A Spoken Language Understanding Approach Using Successive Learners

Wei-Lin Wu, Ru-Zhan Lu, Hui Liu, Feng Gao

Department of Computer Science and Engineering
Shanghai Jiao Tong University, Shanghai, China
{wu-wl,lu-rz,liuhui,gaofeng}@cs.sjtu.edu.cn

Abstract

In this paper, we describe a novel spoken language understanding approach using two successive learners. The first learner is used to identify the topic of an input utterance. With the restriction of the recognized target topic, the second learner is trained to extract the corresponding slot-value pairs. The advantage of the proposed approach is that it is mainly data-driven and requires only minimally annotated corpus for training whilst retaining the understanding robustness and deepness for spoken language. Experiments have been conducted in the context of Chinese public transportation information inquiry domain. The good performance demonstrates the viability of the proposed approach.

Index Terms: spoken language understanding, classification, spoken dialogue system.

1. Introduction

Spoken Language Understanding (SLU) is one of the key components in spoken dialogue systems. Its task is to identify the user's goal and extract from the input utterance the information needed to complete the query.

There are mainly two mainstreams in the SLU researches: knowledge-based approaches, which are based on robust parsing or template matching techniques [1, 2, 3]; and statistical approaches, which are based on stochastic models [4, 5]. Both approaches have their drawbacks. The former is cost-expensive since its grammar development is time-consuming, laboursome and requires linguistic skills. It is also strictly domain-dependent and hence difficult to be adapted to new domains. On the other hand, although addressing such drawbacks, the latter often suffers the data sparseness problem and needs a large amount of fully annotated corpus in order to reliably estimate an accurate model. More recently, some new variation methods are proposed through certain trade-offs, such as the semi-automatically grammar learning approach [6] and Hidden Vector State (HVS) model [7].

This paper proposes a novel SLU approach. The components in our approach mainly include two successive classifiers: topic classifier and semantic classifier. The proposed approach is mainly data-driven and requires only minimally annotated corpus for training whilst retaining the understanding robustness and deepness for spoken language. The evaluation in the context of Chinese public transportation information inquiry domain indicates the viability of the proposed approach. The remainder of this paper is organized as follows.

The next section introduces the system architecture and describes in details its components. Section 3 presents the experimental setup and results. Finally, Section 4 concludes the paper and gives the future works.

2. The system architecture

Figure 1 illustrates the overall system architecture. It also describes the whole spoken language understanding procedure of an example sentence (Because the length is limited, in this paper we only illustrate all the examples in English, which are Chinese sentences, in fact.).

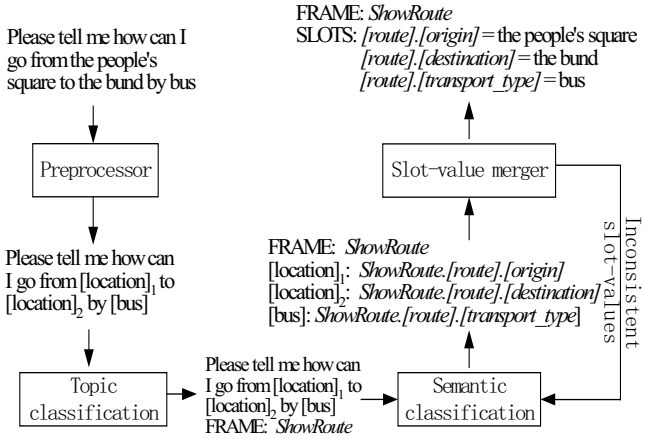

Figure 1 *The system architecture.*

2.1. The Preprocessor

In the development of dialog systems, a set of semantic classes need to be pre-defined, which is associated with semantic concepts such as location names and times. Usually, the preprocessor is to look for the substrings in the sentence corresponding to a semantic class or matching a regular expression and replace them with the class label, e.g., "Huashan Road" and "1954" are replaced with two class labels [road_name] and [number] respectively. In our system, the preprocessor can recognize more complex word sequences. For example, "1954 Huashan Road" can be recognized as [address] through matching a rule like "[address] → [number] [road_name]". The preprocessor is implemented with a local chart parser, which is a variation of the robust parser introduced in [8]. The robust local parser can skip noise words in the sentence, which ensures that the system has the low level robustness. For example, "1954 of the Huashan Road)" can also be recognized as [address] by skipping the words "of the". Unfortunately, the robust local parser possibly skips the words in sentence by mistake and produces an incorrect class label. To avoid this side-effect, this local parser exploits an embedded

