Denote: } W := Meaning(q)$$

Problem 4.2: Given a knowledge domain $\mathcal{K}$ as a Rela-Scripts model and a query q, determine a set of knowledge content matching the meaning of query q.

Problem 4.1 can be solved by using the method in Sect. 4.2 by employing a measure of the similarity of a meaning. This measure is computed using the tube ($tf(w, q)$, $idf(w, K)$), where *tf*(w, q) is the term representing the frequency of keyword w in query q, and *idf*(w, K) is an inverse knowledge frequency representing the specificity of keyword w in knowledge K. The formulas of (*tf*(w, o), *idf*(w, K)) are established as follows:[35]

$$f(w,q) := \mu + (1-\mu)\frac{n_{w,q}}{\max\{n_{w',q} \mid w' \in keywords(q)\}}, \tag{1}$$

$$idf(w,K) := \log\left(\frac{\mathrm{card}\left(\bigcup_{o\in Component(K)} keywords(o)\right)}{1+\mathrm{card}\left(\{o \in Component(K) \mid w \in keywords(o)\}\right)}\right), \tag{2}$$

where $n_{w,o}$ is the number of occurrences of keyword w in object o, $\mu \in [0, 1]$ is a parameter that is the minimum value for every keyword. Given a knowledge domain $\mathcal{K}$ as a Rela-Scripts model and a query q, the following Algorithm 4.2 finds a set of knowledge content that matches the meaning of query q.[25]

```
Input: Knowledge domain $\mathcal{K}$ as Rela-Scripts model.
    Query $q$.
Output: Set of knowledge content appropriate for the meaning of query $q$.
Algorithm 4.2:
    Step 1:
        Extract keywords that are closest in meaning to words in $W$ of a query sentence.
            $W := Meaning(q)$
        $Knowledge$ := {} // set of query results.
        If (there exists a keyword for the type of knowledge in $W$) then
            $Search$ := set of components of $\mathcal{K}$ related to the type of knowledge.
    Step 2:
        Retrieve knowledge from components in $Search$ based on keywords in $W$.
        Update $Knowledge$.
    Step 3:
        Unification of facts in the knowledge model and compare their meaning by solving Problem 4.1.
        Update $Knowledge$.
    Step 4: Return results in $Knowledge$.
```

4.3 Searching for related knowledge

Searching for related knowledge is an important part of processing a knowledge query by an intelligent chatbot. After querying the question, the system should be able to define the target of the question. From the query results, the system searches for related knowledge based on the relations between these results and the inference rules of the knowledge base. However, we want the system to understand which knowledge conforms to the scenarios. Finally, all results are returned in the form of a document for each kind of query. In this paper, the related knowledge is a set of knowledge related to the current search results in the ontology Rela-model representing the knowledge domain.

For a knowledge domain $\mathcal{K}$ as a Rela-Scripts model, a query q, and the results of query q, the following Algorithm 4.3, which was presented in Ref. 25, determines a set of knowledge related to the results of query q.

```
Input: Knowledge domain $\mathcal{K}$ as Rela-Scripts model and query $q$.
Output: Set of knowledge related to results of query $q$.
Algorithm 4.3:
    Step 1:
        Knowledge := set of knowledge results for query $q$ determined using Algorithm 4.2.
    Step 2:
        For each $o$ in Knowledge do
            Determine the set of knowledge related to $o$
                    $\text{Related}(o) := \{o' \mid \exists\, r \in \mathcal{K}.\mathbf{R},\ r(o, o')\}$        (3)
    Step 3:
        Determine the set of knowledge related to the Knowledge set:
                    $\text{Relate}(Knowledge) := \bigcup_{o \in Knowledge} \text{Related}(o)$        (4)
        Classify the knowledge in Relate(Knowledge) based on types of knowledge of integrating
            ontology Rela-Scripts model.
    Step 4:
        Show results in Relate(Knowledge) by its classification as communication scripts of the chatbot's knowledge
            base.
```

5. Design of Intelligent Chatbot for Answering Questions about Knowledge of Programming

Programming is an important skill in STEM education. This knowledge is the foundation of the curriculum of IT. The Introduction to Programming course is a requirement for programming and supplies the basic concepts and skills required by beginner programmers.

