A Weakly Supervised Learning Approach for Spoken Language Understanding

Wei-Lin Wu, Ru-Zhan Lu, Jian-Yong Duan,
Hui Liu, Feng Gao, Yu-Quan Chen
Department of Computer Science and Engineering
Shanghai Jiao Tong University
Shanghai, 200030, P. R. China
{wu-wl,lu-rz,duan-jy,liuhui,gaofeng,chen-yq}
@cs.sjtu.edu.cn

Abstract

In this paper, we present a weakly supervised learning approach for spoken language understanding in domain-specific dialogue systems. We model the task of spoken language understanding as a successive classification problem. The first classifier (topic classifier) is used to identify the topic of an input utterance. With the restriction of the recognized target topic, the second classifier (semantic classifier) is trained to extract the corresponding slot-value pairs. It is mainly data-driven and requires only minimally annotated corpus for training whilst retaining the understanding robustness and deepness for spoken language. Most importantly, it allows the employment of weakly supervised strategies for training the two classifiers. We first apply the training strategy of combining active learning and self-training (Tur et al., 2005) for topic classifier. Also, we propose a practical method for bootstrapping the topic-dependent semantic classifiers from a small amount of labeled sentences. Experiments have been conducted in the context of Chinese public transportation information inquiry domain. The experimental results demonstrate the effectiveness of our proposed SLU framework and show the possibility to reduce human labeling efforts significantly.

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. Traditionally, there are mainly two mainstreams in the SLU researches: knowledge-based approaches, which are based on robust parsing or template matching techniques (Sneff, 1992; Dowding et al., 1993; Ward and Issar, 1994); and data-driven approaches, which are generally based on stochastic models (Pieraccini and Levin, 1993; Miller et al., 1995). Both approaches have their drawbacks, however. The former approach is cost-expensive to develop 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 associated with knowledge-based approaches, the latter approach often suffers the data sparseness problem and hence needs a 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 (Wang and Acero, 2001) and Hidden Vector State (HVS) model (He and Young, 2005). The two methods require only minimally annotated data (only the semantic frames are annotated).

This paper proposes a novel weakly supervised spoken language understanding approach. Our SLU framework mainly includes two successive classifiers: topic classifier and semantic classifier. The main 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. In particular, the two classifiers are trained using weakly supervised strategies: the former one is trained through the combination of active learning and self-training (Tur et al., 2005), and the latter one

Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006), pages 199–207,
Sydney, July 2006.

is trained using a practical bootstrapping technique.

2 The System Architecture

The semantic representation of an application domain is usually defined in terms of the semantic frame, which contains a frame type representing the topic of the input sentence, and some slots representing the constraints the query goal has to satisfy. Then, the goal of the SLU system is to translate an input utterance into a semantic frame. Besides the two key components, i.e., topic classifier and semantic classifier, our system also contains a preprocessor and a slot-value merger. Figure 1 illustrates the overall system architecture. It also describes the whole SLU procedure using an example sentence.

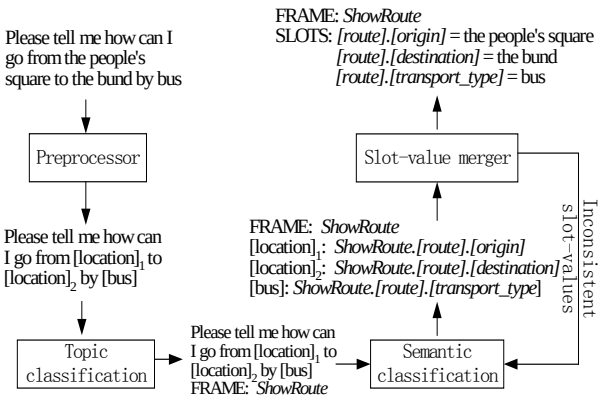

Figure 1: The System architecture[1]

2.1 The Preprocessor

Usually, the preprocessor is to look for the substrings in a sentence that correspond to a semantic class or matching a regular expression and to 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, e.g., "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 (Wang, 1999). 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". However, the robust local parser possibly skips the words in the 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 (Wu et al., 2005). According to our experience, it is fairly easy for a general developer with good understanding of the application to author the small grammar used by the local chart parser and annotate the training cases for the embedded decision tree. The work can be finished in several hours.