decision tree for pruning, of which the details can be seen in [9]. It should be noted that, according to our experience, the job of authoring the grammar used by the local chart parser and annotating the training cases for the embedded decision tree is easy for a general developer with good understanding of the application and can be finished in several hours.

2.2. Topic classification

The semantic representation of an application domain is usually defined in terms of semantic frames. A semantic frame contains a frame type, which represents the topic of the input sentence; and some slots, which represents the constraints the query goal has to satisfy. Given the representation of semantic frame, topic classification can be regarded as identifying the frame type. A straightforward application of topic classification among many others is call-routing [10]. It is suited to be dealt using pattern recognition techniques. The application of statistical pattern techniques to topic classification can improve the robustness of the whole understanding system. Also, in our system, topic classification can greatly reduce the search space and hence improve the performance of subsequent semantic classification. For example, the total number of slots into which the class [location] can be filled in all topics is 33 and the corresponding maximum number in a single topic decreases to 10.

Many statistical pattern recognition techniques have been applied to topic classification, such as Vector Space Model, Naïve Bayes, N-Gram and Support Vector Machines (SVMs) [10, 11]. According to the literature [11] and our experiments, the SVMs showed better performance than many other statistical classifiers. We resorted to the LIBSVM toolkit [12] to construct the SVMs for our experiments. Following the practice in [11], the SVMs use the binary valued features vector. If the simplest feature (Chinese character) is used, each query is converted into a feature vector $\overrightarrow{ch} = < ch_1, \ldots, ch_{|\overrightarrow{ch}|} >$ ($|\overrightarrow{ch}|$ is the total number of Chinese characters occur in the corpus) with binary valued elements: 1 if a given Chinese character is in this input sentence or 0 otherwise. Due to the existence of the preprocessor, we can also include as semantic class labels (e.g., [location]) as features for topic classification. Intuitively, the class label features are more informative than the Chinese character features. At the same time, including class labels as features can also relieve the data sparseness problem.

2.3. Topic-dependent semantic classification

The job of semantic classification is to assign the concepts with the most likely slot. It can also be modeled as a classification problem since the number of possible slot names for each concept is limited. Let's consider the example sentence in Figure 1. After the preprocessing and topic classification, we get the preprocessed result "*Please tell me how can I go from [location]$_1$ to [location]$_2$ by [bus]?*" and the topic **ShowRoute**. We have to work out which slots are to be filled with the values such as [location]$_2$. The first clue is the surrounding literal context. Intuitively, we can infer that it is a [destination] since a [destination] indicator "to" is before it. If [location]$_1$ has already been recognized as a [origin], it is another clue to imply that [location]$_2$ is a [destination]. Since initially the slot context is not available, the slot context is only employed for the semantic re-classification, which will be described in latter section.

2.3.1. Corpus annotation and feature extraction

To automatically learn the context features (the clues stated above) for semantic classification, the training sentences need to be annotated against the semantic frame. Our annotating scenario is relatively simple and can be performed by general developers. For example, for the utterance "*Please tell me how can I go from the people's square to the bund by bus?*", the annotated results are like the following:

FRAME: **ShowRoute**
SLOTS: [route].[origin].[location].(the people's square)
[route].[destination].[location].(the bund)
[route].[transport_type].[by_bus].(bus)

The corresponding slot names can be automatically extracted from the domain model. A domain model is usually a hierarchical structure of the relevant concepts in the application domain. For every occurrence of a concept in the domain model graph, we list all the concept names along the path from the root to its occurrence position and regard their concatenation as a slot name. Thus, the slot name is not flat since it inherits the hierarchy from the domain model. Note that a domain model is necessary for the dialog system development since it also play a critical role in the other components of a dialog system, i.e., dialog management. Therefore, authoring the domain model is not an extra task for our SLU framework.