The knowledge in the course is obtained from books(36,37) and covers the concepts, relationships, and rules of the course. This collected knowledge can be classified in several ways such as chapters, topics, issues, and assignments in the course. Some examples of such knowledge are databases of course knowledge. Based on the collected and database knowledge, the ontology represents knowledge of the course, which is represented using the structure of the Rela-model. This ontology is the knowledge base of the search system. The problems in querying knowledge are solved using this knowledge base. The search engine fetches the results for the entered query.

5.1 Architecture of question-answering chatbot based on knowledge base

The question-answering chatbot has a knowledge base organized on the basis of the Rela-Scripts model in Sect. 3.3. The problems that the search engine of this chatbot encounters when querying its knowledge base were described in Sect. 4. The architecture of the chatbot is illustrated in more detail in Fig. 3.

After receiving a user request, the chatbot must acquire the intents and keywords of the user query. We refer to these as the intents, and several intents appear such as "compare" and

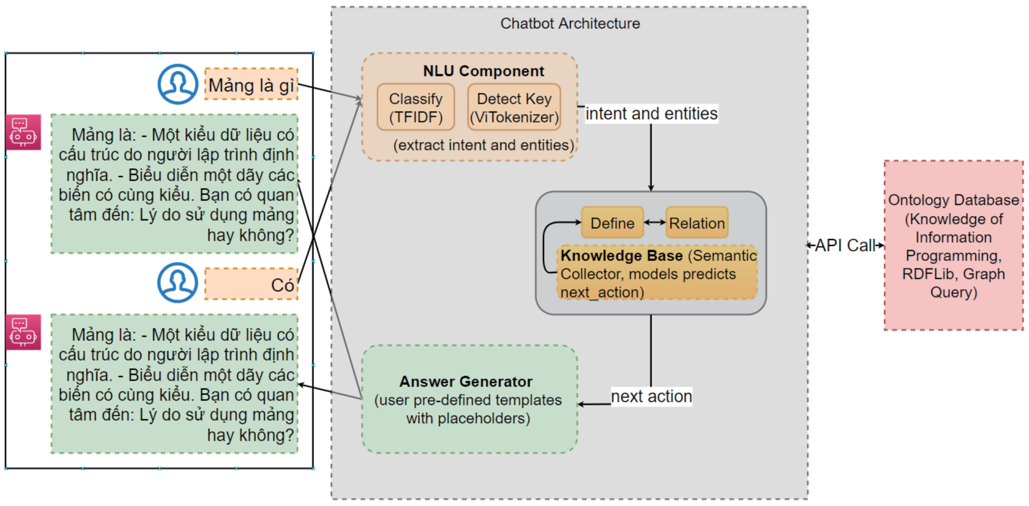

Chatbot Architecture
Mảng là gì
Mảng là: - Một kiểu dữ liệu có cấu trúc do người lập trình định nghĩa. - Biểu diễn một dãy các biến có cùng kiểu. Bạn có quan tâm đến: Lý do sử dụng mảng hay không?
Có
Mảng là: - Một kiểu dữ liệu có cấu trúc do người lập trình định nghĩa. - Biểu diễn một dãy các biến có cùng kiểu. Bạn có quan tâm đến: Lý do sử dụng mảng hay không?
NLU Component
Classify (TFIDF)
Detect Key (ViTokenizer)
(extract intent and entities)
intent and entities
Define
Relation
Knowledge Base (Semantic Collector, models predicts next_action)
API Call
Ontology Database (Knowledge of Information Programming, RDFLib, Graph Query)
next action
Answer Generator (user pre-defined templates with placeholders)

Fig. 3. (Color online) Architecture of the search system of the chatbot based on the knowledge domain.

"concept". The intent in the above example is "concept". Then, the chatbot must find the determined intents, called the *entities*, in the utterance. A few examples of entities are "câu lệnh" (sentence), "từ khóa" (keyword), and "kiểu dữ liệu" (data type).

In Fig. 3, the gray box describes the NLU component. It supports the retrieval of the intents and the entities of the user including the keywords from the user query.

The NLU component comprises the following:

- A supervised ridge classifier, an SGD, etc., and an intent classification model trained with a dataset including diverse sentences as the input and the intents as the output.
- Entity extraction model – The chatbot is trained using conditional random field (CRF) probabilistic models.

The ontology gives the results of the API calling and predicts an appropriate response using RDFLib to query the graph of the ontology. This information constitutes the input X, the feature vector, and the target y. The dialog model is trained using the '*next_action*', which is a one-hot encoded vector corresponding to each action defined in the training data.

