Semantic Parsing for Vietnamese Question Answering System

Vu Xuan Tung, Nguyen Le Minh
Japan Advanced Institute of Science and Technology
Email: {tungvx, nguyenml}@jaist.ac.jp

Duc Tam Hoang
University of Engineering and Technology
Email: tamhd1990@gmail.com

***Abstract*—Based on a framework for English, we developed a Vietnamese Question Answering System. The learning paradigm in the framework reduces the burden of providing supervision during semantic parsing. Whilst taking the advantages from this mechanism, we further create our own feature calculation which is suitable for Vietnamese. A method of dynamic learning for feature computation is also presented in this work.**

I. Introduction

Semantic Parsing is the task of converting a natural language sentence into a special meaning representation which can be processed by computers. An effective way to do this is using machine learning to build a model of relationship between natural language structure and formal representation structure. Unfortunately, most of the current approaches require very large amount of fully annotated data, i.e. pairs of natural sentence and the corresponding representation, in order to obtain a good model. State-of-the-art methodologies often use statistical analysis to build the model and thus generally ill-perform with unseen data. Therefore, in order to get good results, they need to collect a large amount of annotated corpus. The annotation, however, is almost prepared manually and thus is difficult and time consuming.

Recently, James et al. [1] implemented a new learning paradigm aimed at alleviating the supervision burden. The algorithm is able to predict complex structures which only rely on a binary feedback. Borrowing the idea from these authors, we developed a Question Answering System for Vietnamese with suitable feature computations.

II. Related Work

There has been a number of works in Vietnamese Question Answering. Tran et al. [6] propose a model using semantic relations which are extracted by from Vietnamese texts. Although the experimental result of the system is positive, it depends on the number of statistically extracted relations. Nguyen et al. [3] develop a Question Answering System for Vietnamese . Because the system enables users to query an ontological knowledge base using pattern matching, the performance is affected by the size of the ontology. Nguyen and Le [2] utilize restricted semantic grammars to transform a natural question into an SQL query. The authors in [5] proposed a question answering system for Vietnamese named entities, which restricts to only questions of the form "Who?", "Whom?" and "Whose?".

The rest of this paper is organized as follow. Section III describes meaning representation in our system. After illustrating the way we maintain the parsing model in Section IV, we present our experiment result and discuss it in Section V. Section VI analyzes errors of our system. Finally, conclusion and future works are drawn in Section VII.

III. Meaning Representation

Similar to "Target Meaning Representation" of [1], the representation of meaning in our system is the composition of pre-defined functions in which each function allows only one argument. For example, $longest(river(stateId("colorado")))$ is the formal representation of "Con sông dài nhất chảy qua Colorado là gì?" ("What is the longest river runs through Colorado?"). There are two steps of building such a representation from a natural language sentence: finding the mapping from each token to a function, and composing the mapped functions into the final logical form.

A. Predefined Functions

As mentioned, our meaning representation uses a set D of pre-defined functions. Each function receives one typed argument and returns a typed value. For instance, $length$ function receives an argument of type $River$ and returns an $Integer$ value. In our system, the list of functions and the corresponding types are configured in a text file. This way of configuration makes the system flexible because if we want to change the data, changing the functions configuration file is enough.

B. Token - Function mapping

In Natural Language Processing, there often is a pre-process phase named tokenization which generates a list of tokens (words) from the input sentence. Token is an element that has some meanings. For example, "New Mexico" which is a name of a city should be one token instead of being divided into "New" and "Mexico" where each does not make sense in the given context. An other instance is "tiểu bang" in Vietnamese which means "state". If it is separated into "tiểu" and "bang" as tokens, the original meaning is not preserved. Currently, there are some works which have dealt with tokenization for Vietnamese with high accuracies such as vnTokenizer [4].

In our system, we assume that tokenization is provided by some other tools. Our accepted inputs are tokenized sentences,

e.g. "Con sông, dài nhất, chảy, qua, Colorado, là, gì, ?" ("What, is, the, longest, river, runs, through, Colarado, ?"). In our framework, each token is supposed to be aligned with some (may be 0) predefined functions. In the above example, "Con sông", "dài nhất" and "Colorado" are mapped to $river(\_)$, $longest(\_)$ and $stateId(\_)$ functions respectively.

C. Function - Function composition

When alignments between tokens and functions are determined, the composition of the chosen functions needs to be made. A function $f_1(\_)$ could be composed with a function $f_2(\_)$ to form $f_1(f_2(\_))$ if the returned type of $f_2(\_)$ agrees with the argument type of $f_1(\_)$. In the mentioned example, $longest(\_)$ is composed with $river(\_)$ and $river(\_)$ is composed with $stateId(\_)$ to form $longest(river(stateId(\text{"}colorado\text{"})))$.