With provision of the annotated data, we can collect all the literal and slot contexts related to each concept. The examples of features for the concept [location] are illustrated as follows:

(1) *to* within the –3 windows
(2) *from _ to*
(3) *ShowRoute.[route].[origin]* within the ± 2 windows

The former two are literal context features. Feature (1) is a context word that tends to indicate ShowRoute.[route].[destination]. Feature (2) is a collocation that checks for the pattern "from" and "to" immediately before and after the concept [location] respectively, and tends to indicate ShowRoute.[route].[origin]. The third one is a slot context feature, which tends to imply the target concept [location] is ShowRoute.[route].[destination]. Through the feature extraction, we can get an exhaustive list of all the features founded in the training set as well as the corresponding occurrence number. We only used the simple pruning criteria: the features occurred in practically none or all of the training instances are removed. After pruning, we obtained 2,259 literal context features and 369 slot context features for 20 kinds of concepts in our domain.

In nature, these features are equivalent to the rules in the semantic grammar used by the rule-based robust parser. For example, the feature (2) has the same function as the semantic rule "[origin] → from [location] to". One of advantages of our approach is that we can automatically learn the semantic "rules" from the training data rather than manually authoring them. Also, the learned "rules" are intrinsically robust since they may involves gaps, for example, feature (1) allows skipping some noise words between "to" and [location].

2.3.2. Decision List

After the features as well as their corresponding occurrence number are collected, there still exists one problem, namely how to apply these features when predicting a new case. One simple and effective strategy is employed by the decision list

[13, 14], i.e., always applying the strongest features. In a decision list, all the features are sorted in order of descending confidence. When a new target concept is classified, the classifier runs down the list and compares the features against the contexts of the target concept. The first matched feature is applied to make a predication. Obviously, how to measure the confidence of features is a very important issue for the decision list. We use the metric described in [14]. Provided that $P(s_1 \mid f) > 0$, for all i:

$$confidence(f) = \max_i P(s_i \mid f) \quad (1)$$

This value measures the extent to which the context is unambiguously correlated with one particular slot s_i. The probabilities $P(s_i \mid f)$ are estimated using MAP smoothing:

$$P(s_i \mid f) = \frac{C(f, s_i) + \delta}{C(f) + N_s \delta} \quad (2)$$

where $C(f, s_i)$ is the number that the feature f co-occurs with the slots s_i and $C(f)$ is the total number of occurrence of f in the training corpus. A small fixed number δ is used for smoothing and N_s is the total number of possible slots for the target concept.

2.4. Slot-value merging and semantic re-classification

The slot-value merger is to combine the slots assigned to the concepts in an input sentence. It also simultaneously checks the consistency among the identified slot-values. Since the topic-dependent classifiers corresponding to the different concepts are training and running independently, it possibly results in inconsistent predictions. Considering the preprocessed word sequence "*Please tell me how can I go from [location]$_1$ to [location]$_2$ by [bus]*", they are semantically clashed if [location]$_1$ and [location]$_2$ are both classified as Show-Route.[route].[origin]. To relieve this problem, we can use the semantic classifier based on the slot context feature. We apply the context features like, for example, "Show-Route.[route].[origin] within the $\pm k$ windows", which tends to imply Show-Route.[route].[destination]. Note that t the slot context features are no longer order-dependent as the literal context features. The literal contexts reflect the local lexical semantic dependency. The slot contexts, however, are good at capturing the long distance dependency. Therefore, when the slot-value merger finds that two or more slot-value pairs clash, it first anchors the one with the highest confidence. Then, it extracts the slot contexts for the other concepts and passes them to the semantic classification module for re-classification. If the re-classification results still clash, the dialog system can involve the user in an interactive dialog for clarity.