The Message Generator component consists of many patterns that users can adjust (templates including sentences with some placeholders, if applicable), which are related to the action name. Thus, depending on the output predicted by the dialog manager, the corresponding sample message is sent. If the request template fills some placeholder values, these values are also passed through the generator by the conversation manager. A suitable message is then given to the user, and the bot continues its standby status while listening for the user's input.

5.2 Testing and experimental results

The intelligent chatbot for learning the knowledge in the Introduction to Programming course has a user interface as shown in Fig. 4. The user can communicate with this chatbot to search for contents in the course. Moreover, the user also retrieves knowledge, such as definitions, examples, exercises, or practice tests, that is related to the current search content.

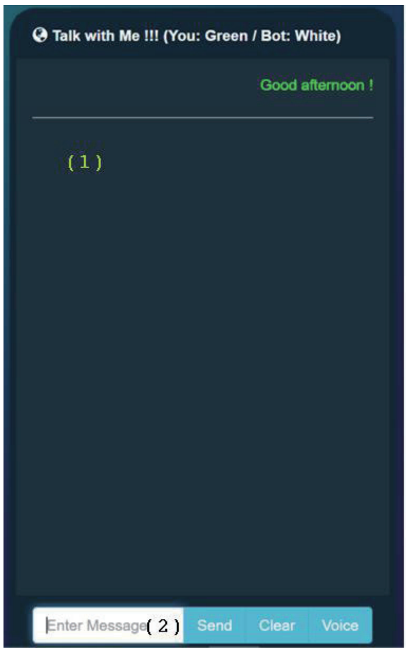


Fig. 4. (Color online) User interface of chatbot to answer questions on knowledge in Introduction to Programming course. (1) Frame for question-answer process. (2) Frame for user to type message.

The performance of different classifiers in classifying inputted queries based on the collected dataset is shown in Table 4, where the F1-score is used for evaluation. This score is calculated from the numbers of true positives (*TP*), false negatives (*FN*), true negatives (*TN*), and false positives (*FP*) for the test set.

Using the ridge classifier, the chatbot can communicate fluently by inputting natural text. Step-by-step answers are given to the user so that they can follow them more easily. The chatbot acts as a tutor by explaining the content of queried knowledge. Example 5.1 shows the process by which a question about an array is answered (in Vietnamese).

Example 5.1: Process by which chatbot answers question about an array (Fig. 5).

The designed chatbot covers the necessary knowledge to learn the fundamentals of programming. Its tutoring is suitable for helping beginners to learn programming, and its tutoring approach is similar to that of an advisor. However, there are still cases in which the responses are not natural and situations in which the chatbot is not flexible, and the chatbot needs to be able to deal with more kinds of queries. Table 5 shows the testing results of the chatbot for the six kinds of queries mentioned in Table 3.

The main ability of the chatbot is to search for knowledge in the Introduction to Programming course to meet users' requirements when they ask about a concept, an implementing method, practice exercises, a comparison between two concepts, or related knowledge. The knowledge content of this course includes the following:

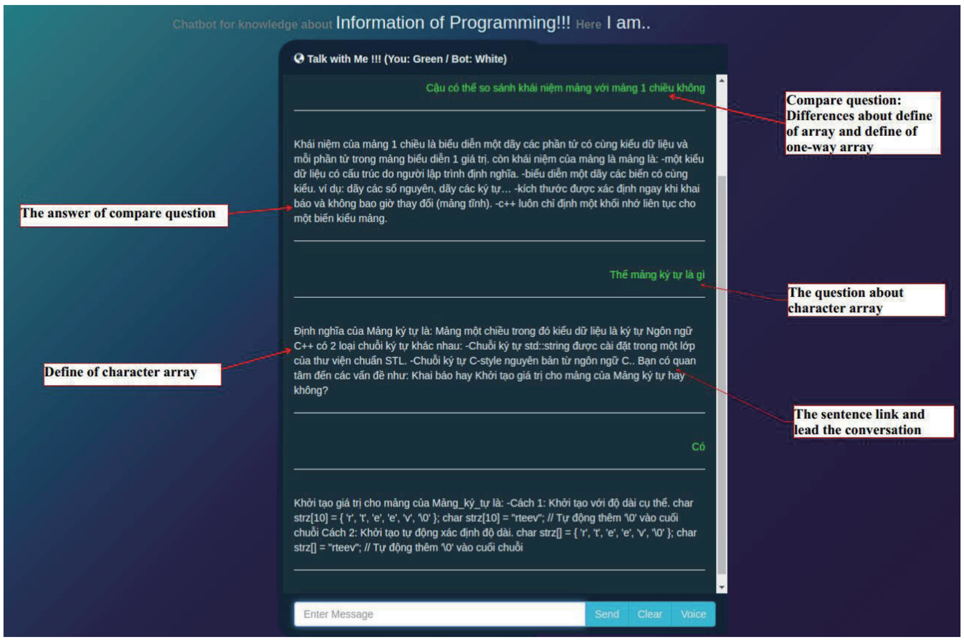
- Introduction to computers
- Algorithms
- Data types and operators of C++

