Available online at www.sciencedirect.com

ScienceDirect

Computer Speech and Language 24 (2010) 358–382

COMPUTER SPEECH AND LANGUAGE

www.elsevier.com/locate/csl

Spoken language understanding using weakly supervised learning ☆

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, PR China

Received 7 April 2007; received in revised form 16 April 2009; accepted 11 May 2009
Available online 20 May 2009

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 two-stage classification problem. Firstly, the topic classifier is used to identify the topic of an input utterance. Secondly, with the restriction of the recognized target topic, the slot classifiers are 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. More importantly, it allows that weakly supervised strategies are employed for training the two kinds of classifiers, which could significantly reduce the number of labeled sentences. We investigated active learning and naive self-training for the two kinds of classifiers. Also, we propose a practical method for bootstrapping topic-dependent slot classifiers from a small amount of labeled sentences. Experiments have been conducted in the context of the Chinese public transportation information inquiry domain and the English DARPA Communicator domain. The experimental results show the effectiveness of our proposed SLU framework and demonstrate the possibility to reduce human labeling efforts significantly.

Keywords: Spoken language understanding; Spoken dialogue system; Topic classification; Active learning; Self-training; Bootstrapping

1. Introduction

Spoken dialogue systems have been attracting extensive interests from the research and industrial communities since they provide a natural interface between human and computer, which has such potential benefits as remote or hands-free access, ease of use, naturalness, and greater efficiency of interaction (Walker et al., 1997). In recent years, many spoken dialogue systems have appeared in a variety of application domains, including customer service (Price, 1990), information inquiring (Blomberg et al., 1993; Lamel et al., 1997; Zue et al.,

☆ This paper is an expanded version of two papers presented at ICSLP-2006 (Wu et al., 2006a) and EMNLP-2006 (Wu et al., 2006b). This work is supported by National Natural Science Foundation of China (NSFC, No. 60496326) and 863 Project of China (No. 2001AA114210-11).

* Corresponding author. Tel.: +86 21 32261098.
E-mail address: wu-wl@cs.sjtu.edu.cn (W.-L. Wu).

0885-2308/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.csl.2009.05.002

2000), call routing (Gorin et al., 1997), planning (Allen et al., 1995), etc. The success of spoken dialogue systems relies on the correct recognition of not only what is said, which is achieved by Automatic Speech Recognition (ASR), but also what is meant, which is accomplished by Spoken Language Understanding (SLU) (Wang et al., 2005). SLU is one of the key components in a spoken dialogue system. Its task is to identify the user's goal and extract from the input utterance the information needed to complete the query.

Although it focus only on limited domains, SLU still faces great challenges. One challenge is the robustness problem. In addition to the difficulties intrinsic to natural language processing, the speech recognizer inevitably makes errors. Also, spoken language is plagued with a large set of spontaneous speech phenomena such as false start, self-correction, repetitions and hesitations, ellipsis, out-of-order structures and so on. Thus, the performance of the SLU component should deteriorate gracefully when the input utterances are ill-formed. Another challenge is the portability problem, which relates to how flexible new SLU components for new applications or new languages can be built quickly at a reasonable cost (Gao et al., 2005). Currently, the development of spoken language systems relies often heavily on human works, which has been one of the main bottlenecks for rapid development of spoken dialogue systems. The rule-based SLU approaches require the linguistic experts to handcraft the domain-specific grammar for parsing, which is a time-consuming, laboursome and error-prone task. On the other hand, although requiring very little a-priori knowledge handcrafted by the linguistic experts, the general statistical SLU approaches need a large amount of labeled data to achieve reasonable performance. These drawbacks prevent the efficient portability of the SLU component to new domains and languages.

In this paper, we try to propose a robust and portable approach for spoken language understanding in hope that this approach has such desirable properties as follows:

- It should have good robustness for ill-formed spoken utterances while keeping the understanding deepness.
- It may be basically data-driven and requires only minimally annotated data for training. Therefore, it can be easily portable across different domains and languages.
- It can be trained using the weakly supervised learning approaches and hence further reduce the cost of labeling training utterances.