The idea of semantic classification and re-classification can be understood as follows: it first finds the concept or slot islands (like partial parsing) and then links them together. This mechanism is well-suited for SLU since the spoken utterance usually consists of several phrases and noises (restart, repeats and filled pauses, etc) are most often between them [1]. Especially, this phenomena and the out-of-order structures are very frequent in the spoken Chinese utterances.

3. Experiments

Our experiments were carried out in the context of Chinese public transportation information inquiry domain. We collected two kinds of corpus for our domain using the different ways. Firstly, a natural language corpus was collected through a specific website which simulated a dialog system. The user can conduct some mixed-initiative conversational dialogues with it in Chinese by typing. Then we collected 2,286 natural language utterances through this way. It was divided into two parts: the training set contained 1,800 sentences (TR), and the test set contained 486 sentences (TS1). Also, a spoken language corpus was collected through the deployment of a preliminary version of telephone-based dialog system, of which the speech recognizer is based on the speaker-independent Chinese dictation system of IBM ViaVoice Telephony and the SLU component is a robust rule-based parser. The spoken utterances corpus contained 363 spoken utterances. Then we obtained two test set from this corpus: one consisted of the recognized text (TS2); the other consisted of the corresponding transcription (TS3). Due to the unique challenges of Chinese speech recognition (homonyms and tonality problems) and the complexity of our domain (there is a large set of entity names, such as location and street names, among which many pairs of homonyms occur), the Chinese character error rate and concept error rate of TS2 are 35.6% and 41.1% respectively. We defined ten types of topic for our domain: **ListStop**, **ShowFare**, **ShowRoute**, **ShowRouteTime**, etc. The first corpus covers all the ten topic types and the second corpus only covers four topic types. Among the ten topics, **ShowRoute** occurs 71.1% of the time in the first corpus and 78.5% in the second corpus. The total number of Chinese characters appear in the data set is 923. All the sentences were annotated against the semantic frame. In our experiments, all the classifiers (topic classifier and semantic classifier) were trained on the natural language training set (TR) and tested on three test sets (TS1, TS2 and TS3).

Firstly, we used 923 binary Chinese character features for topic classification. As mentioned in Section 2.2, we can also include semantic class labels as features. Using the preprocessor, we substituted those Chinese characters with the corresponding semantic class labels. Table 2 compares the results of topic classification using the two kinds of features on three test sets. The topic classification performance is measured by comparing the topic of a sentence predicated by the topic classifier with the reference topic. The results show that including the semantic class labels as features can significantly improve the topic classification performance.

Table 1. *Topic classification error rate (TER)*

Features	TS1	TS2	TS3
Chinese character	4.7%	3.6%	3.0%
Chinese character + semantic class	2.9%	2.2%	1.4%

In the semantic classification experiments, we first evaluated the performance of semantic classifiers using only literal contexts. Then we evaluated the impact of the semantic re-classification using slot contexts. The performance is measured in terms of slot error rate, i.e., comparing the slots generated by our system with these in the reference annotation and counting the insertion, deletion and substitution error rate.

Here, the slot error rates are based on the identified topics by the best SVM. Table 2 shows that semantic re-classification considerably improves the performance. Due to the high concept error rate of recognized utterances, the performance of semantic classification on the TS2 is relatively poor. However, if considering only the correctly recognized concepts on TS2, the slot error rate is 9.2%.

Table 2. *Slot error rates (SER) of semantic classification*

	TS1	TS2	TS3
One-pass Decision List	9.1%	46.7%	5.0%
Two-pass Decision List (+ Re-classification)	8.4%	45.6%	4.5%

Finally, we compared our system with a rule-based robust semantic parser. The parsing algorithm of this parser is same as the local chart parser used by the preprocessor. The handcrafted grammar for this semantic parser took a linguistic expert one month to develop, which consists of 798 rules (except the lexical rules for named entities such as [loc_name]). A general developer independently annotated the corpus against the semantic frame, which take only four days. Table 3 Shows that our SLU method has better performance in both topic classification and slot identification.