Table 4
Performance of different classifiers.

Classifier	F1-score (%)
Ridge classifier	90.51
SGD	90.33
Passive aggressive classifier	88.67
MLP classifier	85.71
Nearest centroid	88.17

Table 5
Results of testing on six kinds of training queries.

Intent	Quantity	Correct	Accuracy
Greeting	5	5	1.0
Bye	5	4	0.80
Concept	106	88	0.83
Setting	65	53	0.82
Comparison	97	79	0.81
Related knowledge	10	9	0.90
OOS	30	23	0.77
Total	**318**	**261**	**0.82**


Chatbot for knowledge about Information of Programming!!! Here I am..
Talk with Me !!! (You: Green / Bot: White)
Compare question: Differences about define of array and define of one-way array
The answer of compare question
The question about character array
Define of character array
The sentence link and lead the conversation
Enter Message
Send
Clear
Voice

Cậu có thể so sánh khái niệm mảng với mảng 1 chiều không

Khái niệm của mảng 1 chiều là biểu diễn một dãy các phần tử có cùng kiểu dữ liệu và mỗi phần tử trong mảng biểu diễn 1 giá trị. còn khái niệm của mảng là mảng là: -một kiểu dữ liệu có cấu trúc do người lập trình định nghĩa. -biểu diễn một dãy các biến có cùng kiểu. ví dụ: dãy các số nguyên, dãy các ký tự… -kích thước được xác định ngay khi khai báo và không bao giờ thay đổi (mảng tĩnh). -c++ luôn chỉ định một khối nhớ liên tục cho một biến kiểu mảng.

Thế mảng ký tự là gì

Định nghĩa của Mảng ký tự là: Mảng một chiều trong đó kiểu dữ liệu là ký tự Ngôn ngữ C++ có 2 loại chuỗi ký tự khác nhau: -Chuỗi ký tự std::string được cài đặt trong một lớp của thư viện chuẩn STL. -Chuỗi ký tự C-style nguyên bản từ ngôn ngữ C.. Bạn có quan tâm đến các vấn đề như: Khai báo hay Khởi tạo giá trị cho mảng của Mảng ký tự hay không?

Có

Khởi tạo giá trị cho mảng của Mảng_ký_tự là: -Cách 1: Khởi tạo với độ dài cụ thể. char strz[10] = { 'r', 't', 'e', 'e', 'v', '\0' }; char strz[10] = "rteev"; // Tự động thêm '\0' vào cuối chuỗi Cách 2: Khởi tạo tự động xác định độ dài. char strz[] = { 'r', 't', 'e', 'e', 'v', '\0' }; char strz[] = "rteev"; // Tự động thêm '\0' vào cuối chuỗi

Fig. 5. (Color online) Answer of chatbot to question in Example 5.1.

- Control structures: condition and loop statements
- Functions and procedures
- Arrays and operators
- Pointers
- Files

The results of testing the chatbot on this knowledge are shown in Table 6. Here, users inputted their queries to the chatbot, which classified the queries into intents, concepts, comparisons, and settings, then extracted results for users. The process of the chatbot was also checked by lecturers teaching this course. Figure 6 shows the accuracy of the chatbot for each content.

Some contents, such as "Arrays" and "Files", have lower accuracy. Because their meaning is complex, they involve many concepts and it is difficult to express the difference between them,

Table 6
Results of testing on knowledge content of the course.