The remainder of this paper is organized as follows. The next section introduces the related works about SLU. Section 3 presents our SLU framework and describes its components in depth. Section 4 focuses on the weakly supervised training approaches for our SLU framework. Section 5 gives the experimental setup and results. Finally, Section 6 concludes the paper and gives the future works.

2. Related works

Generally, there are two mainstreams in the SLU research: rule-based approaches and data-driven approaches. These two kinds of approaches also can be combined.

2.1. The rule-based approaches

Traditionally, the SLU components in most spoken dialogue systems have been based on grammar-based parsers, which translate the input sentence into a parse tree. Usually, the hand-crafted grammar used by the parser interleaves syntax and semantics (Seneff, 1992; Dowding et al., 1993), or is purely semantic (Ward and Issar, 1994; Wang, 1999). Thus, the key information conveyed by an input sentence can be directly read from the corresponding parse tree and filled into slots of the semantic frame. To account for the spontaneous input utterance, robust parsing strategies are often applied, such as partial parsing and word skipping ability.

Although the rule-based parsers perform well, their development costs are normally very expensive. The task of manually authoring the grammar is time-consuming, laboursome and requires linguistic skills. It is also daunting to maintain and augment the grammar. These drawbacks make it difficult and expensive to adapt the rule-based SLU systems to new domains and languages. In addition, in the face of the spoken input with speech recognition errors and unforseen spontaneous phenomena, typical rule-based parsers are brittle. Certainly, the rule-based approaches have a major advantage that they do not need a large amount of annotated training data. This property makes it more preferable at the initial system development phase since there is little training data at that time.

2.2. The data-driven approaches

Many data-driven SLU systems depend on statistical models to derive the corresponding semantic representation from an input utterance. Various statistical models have been employed for semantic decoding, which can be automatically learned from the labeled training data. A simple but effective semantic decoding model is the Hidden Markov Model (HMM), which was adopted in the AT&T's CHRONUS (Pieraccini and Levin, 1993) and LIMSI-CNRS systems (Minker et al., 1996). The shortage of the HMM-based model lies in its inability to represent the embedded structures. A naturally extended idea is the Probabilistic Context-Free Grammar (PCFG) style model. A typical SLU system based on the PCFG model was the BBN's HUM system (Miller et al., 1994). Although the expressive power is strengthened, it typically requires a fully annotated corpus in order to reliably estimate an accurate PCFG style model. More recently, He and Young (2005) proposed a compromised model named Hidden Vector State (HVS) model, which is essentially a stochastic push-down automaton. It requires only minimally annotated data (i.e., annotated against the semantic frame). Some researchers introduced the SLU models based on statistical machine translation technology (Pietra et al., 1997; Macherey et al., 2001), which translates the input utterances into formal languages for meaning representation. Decision trees were also applied to language understanding, which can be automatically learned from semantically annotated data (Kuhn and De Mori, 1995). In addition, in some applications such as the automatic call routing system (Gorin et al., 1997; Carpenter and Chu-Carroll, 1998; Gupta et al., 2006), the task of SLU is just to classify an input utterance to one of the predefined topics. It is a typical pattern classification problem and hence suitable to be dealt with by various kinds of statistical classification techniques.

The statistical SLU systems can be automatically trained from the semantically annotated data. They tend to be more robust since they are able to model the variations found in real data (He and Young, 2005). More importantly, they reduce the heavy authoring costs associated with the rule-based systems since annotating the sentences is much easier than handcrafting the grammar rules. However, the statistical approaches often suffer from the data sparseness problem. In order to reliably estimate an robust and accurate model, a large amount of semantically annotated data (even fully annotated tree-bank) are required, which is not very practical for most real applications.