IV. Semantic Parsing Model

Each semantic parser maintains a model containing data to be used for transforming a natural language sentence into a formal representation. Similar to [1], the model in our system consists of two parts: Token-Function feature calculator and Function-Function feature calculator. Before going into the details of these parts, the framework in [1] is summarized first.

A. Semantic Parsing

Semantic Parsing can be formalized as a function:

$$F : \mathbf{S} \mapsto \mathbf{M}$$

where $\mathbf{S}$ is the set of all natural sentences; and $\mathbf{M}$ is the set of all meaning representations, e.g. function compositions. For each sentence s in $\mathbf{S}$, there are several candidates of the corresponding meaning representation, and Semantic Parsers often choose one of those by maximizing some goal whose higher value makes the meaning representation more likely to be correct (in terms of human understanding of the input sentence). In [1], the function F is defined as following:

$$F(s) = w^T \arg\max_{\alpha,\beta}(\sum_{t\in s}\sum_{f\in D}\alpha_{tf}\phi_1(s,t,f) + \sum_{t_1,t_2\in s}\sum_{f_1,f_2\in D}\beta_{t_1f_1,t_2f_2}\phi_2(s,t_1,f_1,t_2,f_2)) \quad (1)$$

where each α_{tf} is a binary variable indicating whether a token t is mapped to a function f and $\phi_1(s,t,f)$ calculates how likely such mapping is in the context of the sentence s. Similarly, each binary variable $\beta_{t_1f_1,t_2f_2)}$ indicates that the mapped function f_1 of t_1 is composed with the mapped function f_2 of t_2 and $\phi_2(s,t_1,f_1,t_2,f_2)$ represents how likely such mapping is. Function $\phi_1(s,t,f)$ (and similarly $\phi_2(s,t_1,f_1,t_2,f_2)$) returns a vector of values, which is called the feature vector; and w is a weight vector. There is a number of constraints over variables α_{tf} and $\beta_{t_1f_1,t_2f_2}$ as following.

- $\beta_{t_1f_1,t_2f_2} = 1$ if and only if $\alpha_{t_1f_1} = 1$ and $\alpha_{t_2f_2} = 1$.
- A token can be mapped to only one function.
- If $\beta_{t_1f_1,t_2f_2} = 1$, then argument type of f_1 must be compatible with returned type of f_2.
- Functions composition is directional and acyclic.

Integer Linear Programming is employed to solve the right hand side formula of Equation 1 with respect to the above constraints. The values of variables α_{tf} and $\beta_{t_1f_1,t_2f_2}$ are then utilized to construct the function composition which is regarded as the meaning representation. Out main contributions are two functions $\phi_1(s,t,f)$ and $\phi_2(s,t_1,f_1,t_2,f_2)$ for Vietnamese, which are described next.

B. Token-Function feature vector - $\phi_1(s,t,f)$

This feature vector is computed from a sentence s, a token t, and a function f. Each function is configured to have a set of surface forms (words), which are typically associated with the function. The feature vector is calculated by comparing the token with these surface forms. Thus, the larger the number of surface forms is, the more accurate the feature vector is. Similar to [1], we restrict the number of surface forms per function. In our current experiment, there is in average 1.39 words per functions, in comparison with 1.42 of [1].

In our system, the feature vector is generated based on lexical matching between token and surface forms. In many cases in Vietnamese, two words with lexical similarity may have the same meaning. For example "Tiểu bang" and "bang" both have meaning as "state". Similary, we have "con sông" and "sông" (river), "ngọn núi" and "núi", etc. Further features are proposed in Section VII as our future works.

C. Function - Function feature vector - $\phi_2(s,t_1,f_1,t_2,f_2)$

$\phi_2(s,t_1,f_1,t_2,f_2)$ measures how likely $f_1(f_2(\_))$ will be formed in the context of s, t_1 and t_2. One feature value is reserved for the requirement that returned type of f_2 must agree with the argument type of f_1. The second value is the distance between the tokens t_1 and t_2. If the distance is small, they will have more chances to be composed. This is due to the usual speaking style of Vietnamese. Let us consider the following example: "Con sông nào chảy qua tiểu bang missisipi?" ("What river runs through state mississippi?"). There are at least two candidates for the result of semantic parser which are illustrated in Figure 1 and Figure 2. Our system permits a higher feature value for the first case. The intuition here is that because "con sông" is near to "tiểu bang" in comparison to "Mississipi", the function $river$ is preferred to combine with $state$ to form $river(state(\_))$.

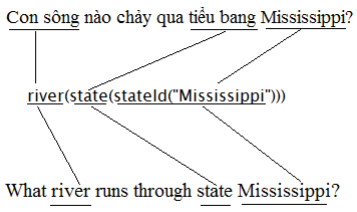

Fig. 1. One result of parsing "con sông nào chảy qua tiểu bang missisipi?"