Table 3. *Performance comparison of a rule-based robust semantic parser and our SLU system (TER: Topic Error Rate; SER: Slot Error Rate)*

	TS1		TS2		TS3	
	TER	SER	TER	SER	TER	SER
Rule-based semantic parser	6.8%	11.6%	4.1%	47.9%	3.0%	5.4%
Our system	2.9%	8.4%	2.2%	45.6%	1.4%	4.6%

4. Conclusions and future works

We have presented a novel SLU approach using two successive learners. The preliminary results show that the proposed approach is promising. The proposed approach exhibits the advantages as follows. It has good robustness on processing spoken language: (1) The preprocessor provide the low level robustness; (2) It inherits the robustness of topic classification using statistical pattern recognition techniques; (3) The strategy of first finding the concept or slot islands and then linking them is suited for processing spoken language. It also keeps the understanding deepness: (1) The class of semantic classification is the slot name, which inherits the hierarchy from the domain model. (2) The semantic re-classification mechanism ensures the consistency among the identified slot-value pairs. Moreover, it can make use of topic classification to guide slot filling. Most importantly, it is mainly data-driven and requires only minimally annotated corpus for training.

To study the general applicability of our approach, we intend to evaluate our approach in other domains and languages. We also plan to integrate this understanding system into a whole dialog system. Then, the high level knowledge, such as the dialog context, can also be included as the features of topic and semantic classifiers. Currently, the two successive classifiers in our SLU framework are trained using supervised techniques. We are working on the weakly supervised training techniques for the two classifiers, which are potential to reduce the cost of annotating the training sentences

5. Acknowledgements

The authors would like to thank the anonymous reviewers for their careful reading and helpful suggestions. This work is supported by National Natural Science Foundation of China (NSFC, No. 60496326) and 863 project of China (No. 2001AA114210-11).

6. References

[1] W. Ward and S. Issar, "Recent Improvements in the CMU Spoken Language Understanding System", In Proc. of ARPA Workshop on HLT, 1994.

[2] S. Seneff, "TINA: A natural language system for spoken language applications", Computational Linguistics, vol. 18, no. 1., pp. 61-86, 1992.

[3] J. Dowding, J. M. Gawron, D. Appelt, J. Bear, L. Cherny, R. Moore, and D. Moran, "GEMINI: A natural language system for spoken-language understanding", In Proc. of ACL, Columbus, Ohio, pp. 54-61, 1993.

[4] R. Pieraccini and E. Levin, "A learning approach to natural language understanding", NATO-ASI, New Advances & Trends in Speech Recognition and Coding, Springer-Verlag, Bubion (Granada), Spain, 1993.

[5] S. Miller, R. Bobrow, R. Ingria, and R. Schwartz, " Hidden Understanding Models of Natural Language", In Proc. of ACL, 1994.

[6] Y. Wang and A. Acero, "Grammar learning for spoken language understanding", In Proc. of ASRU Workshop, Madonna di Campiglio, Italy, 2001.

[7] Y. He and S. Young. 2005. "Semantic Processing using the Hidden Vector State Model", Computer Speech and Language 19(1): 85-106.

[8] Y. Wang, "A Robust Parser for Spoken Language Understanding", In Proc. of EUROSPEECH, Budapest, Hungary, 1999.

[9] W. Wu, J. Duan, R. Lu, F. Gao, "Embedded machine learning systems for Robust Spoken Language Parsing", In Proc. of IEEE NLP-KE, Wuhan, China, 2005.

[10] J. Chu-Carroll and B. Carpenter, "Vector-based natural language call routing", Computational Linguistics, vol. 25, no. 3, pp. 361-388, 1999.

[11] Y. Wang, A. Acero, C. Chelba, B. Frey, and L. Wong, "Combination of Statistical and Rule-based Approaches for Spoken Language Understanding", In Proc. of ICSLP, Denver, Colorado, 2002.

[12] C. Chang and C. Lin. 2001. "LIBSVM: a library for support vector machines", Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[13] R. L. Rivest, "Learning decision lists", Machine Learning, 2(3): 229-246, 1987.

[14] D. Yarowsky, "Decision Lists for Lexical Ambiguity Resolution: Application to Accent Restoration in Spanish and French", In Proc. of ACL, 1994.