Another direction is to automatically (or semi-automatically) induce the grammar from training data. The early work on grammar inference focused on the automatic learning of finite state automata (Fu and Booth, 1975). Stolcke and Omohundro (1994) proposed a Bayesian approach to learn stochastic automata. As Wang and Acero (2001) stated, the problem of automatic grammar inference exhibited great theoretical complexity. Therefore, most grammar learning works in the SLU area are engineering-oriented, which aim at semi-automatically deriving the good quality grammar from training data. One practice is to automatically or semi-automatically acquire the domain-specific language structures for SLU using distributional clustering techniques (Wang and Waibel, 1998; Pargellis et al., 2001; Meng and Siu, 2002). Wang and Acero (2001) proposed a semi-automatic grammar learning methodology by taking advantage of multiple information sources, such as automatically generated template grammar from semantic schema, the semantically annotated corpus, syntactic constraints and grammar library. Although the grammar learning approaches harbor the similar advantages as the statistical models, they still face the data sparseness problem. In order to infer a good quality grammar, the grammar learning approaches often require a large amount of annotated data or linguistic experts' intervention.

2.3. The hybrid approaches

An emerging trend of SLU is to combine the rule-based and data-driven methods in order to make use of their advantages. Wang et al. (2002) investigated a hybrid SLU approach, in which a statistical classifier was first applied to topic identification and then a robust rule-based parser was used to extract the detailed information. In contrast, Wutiwiwatchai and Furui (2003) proposed a multi-stage approach, which first extracted concepts via weighted finite state transducers and then identified the goal using a statistical classifier. Wang et al. (2006) studied the integration of prior knowledge and statistical learning in a SLU framework based on conditional model. In addition, Rochery et al. (2002) proposed to combine the hand-crafted rules and the statistics of training data for the call routing system.

3. A SLU approach based on two-stage classification

3.1. General knowledge source for SLU

In order to enable a spoken dialogue system to support a conversation between a human and an information back-end, it is important to model the semantic structure of the corresponding application domain. Usually, The semantic structure of an application domain is defined in terms of a set of semantic frames, which is often called domain model. A semantic frame contains a frame type representing the topic of the input sentence, and some slots representing the constraints the query goal has to satisfy. For example, [Route]([Origin]([location]) [Dest]([location])) represents a [Route] frame with two slots [Origin] (stands for "origin") and [Dest] (stands for "destination"), which both require a location name as their fillers. A frame can also nest another frame as one of its slots. Thus, the domain model is a hierarchical structure of the relevant concepts in the application domain. Fig. 1 shows a partial domain model in the Chinese public transportation information inquiry domain. The terminal concepts (enclosed by the rectangle in the Fig. 1) in the domain model are the basic semantic classes, which is associated the entity names such as road names.[1] The entries for these basic semantic classes are also collected, for example, the semantic class [Road_Name] includes all the road names. The domain model is not only an important knowledge source for SLU, but also used by the other components in a dialog system, especially the dialog manager.

3.2. System architecture

In this paper, we proposes a new SLU approach using weakly supervised learning. Our SLU framework mainly consists of two kinds of classifiers: topic classifier and slot classifier (Wu et al., 2006a). Besides these two key components, our system also contains a preprocessor and a slot-value merger. Fig. 2 illustrates the overall system architecture. It also describes the whole SLU procedure using an example sentence in the context of Chinese public transportation information inquiry domain. Firstly, the preprocessor recognizes the concepts in an input utterance, for example, the location names such as "人民广场 (the People's Square)". Secondly, the topic classifier is used to assign an input utterance with the corresponding topic, i.e., the frame type. Thirdly, the slot classifiers are trained to fill the recognized concepts into the possible slots associated with the tagged frame type. For example, the location name "人民广场 (the People's Square)" is filled into the slot ShowRoute.[Route].[Origin]. Finally, the extracted slot-value pairs are combined into one or more consistent semantic frames for the input sentence.

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 kinds of classifiers are trained using weakly supervised strategies (Wu et al., 2006b), which can reduce human labeling cost significantly. The following subsections will describe the implementation details of each component.

3.2.1. The preprocessor

Usually, the preprocessor is used to look for the sub-strings in a sentence that correspond to semantic classes or matching regular expressions and to replace them with the class labels. For example, "华山路 (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., "华山路1954号 (1954 Huashan Road)" can be recognized as [address] through matching a rule like "[address] → [road_name][number] 号". Namely,

[1] Note that the terms "concept" and "semantic class label" are used interchangeably in this paper.