2.2 Topic Classification

Given the representation of semantic frame, topic classification can be regarded as identifying the frame type. 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 concept [location] can be filled in all topics is 33 and the corresponding maximum number of slots in a single topic is decreased to 10.

Many statistical pattern recognition techniques have been applied to similar tasks, such as Naïve Bayes, N-Gram and Support Vector Machines (SVMs) (Wang et al., 2002). According to the literature (Wang et al., 2002) and our experiments, the SVMs showed the best performance among many other statistical classifiers. Also, it has been showed that active learning can be effectively applied to the SVMs (Schohn and Cohn, 2000; Tong and Koller, 2000). Therefore, we choose the SVMs as the topic classifier. We resorted to the LIBSVM toolkit (Chang and Lin, 2001) to construct the SVMs for our experiments. Following the practice in (Wang et al., 2002), the SVMs use a 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 semantic class labels (e.g., [location]) as features for topic classification. Intuitively, the class label features are more informative than the

[1] Because the length is limited, in this paper we only illustrate all the example sentences in English, which are Chinese sentences, in fact.

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 slots. 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 [destina tion]. 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.

To learn the topic-dependent semantic classifiers, 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 sentence "*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.

With provision of the annotated data, we can collect all the literal and slot context features 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 of type ShowRoute.[route].[destination]. In nature, these features are equivalent to the rules in the semantic grammar used by the robust rule-based parser. For example, the feature (2) has the same function as the semantic rule "[origin] → from [location] to". The advantage 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].

The next problem is how to apply these features when predicting a new case since the active features for a new case may make opposite predictions. One simple and effective strategy is employed by the decision list (Rivest, 1987), 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 (Yarowsky, 1994; Golding, 1995). 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.

2.4 Slot-value merging and semantic reclassification

The slot-value merger is to combine the slots assigned to the concepts in an input sentence. Another simultaneous task of the slot-value merger is to check the consistency among the identified slot-values. Since the topic-dependent classifiers corresponding to 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 ShowRoute.[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, "ShowRoute.[route].[origin] within the $\pm k$ windows", which tends to imply ShowRoute.[route].[destination]. 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 (Ward and Issar, 1994). Especially, this phenomena and the out-of-order structures are very frequent in the spoken Chinese utterances.

3 Weakly Supervised Training of the Topic Classifier and Topic-dependent Semantic Classifiers

As stated before, to train the classifiers for topic identification and slot-filling, we need to label each sentence in the training set against the semantic frame. Although this annotating scenario is relatively minimal, the labeling process is still time-consuming and costly. Meanwhile unlabeled sentences are relatively easy to collect. Therefore, to reduce the cost of labeling training utterances, we employ weakly supervised techniques for training the topic and semantic classifiers.

The weakly supervised training of the two classifiers is successive. Assume that a small amount of seed sentences are manually labeled against the semantic frame. We first exploit the labeled frame types (e.g. **ShowRoute**) of the seed sentences to train a topic classifier through the combination of active learning and self-training. The resulting topic classifier is used to label the remaining training sentences with the corresponding topic, which are not selected by active learning. Then, we use all the sentences annotated against the semantic frame (including the seed sentences and sentences labeled by active learning) and the remaining training sentences labeled the topic to train the semantic classifiers using a practical bootstrapping technique.

3.1 Combining Active Learning and Self-training for Topic Classification

We employ the strategy of combining active learning and self-training for training the topic classifier, which was firstly proposed in (Tur et al., 2005) and applied to a similar task.

One way to reduce the number of labeling examples is active learning, which have been applied in many domains (McCallum and Nigam, 1998; Tang et al., 2002; Tur et al., 2005). Usually, the classifier is trained by randomly sampling the training examples. However, in active learning, the classifier is trained by selectively sampling the training examples (Cohn et al., 1994). The basic idea is that the most informative ones are selected from the unlabeled examples for a human to label. That is to say, this strategy tries to always select the examples, which will have the largest improvement on performance, and hence minimizes the human labeling effort whilst keeping performance (Tur et al., 2005). According to the strategy of determining the informative level of an example, the active learning approaches can be divided into two categories: uncertainty-based and committee-based. Here, we employ the uncertainty-based strategy for selective sampling. It is assumed that a small amount of labeled examples is initially available, which is used to train a basic classifier. Then the classifier is applied to the unannotated examples. Typically the most unconfident examples are selected for a human to label and then added to the training set. The classifier is re-trained and the procedure is repeated until the system performance converges.