Content	Number of queries	Number of correct results				Proportion (%)
		Concept	Comparison	Setting	Total	
Introduction to computers	12	6	2	2	10	83
Algorithms	16	6	6	3	15	94
Data types and operators of C++	23	9	7	4	20	87
Control structures: condition and loop statements	44	14	11	12	37	84
Functions and procedures	48	15	18	9	42	88
Arrays and operators	49	16	13	8	37	76
Pointers	48	14	16	10	40	83
Files	28	8	6	5	19	68
Total	**268**	**88**	**79**	**53**	**220**	**82**

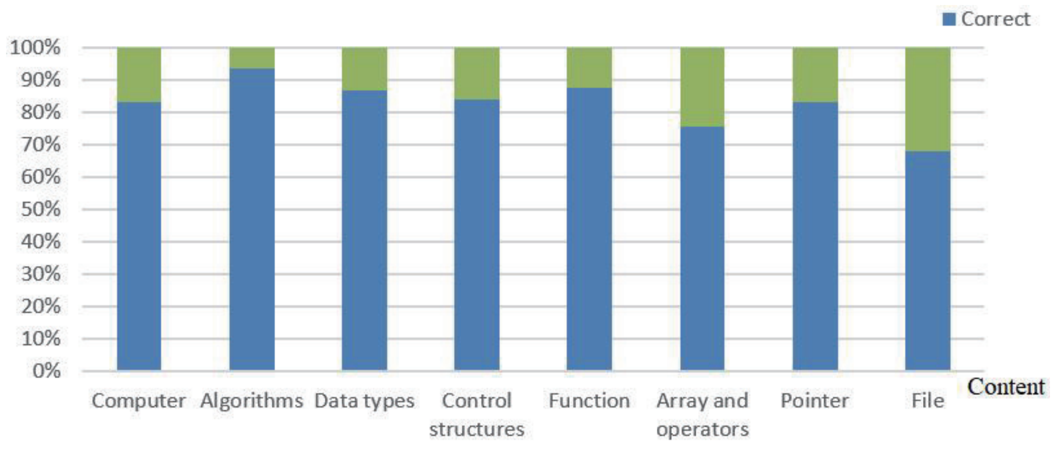


Fig. 6. (Color online) Accuracy of the chatbot for the knowledge content of the course.

such as “two-dimensional array” and “*n*-dimensional array”. Thus, the results retrieved by this chatbot for such contents do not meet the student’s requirements, and the semantic representation of the knowledge must be improved. However, the average accuracy of this chatbot is over 80% and it can extract necessary information for students for other contents. Hence, this system was positively evaluated by students and lecturers in universities.

Wolfram Alpha is an answer engine for searching for knowledge in many domains.(38) It can also extract knowledge about programming. FPT.AI Conversation provides a platform for creating and designing chatbots with a user interface.(24) The user can manage scripts by adding pairs of questions and answers. Table 7 shows a comparison of FPT.AI Conversation, Wolfram Alpha, and our intelligent chatbot regarding answering questions on knowledge about programming.

Table 7
Comparison between the designed chatbot and other systems regarding answering questions on knowledge about programming.

Criteria	FPT.AI Conversation	Wolfram Alpha	Designed chatbot
Sufficiency of knowledge base	Not equipped with a specific knowledge domain.	Knowledge of programming is adequate for searching for basic knowledge.	
	• Only supports creation of simple scripts by adding questions and answers.	• Knowledge base is collected from many sources. Knowledge is not organized as a course.	• Knowledge base is organized similarly to the content of the syllabus of this course.
Ability to extract required knowledge	• Can only supply information for the inputted query.	• Can retrieve knowledge based on the meaning of the inputted query. • Gives more knowledge related to the main search results.	
	• Only processes a single query. • Cannot automatically extract other information related to the current query.	• Only processes a single query. • Related knowledge is extracted from relations between keywords in the query and the stored knowledge.	• Interacts with users through communication script, so can help users understand more than the searched content. • Related knowledge is extracted from relations between search results and the meaning of the query.
Support of studying	Targeted at general users rather than specific to education.		Built to assist studying of students.
	• Searching is based on the extraction of intents and entities from the FPT.AI system. • Cannot extract related information. • Can communicate in many languages, such as Vietnamese, English, Japanese, Indonesian, and Malay.	• Search results only answer query. Results do not belong to any lesson or unit of a course. • Related results are extracted automatically. Some results are not relevant to users. • Can only communicate in English.	• Knowledge base is organized as the syllabus of the corresponding course. Hence, the search results support understanding of more than the extracted knowledge. • The chatbot acts as a tutor and explains the meaning of the content. Hence, the extracted knowledge is relevant to users. • Can only communicate in Vietnamese.