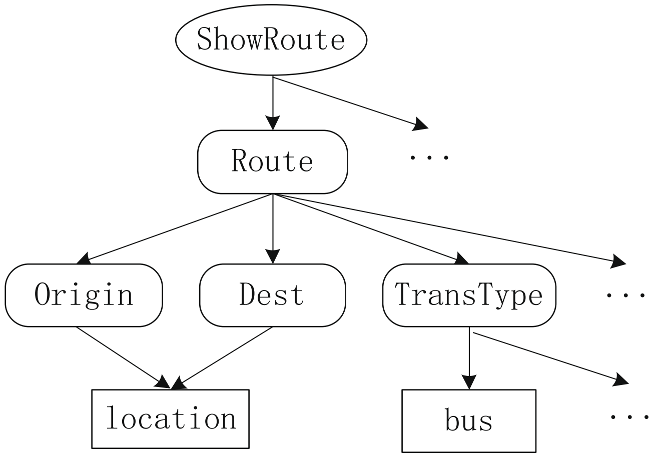

Fig. 1. Partial domain model in the Chinese public transportation information inquiry domain.

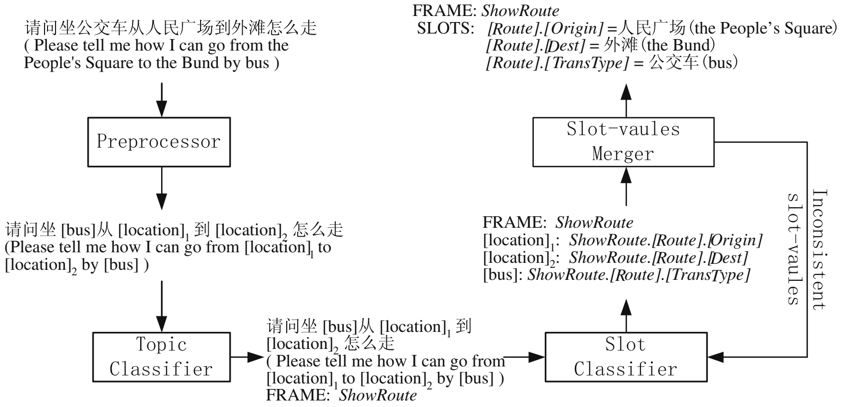
请问坐公交车从人民广场到外滩怎么走
(Please tell me how I can go from the People's Square to the Bund by bus)
Preprocessor
请问坐 [bus]从 [location]$_1$ 到 [location]$_2$ 怎么走
(Please tell me how I can go from [location]$_1$ to [location]$_2$ by [bus])
Topic Classifier
请问坐 [bus]从 [location]$_1$ 到 [location]$_2$ 怎么走
(Please tell me how I can go from [location]$_1$ to [location]$_2$ by [bus])
FRAME: ShowRoute
Slot Classifier
FRAME: ShowRoute
[location]$_1$: ShowRoute.[Route].[Origin]
[location]$_2$: ShowRoute.[Route].[Dest]
[bus]: ShowRoute.[Route].[TransType]
Slot-vaules Merger
Inconsistent slot-vaules
FRAME: ShowRoute
SLOTS: [Route].[Origin] =人民广场 (the People's Square)
[Route].[Dest] = 外滩 (the Bund)
[Route].[TransType] = 公交车 (bus)

Fig. 2. The SLU framework based on two-stage classification.

the preprocessor recognizes the concepts in the input sentence which can be filled into the slots. The preprocessor is implemented with a local chart parser, which is a modified version 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 low level robustness. For example, "华山路嗯1954号 (1954 uh Huashan Road)" can still be recognized as [address] by skipping the pause "嗯(uh)". However, the robust local parser may skip the words in the sentence by mistake and produces incorrect concepts. To avoid this side-effect, this local parser exploits an embedded decision tree for pruning, the details of which 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. This work can be finished in several hours.