Another alternative for reducing human labeling effort is self-training. In self-training, an initial classifier is built using a small amount of annotated examples. The classifier is then used to label the unannotated training examples. The examples with classification confidence scores

over a certain threshold, together with their predicted labels, are added to the training set to retrain the classifier. This procedure repeated until the system performance converges.

These two strategies are complementary and hence can be combined. The combination strategy is quite straightforward for pool-based training. At each iteration, the current classifier is applied to the examples in the current pool. The most unconfident examples in the pool are selected by active learning and labeled by a human. The remaining examples in the pool are automatically labeled by the current classifier. Then, these two parts of labeled examples are both added into the training set and used for retraining the classifier. Since the LIBSVM toolkit provides the class probability, we directly use the class probability as the confidence score. Our dynamic pool-based (the pool size is n) algorithm of combining active learning and self-training for training the topic classifier is as follows:

1. Given a small amount of human-labeled training set S_t (n sentences) and a larger amount of unlabeled set S_u, build the initial classifier using S_t.
2. While labelers/ sentences are available
   (a) Get n unlabeled sentences from S_u
   (b) Apply the current classifier to n unlabeled sentences
   (c) Select m examples which are most informative to the current classifier and manually label the selected m examples
   (d) Add the m human-labeled examples and the remaining $n-m$ machine-labeled examples to the training set S_t
   (e) Train a new classifier on all labeled examples

3.2 Bootstrapping the Topic-dependent Semantic Classifiers

Bootstrapping refers to a problem of inducing a classifier given a small set of labeled data and a large set of unlabeled data (Abney, 2002). It has been applied to problems such as word-sense disambiguation (Yarowsky, 1995), web-page classification (Blum and Mitchell, 1998), named-entity recognition (Collins and Singer, 1999) and automatic construction of semantic lexicon (Thelen and Riloff, 2003). The key to the bootstrapping methods is to exploit the redundancy in the unlabeled data (Collins and Singer, 1999). Thus, many language processing problems can be dealt using the bootstrapping methods since language is highly redundant (Yarowsky, 1995). The semantic classification problem here also exhibits the redundancy. In the example "*Please tell me how can I go from [location]$_1$ to [location]$_2$ by [bus]?*", there are multiple literal context features which all indicate that [location]$_1$ is of type ShowRoute.[route].[origin], such as:
(1) *from* within the –1 windows;
(2) *from _ to* ;
(3) *to* within the +1 windows.
If the [location]$_2$ has already be recognized as ShowRoute.[route].[destination], thus the slot context feature "ShowRoute.[route].[origin] within the ± 2 windows" is also a strong evidence that [location]$_1$ is of type ShowRoute.[route].[origin]. That is to say, the literal context and slot context features above effectively overdetermine the slot of a concept in the input sentence. Especially, the literal and slot context features can be seen as two natural "views" of an example from the respective of "Co-Training" (Blum and Mitchell, 1998). Our bootstrapping algorithm exploits the property of redundancy to incrementally identify the features for assigning slots of a concept, given a few annotated seed sentences.

The bootstrapping algorithm is performed on each topic T_i ($1 \le i \le n$, n is the number of topic) as follows:

1. For each concept C_j in T_i ($1 \le j \le m$, m is the number of concepts appears in the sentences of topic T_i):

(1.1) Build the two initial classifiers based on literal and slot context features respectively using a small amount of labeled seed sentences.

(1.2) Apply the current classifier based on the literal context feature to the remaining unlabeled concepts in the training sentences belong to topic T_i. Keep those classified slots with confidence score above a certain threshold (In this paper, the threshold is fixed on 0.5).

2. Check the consistency of the classified slots in each sentence. If some slots in a sentence clashed, take the one with the highest confidence score among them and leave the others unlabeled.
3. For each concept C_j in T_i, apply the current classifier based on the slot context to the residual unlabeled concepts. Keep those classi-

fied slots with confidence score above a certain threshold. Repeat Step 3.