6. Conclusions

An intelligent chatbot requires a knowledge base to support users better. The chatbot proposed in this paper is a question-answering system that can help users complete specific tasks from one-to-one conversations, and a knowledge model integrating the Rela-model and the structure of scripts, called the Rela-Scripts model, is presented. The Rela-model is an approach based on ontology technology for the knowledge of relations, which includes concepts, relations between them, and inference rules of the knowledge domain. The structure of a script represents a common conversation based on query knowledge. The Rela-Scripts model is useful for organizing a knowledge base as materials to build an intelligent chatbot. On the basis of these materials, some problems in querying knowledge using an intelligent chatbot are proposed and solved: classifying an inputted query, retrieving knowledge content based on matched keywords, and searching for related knowledge based on the results of a query.

Knowledge on programming is important in many university courses. An intelligent chatbot to answer questions on knowledge on programming is designed using the proposed architecture. The chatbot can search for knowledge required by students in the Introduction to Programming course. It communicates with students in Vietnamese and gives results that meet the requirements of users. This chatbot is useful for helping students better understand the lessons in the Introduction to Programming course and consolidating their knowledge. Its instruction helps students enhance their programming skills and augments the contents of this course.

In the future, some methods for integrating multiple knowledge domains will be studied.(31,39) The knowledge base of the chatbot will be widened to incorporate many knowledge domains. Such integration also needs to be organized on the basis of conversation with the chatbot to increase its ability to retrieve knowledge from many sources. Moreover, natural language processing for Vietnamese will be studied in depth by analyzing the grammatical structures of queries(40,41) and combining natural language processing with deep learning techniques(42) to represent the meaning of queries more exactly.

Moreover, to support e-learning, in addition to the ability to search for knowledge, the chatbot must incorporate tutoring methods to solve problems in courses.(9,43) Some tutoring techniques will be improved to solve more difficult problems with complex requirements.

Acknowledgments

This research was funded by Vietnam National University Ho Chi Minh City (VNU-HCM) under grant number DSC2021-26-07.

References

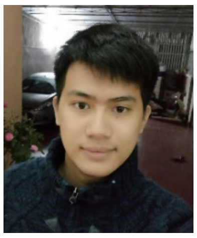
1 A. Shevat: Designing Bots: Creating Conversational Experiences (O'Reilly, 2017).
2 B. Liu and I. Lane: Proc. 2018 Conf. North American Chapter of the Association for Computational Linguistics (NAC-ACL): Student Research Workshop (ACM, Louisiana, 2018) 67–73. https://www.aclweb.org/anthology/N18-4010
3 A. Davani, A. Shirehjini, and S. Daraei: J. Ambient. Intell. Human Comput. **9** (2018) 187. https://doi.org/10.1007/s12652-016-0433-9
4 Cortana: https://www.microsoft.com/en-us/cortana/ (accessed May 2021).
5 Google Assistant: https://assistant.google.com/ (accessed May 2021).
6 Siri: https://www.apple.com/ios/siri/ (accessed May 2021).
7 Alexa: https://developer.amazon.com/alexa/ (accessed May 2021).
8 Chatbot 1022: https://1022.vn/ (accessed May 2021).
9 M. Carisi, A. Albarelli, and F. Luccio: Proc. 5th EAI Int. Conf. Smart Objects and Technologies for Social Good (ACM, Spain, 2019) 49–54. https://doi.org/10.1145/3342428.3342664
10 Mathway: https://www.mathway.com/Graph (accessed May 2021).
11 H. Nguyen, V. Pham, D. Tran, and T. Le: Proc. 11th IEEE Int. Conf. Knowledge and Systems Engineering (IEEE, Vietnam, 2019) 145–150. https://ieeexplore.ieee.org/document/8919396
12 H. D. Nguyen, D. A. Tran, H. P. Do, and V. T. Pham: Int. J. Integr. Eng. **12** (2020) 211. https://doi.org/10.30880/ijie.2020.12.07.024
13 L. Zhou, J. Gao, D. Li, and H. Shum: Comput. Linguist. **46** (2020) 53. https://doi.org/10.1162/coli_a_00368
14 Cleverbot: https://www.cleverbot.com/ (accessed May 2021).
15 A. Aizawa: Inf. Process. Manag. **39** (2003) 45. https://doi.org/10.1016/S0306-4573(02)00021-3
16 N. Do, H. Nguyen, and A. Selamat: Int. J. Software Eng. Knowl. Eng. **28** (2018) 1047. https://doi.org/10.1142/S0218194018500304