The preprocessor can bring at least the following benefits:

- It can provide deeper features for the topic classifier and hence improve the performance of topic classification.
- It can reduce the number of the slot classifiers and make the parameter estimation of the slot classification model more reliable. For example, since the preprocessor can cluster the concept [address] and [loc_name] into a high-level concept [location] through matching the rules [location] → [address] and [location] → [loc_name], the slot classifiers for the concepts [address] and [loc_name] can be merged.

3.2.2. Topic classification

Topic classification is a subproblem of SLU in dialogue systems, whose goal is to identify the topics of the input utterances. A straightforward application of topic classification among many others is call routing (Gorin et al., 1997; Carpenter and Chu-Carroll, 1998; Tur et al., 2005). Given the representation of semantic frame, topic classification can be regarded as identifying the frame type. It is suitable to be dealt with by pattern recognition techniques. The application of statistical pattern techniques to topic classification can improve the robustness of the whole understanding system. Furthermore, in our system, topic classification can greatly reduce the search space of the subsequent slot classification and hence improve the slot classification performance. For example, the total number of possible slots in all types of topics for the concept [location] is 33 and the corresponding maximum number of slots in a single topic is only 10.

Many statistical pattern recognition techniques have been applied to the topic classification problem. Various methods have been proposed in the call routing systems, such as maximum a posteriori (MAP)-based method (Gorin et al., 1997), vector-based method (Carpenter and Chu-Carroll, 1998), boosting algorithm (AdaBoost) (Tur et al., 2005). Naive Bayes, N-Gram, Support Vector Machines (SVM) and artificial neural network have also been applied to topic classification (Wang et al., 2002; Wutiwiwatchai and Furui, 2003). According to the literature (Wang et al., 2002) and our experiments, the SVMs showed better performance than many other statistical classifiers. Also, it has been shown 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. The SVMs learning method is well-founded in terms of computational learning theory (Vapnik, 1998). Its basic idea can be seen as an attempt to find a hyper-surface among the space of possible inputs of feature vectors. This hyper-surface (decision surface) separates positive training examples from negative ones by the maximum margin with respect to the two classes. The SVMs is independent of the dimensionality of the feature space, which allows it to handle a large feature space.

In our work, we resorted to the LIBSVM toolkit (Chang and Lin, 2001) to construct the SVM topic classifier, which uses a binary-valued feature vector. If the simplest feature (Chinese character) is used, each query is converted into a feature vector $\overrightarrow{ch} = \langle ch_1, \ldots, ch_{|\overrightarrow{ch}|} \rangle$ ($|\overrightarrow{ch}|$ is the total number of Chinese characters occurring in the training corpus) with binary valued elements: 1 if a given Chinese character is in the 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 Chinese character features. At the same time, including class labels as features can also relieve the data sparseness problem.

3.2.3. Topic-dependent slot classification

The slot classification is a slot-filling task, i.e., assigning the most likely slot to the concept. It can also be modeled as a classification problem since the number of possible slot names for each concept is limited. Informally, let’s consider the example sentence in Fig. 2. After the preprocessing and topic classification, we get the preprocessed result “请问坐$[bus]$从$[location]_1$到$[location]_2$怎么走 (Please tell me how I can 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 lexical context. Intuitively, we can infer that $[location]_2$ is a destination since a destination indicator “到(to)” is before it. If $[location]_1$ has already been recognized as an origin, it is another clue to imply that $[location]_2$ is a destination. That is to say, in order to assign the target concepts with the corresponding slots, the slot classifier need to learn the features that characterize the context in which each slot tends to appear.

To automatically extract the aforementioned context features of slot classification, the training sentences need to be annotated against the semantic frame. This annotating paradigm is relatively simple and can be performed by general developers. For example, for the sentence “请问坐公交车从人民广场到外滩怎么走 (Please tell me how I can go from the People’s Square to the Bund by bus)”, the annotated results are like the Table 1. The corresponding slot names can be automatically extracted from the pre-defined domain model. For every occurrence of a terminal 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. To cater for irrelevant concepts and negative concepts, the dummy slots and negative slots are added. For example, ShowRoute.[Dummy_location]

Table 1
The human-annotated results against semantic frame for an example sentence.