4. Augment the new classified cases into the training set and retrain the two classifiers based on literal and slot context features respectively.
5. If new slots are classified from the training data, return to step 2. Otherwise, repeat 2-5 to label training data and keep all new classified slots regardless of the confidence score. Train the two final semantic classifiers based on the literal and context features respectively using the new labeled training data.

4 Experiments and Results

4.1 Data Collection and Experimental Setting

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 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 by typing Chinese queries. 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). 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. 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, the topic classifier and semantic classifiers were trained on the natural language training set (TR) and tested on three test sets (TS1, TS2 and TS3).

The performance of topic classification and semantic classification are measured in terms of topic error rate and slot error rate respectively. Topic performance is measured by comparing the topic of a sentence predicated by the topic classifier with the reference topic. The slot error rate is measured by counting the insertion, deletion and substitution errors between the slots generated by our system and these in the reference annotation.

4.2 Supervised Training Experiments

Firstly, in order to validate the effectiveness of our proposed SLU system using successive learners, 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]). In our SLU system, we first use the SVMs to identify the topic and then apply the semantic classifier (decision list) related to the identified topic to assign the slots to the concepts. The SVMs used the augmented binary features (923 Chinese characters and 20 semantic class labels). A general developer independently annotated the TR set against the semantic frame, which took only four days. Through feature extraction from the TR set and feature pruning, we obtained 2,259 literal context features and 369 slot context features for 20 kinds of concepts in our domain. Table 1 Shows that our SLU method has better performance than the rule-based robust parser in both topic classification and slot identification. 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 identified concepts on TS2, the slot error rate is 9.2%. Note that, since the TS2 (recognized speech) covers only four types of topic but TS1 (typed utterance) covers ten topics, the topic error on the TS2 (recognized speech) is lower than that on TS1.

Table 1 also compares our system with the two-stage classification with the reversed order. Another alternative for our system is to reverse the two main processing stages, i.e., finding the roles for the concepts prior to identifying the topic. For instance, in the example sentence in Fig.1, the concept (e.g., [location]) in the preprocessed sequence is first recognized as slots (e.g., [route].[origin]) before topic classification.

Therefore, the slots like [route].[origin] can be included as features for topic classification, which is deeper than the concepts like [location] and potential to achieve improvement on performance of topic classification. This strategy was adopted in some previous works (He and Young, 2003; Wutiwiwatchai and Furui, 2003). However, the results indicate that, at least in our two-stage classification formwork, the strategy of identifying the topic before assigning the slots to the concepts is more optimal. According to our error analysis, the unsatisfied performance of the reversed two-stage classification system can be explained as follows: (1) Since the semantic classification is performed on all topics, the search space is much bigger and the ambiguities increase. This deteriorates the performance of semantic classification. (2) In the case that the slots and Chinese characters are included as features, the topic classifier relies heavily on the slot features. Then, the errors of semantic classification have serious negative effect on the topic classification.

Table 1: Performance comparsion of the rule-based robust semantic parser, the reversed two-stage classification system and our SLU systems (TER: Topic Error Rate; SER: Slot Error Rate; DL: Decision List)

	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
Reversed two-stage classification system	4.9	11.1	3.6	47.4	2.5	4.9
Our system	2.9	8.4	2.2	45.6	1.4	4.6

4.3 Weakly Supervised Training Experiments

4.3.1 Active Learning and Self-training Experiments for Topic Classification

In order to evaluate the performance of active learning and self-training, we compared three sampling strategies: random sampling, active learning only, active learning and self-training. At each iteration of pool-based active learning and self-training, we get 200 sentences (i.e., the pool size is set as 200) and select 50 most unconfident sentences from them for manually labeling and exploit the remaining sentences using self-training. All the experiments were repeated ten times with different randomly selected seed sentences and the results were averaged. Figure 1 plots the learning curves of three strategies trained on TR and tested on the TS1 set. It is evident that active learning significantly reduces the need for labeled data. For instance, it requires 1600 examples if they are randomly chosen to achieve a topic error rate of 3.2% on TS1, but only 600 actively selected examples, a saving of 62.5%. The strategy of combing active learning and self-training can further improve the performance of topic classification compared with active learning only with the same amount of labeled data.