17 X. Pham, T. Tran, V. Nguyen-Le, V. Pham, and H. Nguyen: Proc. 12th IEEE Int. Conf. Knowledge and Systems Engineering (IEEE, Vietnam, 2020) 207–212. https://ieeexplore.ieee.org/document/9287775
18 H. Shum, Xd. He, and D. Li: Front. Inform. Tech. El. **19** (2018) 10. https://doi.org/10.1631/FITEE.1700826
19 L. K. Fryer, D. Coniam, R. Carpenter, and D. Lapusneanu: Lang. Learn. Technol. **24** (2020) 8. http://hdl.handle.net/10125/44719
20 A. M. Elkahy, Y. Song, and X. He: Proc. 24th Int. Conf. World Wide Web (ACM, Florence, 2015) 278–288. https://doi.org/10.1145/2736277.2741667
21 X. He and L. Deng: IEEE Signal Process Mag. **34** (2017) 109. https://ieeexplore.ieee.org/document/8103169
22 F. Alam, M. Danieli, and G. Riccardi: Comput. Speech Lang. **50** (2018) 40. https://doi.org/10.1016/j.csl.2017.12.003
23 C. Truong and T. Tran: Proc. 7th Int. Conf. System Science and Engineering (IEEE, Quang Binh, 2019) 324–328. https://ieeexplore.ieee.org/abstract/document/8823516
24 FPT.AI: https://fpt.ai/chatbot (accessed May 2021).
25 N. Do, H. Nguyen, and L. Hoang: Proc. 12th Int. Conf. Knowledge Engineering and Ontology Development (Scitepress, Budapest, 2020) 313–320. https://www.scitepress.org/Link.aspx?doi=10.5220/0010174403130320
26 N. Noy and D. McGuinness: Final Report on the 2013 NSF Workshop on Research Challenges and Opportunities in Knowledge Representation. National Science Foundation Workshop Report (Alexandria, USA, 2013). http://krnsfworkshop.cs.illinois.edu/home
27 C. Li, A. Li, Y. Wang, H. Tu, and Y. Song: Proc. IEEE 5th Int. Conf. on Data Science in Cyberspace (IEEE, Hong Kong, 2020) 312–319. https://ieeexplore.ieee.org/document/9172857
28 H. Nguyen, N. Do, N. Tran, X. Pham, and V. Pham: Appl. Comput. Intell. Soft Comput. **2020** (2020) 9834218. https://doi.org/10.1155/2020/9834218
29 K. Munir and M. S. Anjum: Appl. Comput. Inf. **14** (2018) 116. https://doi.org/10.1016/j.aci.2017.07.003
30 M. Dumontier, C. J. Baker, J. Baran, A. Callahan, L. Chepelev, J. Toledo, N. Rio, G. Duck, L. Furlong, N. Keath, D. Klassen, J. McCusker, N. Queralt-Rosinach, M. Samwald, N. Villanueva-Rosales, M. Wilkinson, and R. Hoehndorf: J. Biomed. Semantics **5** (2014) 14. https://doi.org/10.1186/2041-1480-5-14
31 N. Do, H. Nguyen, and T. Mai: Appl. Sci. **9** (2019) 3793. https://doi.org/10.3390/app9183793
32 S. Lin, J. Li, and C. Yu: Sens. Mater. **31** (2019) 1789. https://doi.org/10.18494/SAM.2019.2333
33 S. Larson: Proc. 2019 Conf. Empirical Methods in Natural Language Processing (ACM, Hong Kong, 2019) 1311–1316. https://www.aclweb.org/anthology/D19-1131
34 K. Shridhar, A. Dash, A. Sahu, G. Pihlgren, P. Alonso, V. Pondenkandath, G. Kovacks, F. Simistira, and M. Liwicki: Proc. 2019 Int. Joint Conf. Neural Networks (IEEE, Hungary, 2019). https://ieeexplore.ieee.org/document/8852420
35 T. Le, S. Luu, H. Nguyen, and N. Do: Proc. 25th Asia-Pacific Conf. Communications (IEEE, Vietnam, 2019) 310–315. https://ieeexplore.ieee.org/document/9026411
36 T. Tran, P. Nguyen, T. Dinh, and T. Tran: Introduction to Programming (Publishes of Science and Technology, 2015) (in Vietnamese).
37 J. Hubbard: Theory and Problems of Fundamentals of Computing with C++ (McGraw-Hill, 1998).
38 Wolfram Alpha: https://www.wolframalpha.com/ (accessed May 2021).
39 H. D. Nguyen, N. V. Do, V. T. Pham, A. Selamat, and E. Herrera-Viedma: IEEE Access **8** (2020) 76991. https://ieeexplore.ieee.org/abstract/document/9072439/
40 B. Nguyen, H. Nguyen, and L. Romary: Lang. Resour. Eval. **40** (2006) 291. https://www.academia.edu/9619228/Lexical_descriptions_for_Vietnamese_language_processing
41 H. Nguyen, T. Huynh, S. Hoang, V. Pham, and I. Zelinka: Proc. 15th Int. Conf. Evaluation of Novel Approaches to Software Engineering (Scitepress, Prague, 2020) 339–346. https://www.scitepress.org/Link.aspx?doi=10.5220/0009358803390346
42 A. Huynh, B. Nguyen, H. T. Nguyen, H. Nguyen, S. Vu, and H. Nguyen: Proc. 16th Int. Conf. Evaluation of Novel Approaches to Software Engineering (Scitepress, Prague, 2021) 372–379. https://www.scitepress.org/Link.aspx?doi=10.5220/0010478903720379
43 M. Phan, H. Nguyen, D. Tran, T. Le, and N. Tran: Eng. Lett. **28** (2020) 1108. http://www.engineeringletters.com/issues_v28/issue_4/EL_28_4_17.pdf