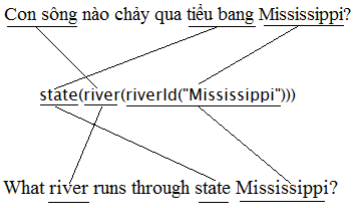

Fig. 2. Another result of parsing "con sông nào chảy qua tiểu bang missisipi?"

James et al. [1] consider another feature indicating which logical symbols are usually composed together, e.g. $state$ function tends to receive $stateId$ function as its argument. However, the authors do not clearly state the way of computing this kind of feature. If this task is done by manually configuring such feature when defining functions, the result would be thoroughly good. Nevertheless, that idea is indeed not positive with respective to scalability since large number of functions will overwhelm annotators.

In our system, this feature value is learned automatically during the running time of the system. We refer this as dynamic compositional preferences learning. Suppose f_1 has a variable p denoting how likely it will be composed with f_2. Executing the final functions composition gives us an answer to the input question. By receiving the feedback from the outside world, we may know the correctness of that answer.

If the result is correct: we will increase the value of p by Formula 2. It is clear that p will be increased but it is always smaller than M which the maximum value of a feature.

$$p = \frac{(p+1)M}{M+1} \tag{2}$$

If the result is wrong: p needs to be decreased which is achieved by Formula 3. The formula preserves the fact that $p \geq 0$.

$$p = \frac{pM}{M+1} \tag{3}$$

V. Experiment

For evaluating our system, we use the Geoquery domain which consists of a database and Prolog query language for U.S. geographical facts. The corpus has 880 pairs of one English tokenized natural language query and one Prolog query. The Prolog queries are executed against the database to get the results which are the input of our training process. In other words, Prolog queries (annotations) do not play any further role in the learning phase. English queries are translated into Vietnamese ones by Google Translate API. Manual correction is needed to obtain meaningful Vietnamese questions because of some word-to-word translations.

Following the experiment in [1], we randomly select 250 queries for training, 250 other queries for testing from the above 880 queries. Our experiment is used to answer the following doubt:

1) What are the effects of dynamic compositional preferences learning?
2) How do two learning approaches of [1] perform in our system?

A. What are the effects of dynamic compositional preferences learning?

In this experiment, we run our system with and without dynamic learning of compositional preferences. The experiment result is described in Table II. The result is computed as $c * 100/250$ where c is the number of correct answers.

"No dynamic learning" row means that we do not use dynamic learning of compositional preferences. It is clear from the table that the strategy improves the accuracy of the system by 4.4% and 3.2% for training data and testing data respectively. We also may wonder that if a set of questions are asked many times by users, how does this strategy perform? The last row of Table II shows that if we repeatedly learn on the training data 10 times, the results on the training and testing data are 62% and 53.2% resp.

TABLE II
ACCURACY OF MODEL ON TRAING DATA AND TESTING DATA.

Algorithm	Training set	Testing set
No dynamic learning	53.6%	51.6%
Dynamic learning	58%	54.8%
Dynamic learning with 10 repetitions on training data	62%	53.2%

B. How do two learning approaches of [1] perform in our system?

We further implemented two learning mechanisms mentioned in [1], namely "Direct Approach" and "Aggressive Approach". The former did not successfully learn the weight vector as it contains negative values. This is because the way we choose the feature value of "Token-Function mapping" is simple, just being word matching. We need some features which measure the semantic similarity between words. James et al. [1] use Wordnet which is not available for Vietnamese. We have an idea of using word2vec, a tool of Google, to tackle this problem. This is left for our future works.

On the other hand, the second mechanism can learn the weight vector but the result is not improved, around 51%. In future, we need to investigate more about this to take advantage from machine learning.

VI. Errors Analysis

Studying on the failed cases of the system, we observe that complex noun phrases often lead to failure. Let us take the following sentence as an example: "Dân số của thủ phủ của tiểu bang lớn nhất mà Mississippi chạy qua là gì?" ("What is the population of the capital of the largest state through which the Mississippi runs?"). While the correct parsing should be as in Figure 3, the answer of our system is as in Figure 4.