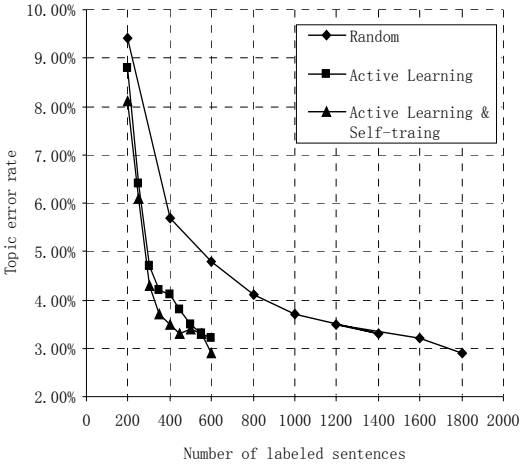

Figure 2: Learning curves using different sampling strategies.

We also evaluated the performance of topic classification using active learning and self-training with the pool size of 200 on the three test sets. Table 2 shows that active learning and self-training with the pool size of 200 achieves almost the same performance on three test sets as random sampling, but requires only 33.3% data.

Table 2: The topic error rate using active learning and self-training with pool size of 200 on the three test sets (AL: Active Learning)

	TS1 (%)	TS2 (%)	TS3 (%)	Labeled Sent.(#)
Random	2.9	2.2	1.4	1,800
AL	3.2	2.5	1.7	600
AL & self-training	2.9	2.5	1.4	600

4.3.2 Bootstrapping Experiments for Semantic Classification

As stated before, the bootstrapping procedure begins with a small amount of sentences annotated against the semantic frame, which is the initial seed sentence or annotated by active learning, and the remaining training sentences, the topics of which are machine-labeled by the resulting topic classifier. For example, in the

weakly supervised training scenario with the pool size of 200, the active learning and self-training procedure ran 8 iterations. At each iteration, 50 sentences were selected by active learning. So the total number of labeled sentences is 600. We compared our bootstrapping methods with supervised training for semantic classification. We tried two bootstrapping methods: using only the literal context features (Bootstrapping 1) and using the literal and slot context features (Bootstrapping 2). If the step 4 of the bootstrapping algorithm in Section 3.2 is canceled, the new bootstrapping variation corresponds to Bootstrapping 2. Also, we repeated the experiments ten times with different labeled sentences and the results were averaged. Figure 3 plots the learning curves of bootstrapping and supervised training with different number of labeled sentences on the TS1 set. The results indicate that bootstrapping methods can effectively make use of the unlabeled data to improve the semantic classification performance. In particular, the learning curve of bootstrapping 1 achieves more significant improvement than the curve of bootstrapping 2. It can be explained as follows: including the slot context features further increases the redundancy of data and hence corrects the initial misclassified cases by the semantic classifier using only literal context features or provides new cases.

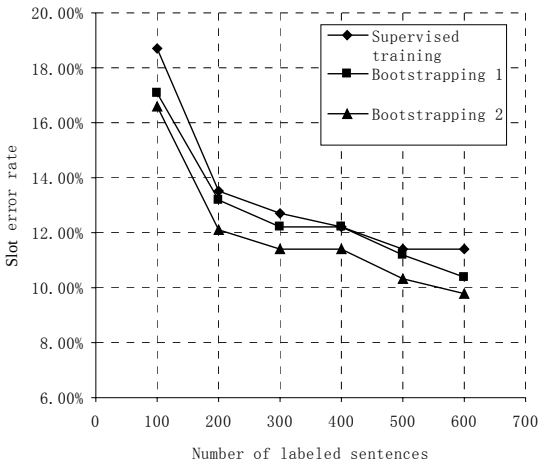

Figure 3: Learning curves of bootstrapping methods for semantic classification on TS1.

Finally, we compared two SLU systems through weakly supervised and supervised training respectively. The supervised one was trained using all the annotated sentences in TR (1800 sentences). In the weakly supervised training scenario (the pool size is still 200), The topic classifier and semantic classifiers were both trained using only 600 labeled sentences. Table 3 shows that the weakly supervised scenario achieves comparable performance to the supervised one, but requires only 33.3% labeled data.