About the Authors

Hien D. Nguyen received his B.S. and M.S. degrees from the University of Sciences, VNU-HCM, Vietnam, in 2008 and 2011, respectively. He received his Ph.D degree from the University of Information Technology, VNU-HCM, in 2020. He is a senior lecturer at the Faculty of Computer Science, University of Information Technology, VNU-HCM. His research interests include knowledge representation, automated reasoning, and knowledge engineering, especially intelligent systems in education, such as intelligent problem solvers. He received the Best Presentation Clip Award at AI-Socha (Ho Chi Minh City) in 2020, and Consolation Prizes of the Vietnam Fund for Scientific and Technological Creations (VIFOTEC) Award and the Technological Creation Awards of Binh Duong province in 2016 and 2015, respectively. (hiennd@uit.edu.vn)

Tuan-Vi Tran is an undergraduate student at the Faculty of Computer Science, University of Information Technology (UIT), VNU-HCM, Vietnam. His research interests are knowledge presentation and natural language processing, especially applying NLP to intelligent systems in education, such as constituency parsing and classification problems in Vietnamese. He got the Best Student Paper Award at KSE 2020. (18520245@gm.uit.edu.vn)



Xuan-Thien Pham is a student at the Faculty of Computer Science, University of Information Technology (UIT), VNU-HCM, Vietnam. His research interests are knowledge-based systems and automated reasoning, especially building intelligent systems in education. He got the Best Student Paper Award at KSE 2020. (18520158@gm.edu.vn)

Anh T. Huynh received his B.S. and M.S. degrees from the University of Information Technology (UIT), VNU-HCM, Vietnam, in 2011 and 2014, respectively. From 2011 to 2014, he was an assistant lecturer at the Faculty of Software Engineering, UIT. Since 2014, he has been a lecturer at the Faculty of Software Engineering, UIT. His research interests are in ontology, knowledge bases, NLP, and machine learning. (anhht@uit.edu.vn)

Nhon V. Do received his M.Sc. and Ph.D. degrees in computer science from the University of Sciences, VNU-HCM, Vietnam in 1995 and 2002, respectively. He was an associate professor at the University of Information Technology, VNU-HCM, Vietnam, from 2006 to 2018 and at Hoa Sen University, Vietnam, from 2018 to 2019. He was also an associate professor and the head of the Artificial Intelligence Department at Ho Chi Minh City Open University, Vietnam, from 2019 to March 2020. He has been an associate professor and the head of the Information Technology Department at Hong Bang International University, Vietnam, from March 2020. His research interests include artificial intelligence, computer science, and their practical applications, especially intelligent and knowledge-based systems. (nhondv@hiu.vn)