The noun phrase "Dân số của thủ phủ của tiểu bang lớn nhất mà Mississippi chạy qua" ("the population of the capital of the largest state through which the Mississippi runs") consists of head word "Dân số" ("the population") and the adnominal word

TABLE I
PARSING RESULT NOUN PHRASES

Noun Phrase	Head Word	Parsing Result
Dân số của thủ phủ của tiểu bang lớn nhất mà Mississippi chạy qua (the population of the capital of the largest state through which the Mississippi runs)	Dân số (the population)	$population(arg)$
thủ phủ của tiểu bang lớn nhất mà Mississippi chạy qua (the capital of the largest state through which the Mississippi runs)	thủ phủ (the capital)	$capital(arg)$
tiểu bang lớn nhất mà Mississippi chạy qua (the largest state through which the Mississippi runs)	tiểu bang lớn nhất (the largest state)	$largest(state(run(riverId(”Mississipi”))))$

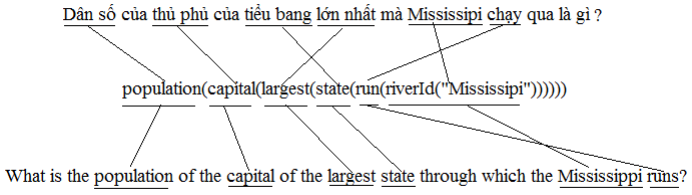

Fig. 3. Correct parsing of "Dân số của thủ phủ của tiểu bang lớn nhất mà Mississippi chạy qua là gì?"

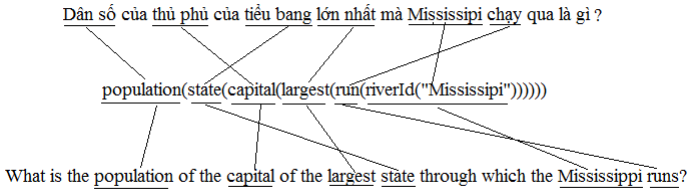

Fig. 4. Result of our system on "Dân số của thủ phủ của tiểu bang lớn nhất mà Mississippi chạy qua là gì?"

"thủ phủ của tiểu bang lớn nhất mà Mississippi chạy qua". As a consequence, the parsing result of this noun phrase is expected to be $population(arg)$ where arg is the parsing result of the adnominal word. Similarly applying this reason for the other noun phrases we have the result of parsing for each as presented in Table I. As a result, the final function composition becomes $population(capital(largest(state(run(riverId(Mississipi))))))$. Our current system does not support this kind of priority where the function mapped to the head word in a noun phrase is given the priority to combine with the function mapped to adnominal word. This can be done by using a syntactic parser and adding a feature value to "Function-Function feature vector" for that priority.

VII. CONCLUSION

We developed a Vietnamese Question Answering System based on a framework for English. A performance of 54.8% accuracy is considered highly promising for Vietnamese Semantic Parsing. However, the current ways of calculating feature values remain simple. We propose a number of possibilities as follows.

1) We plan to use word2vec to build a model of semantic similarity between Vietnamese words.
2) For function-function mapping, we can first apply syntax parsing for the sentence. If two tokens are both in a syntactical element, e.g. subject, their two mapped functions are more likely to be combined with each other.
3) We plan to give priorities for the function mapped to the head word of a noun phrase to combine with the function mapped to the adnominal word.

Early analysis shows that machine learning methods in [1] do not perform well in our system; which need further investigation.

Currently, one text file is employed to configure the logical functions and database type, which makes the system flexible in terms of database. It is possible to change the database from geometric to others. We can also switch the type of database from Prolog to others, e.g. SQL.

REFERENCES

[1] Clarke James, Goldwasser Dan, Chang Ming-Wei, and Roth Dan. Driving semantic parsing from the world's response. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, CoNLL '10, pages 18–27, 2010.

[2] AnhKim Nguyen and HuongThanh Le. Natural language interface construction using semantic grammars. In *PRICAI 2008: Trends in Artificial Intelligence*, volume 5351 of *Lecture Notes in Computer Science*, pages 728–739. Springer Berlin Heidelberg.

[3] Dai Quoc Nguyen, Dat Quoc Nguyen, and Son Bao Pham. A vietnamese question answering system. In *Proceedings of the 2009 International Conference on Knowledge and Systems Engineering*, KSE '09, pages 26–32, 2009.

[4] Le Hong Phuong, Nguyen Thi Minh Huyen, Azim Roussanaly, and Ho Tuong Vinh. A hybrid approach to word segmentation of vietnamese texts. In *Language and Automata Theory and Applications*, pages 240–249. 2008.

[5] Mai-Vu Tran, Duc-Trong Le, Xuan-Tu Tran, and Tien-Tung Nguyen. A model of vietnamese person named entity question answering system. *26th Pacific Asia Conference on Language, Information and Computation (PACLIC 26)*, page 325, 2012.

[6] Vu Mai Tran, Vinh Duc Nguyen, Oanh Thi Tran, Uyen Thu Thi Pham, and Thuy Quang Ha. An experimental study of vietnamese question answering system. In *International Conference on Asian Language Processing, 2009. IALP '09.*, pages 152–155.