Table 3: Performance comparison of two SLU systems through weakly supervised and supervised training on the three test sets (TER: Topic Error Rate; SER: Slot Error Rate)

	TS1		TS2		TS3	
	TER (%)	SER (%)	TER (%)	SER (%)	TER (%)	SER (%)
Supervised	2.9	8.4	2.2	45.6	1.4	4.6
Weakly Supervised	2.9	9.7	2.5	44.8	1.4	5.7

5 Conclusion and Future work

We have presented a new SLU framework using two successive classifiers. The proposed framework exhibits the advantages as follows.

- It has good robustness on processing spoken language: (1) The preprocessor provides the low level robustness. (2) It inherits the robustness of topic classification using statistical pattern recognition techniques. It can also make use of topic classification to guide slot filling. (3) The strategy of first finding the concepts 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.
- It is mainly data-driven and requires only minimally annotated corpus for training. Most importantly, our proposed SLU framework allows the employment of weakly supervised strategies for training the two classifiers, which can reduce the cost of annotating labeled sentences.

The future work includes further evaluation of our approach in other application domains and languages. We also plan to integrate this understanding system into a whole dialog system. Then, high level knowledges, such as the dialog context, can also be included as the features of topic and semantic classifiers. Moreover, currently, the topics are manually defined through examination of the example sentences by human. Then, it is worthwhile to investigate how to appropriately define topics and the probability of

exploiting the sentence clustering techniques to facilitate the topic (frame) designment.

6 Acknowledgements

The authors would like to thank three 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).

References

S. Abney. 2002. *Bootstrapping*. In Proc. of ACL, pp. 360-367, Philadelphia, PA.

A. Blum and T. Mitchell. 1998. *Combining labeled and unlabeled data with co-training*. In Proc. of COLT, Madison, WI.

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

D. Cohn, L. Atlas and R. Ladner. 1994. *Improving generalization with active learning*. Machine Learning 15, pp.201-221.

M. Collins and Y. Singer.1999. *Unsupervised models for named entity classification*. In Proc. of EMNLP.

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

R. Golding. 1995. *A Bayesian Hybrid Method for Context-sensitive Spelling Correction*. In Proc. 3rd Workshop on Very Large Corpora, Boston, MA.

Y. He and S.J. Young. 2003. *A Data-Driven Spoken Language Understanding System*. IEEE Workshop on Automatic Speech Recognition and Understanding, US Virgin Islands.

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

A. McCallum and K. Nigam.1998. *Employing EM and pool-based active learning for text classification*. In Proc. of ICML.

S. Miller, R. Bobrow, R. Ingria, and R. Schwartz. 1994. *Hidden Understanding Models of Natural Language*. In Proc. of ACL, pp. 25-32.

R. Pieraccini and E. Levin. 1993. *A learning approach to natural language understanding*. NATO-ASI, New Advances & Trends in Speech Recognition and Coding, Springer-Verlag, Bubion, Spain.

R. L. Rivest. 1987. *Learning decision lists*. Machine Learning, 2(3):229--246, 1987.

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

G. Schohn and D. Cohn. 2000. *Less Is More: Active Learning with Support Vector Machines*. In Proc. of ICML, pp. 839-846.

M. Tang, X. Luo, S. Roukos.2002. *Active learning for statistical natural language parsing*. In Proc. of ACL, Philadelphia, Pennsylvania.

M. Thelen and E. Riloff. 2002. *A Bootstrapping Method for Learning Semantic Lexicons using Extraction Pattern Contexts*. In Proc. of EMNLP'02.

S. Tong and D. Koller. 2000. *Support Vector Machine Active Learning with Applications to Text Classification*. In Proc. of ICML, pp. 999-1006.

G. Tur, D. Hakkani-Tür, Robert E. Schapire. *Combining Active and Semi-Supervised Learning for Spoken Language Understanding*. Speech Communication, Vol. 45, No. 2, pp. 171-186, 2005.

Y. Wang. 1999. *A Robust Parser for Spoken Language Understanding*. In Proc. of EUROSPEECH. Budapest, Hungary.

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

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

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

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

C. Wutiwiwatchai and S. Furui. 2003. *Combination of Finite State Automata and Neural Network for Spoken Language Understanding*. In Proc. of EUROSPEECH2003, Geneva, Switzerland.

D. Yarowsky. 1994. *Decision Lists for Lexical Ambiguity Resolution: Application to Accent Restoration in Spanish and French*. In Proc. of ACL 1994, pp. 88-95.