FRAME:	ShowRoute
SLOTS:	[Route].[Origin].[location].{人民广场 (the People's Square)}
	[Route].[Dest].[location].{外滩 (the Bund)}
	[Route].[TransType].[location].{公交车 (bus)}

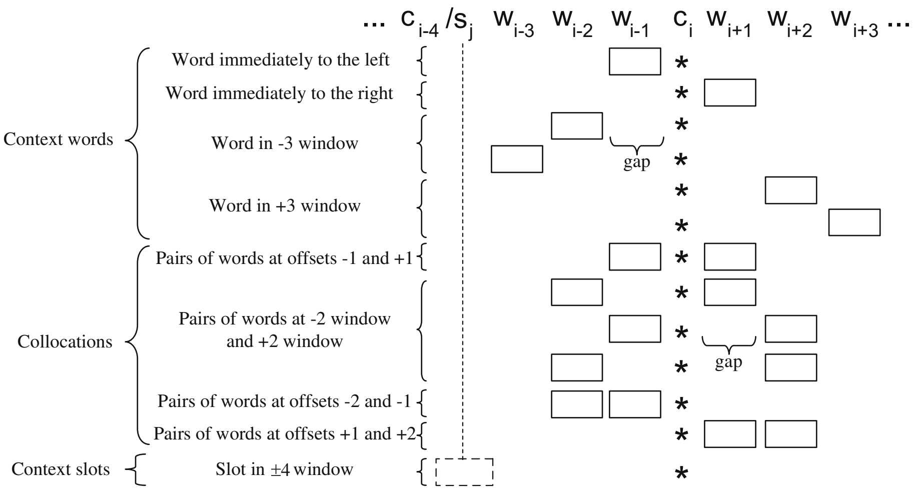
... c_{i-4} /s_j w_{i-3} w_{i-2} w_{i-1} c_i w_{i+1} w_{i+2} w_{i+3} ...
Context words
Word immediately to the left
Word immediately to the right
Word in -3 window
gap
Word in +3 window
Collocations
Pairs of words at offsets -1 and +1
Pairs of words at -2 window and +2 window
gap
Pairs of words at offsets -2 and -1
Pairs of words at offsets +1 and +2
Context slots
Slot in ±4 window

Fig. 3. The features for slot classification: context words, collocations and context slots. The first line is a fragment of a labelled sentence, where the concept C_i is the target concept and the concept C_{i-4} is labelled with the slot S_j. The asterisk denotes the place of the concept to be classified and the boxes denote the positions where a feature will be sought.

represents an irrelevant location and ShowRoute.[Route].[NEG_Dest] represents a destination which is disabled by the user's denial or correction.[2] As mentioned in Section 3.1, a domain model is necessary for the dialog system development since it is also used by the dialog manager. Therefore, authoring the domain model is not an extra requirement of our SLU framework. At the same time, it is relatively easy for a general developer with good understanding of the application domain to define a domain model.

With provision of the annotated data, we can automatically collect all the lexical and slot context features related to each concept. We used two types of lexical context features: context words and collocations. Context words refer to the particular words or concepts within $\pm k_w$ window of the target concept. Collocations refer to the patterns of up to m words or concepts (in this paper, m is set to 2) within $\pm k_c$ window around the target concept. Here, it does not require that the collocations should be patterns of contiguous words or concepts. On the contrary, they can involve gaps. For the slot context feature, we only consider the context slots, which test for the presence of a particular slot within $\pm k_s$ window around the target concepts. In our experiments, the maximal values of k_w, k_c and k_s were set to 3, 2 and 6. Fig. 3 illustrates how the context words, collocations and context slots are extracted. In this figure, the concept C_i is the target concept and the previous concept C_{i-4} has been already tagged with the slot S_j. Since the slot context is not initially available, the slot context is only employed for the slot re-classification, which will be described in latter section.

More specifically, the feature examples for the concept [location] are illustrated as follows:

(1) "到(to)" within the −3 windows
(2) "从(from)" __ "到(to)"
(3) ShowRoute.[Route].[Origin] within the ±2 windows

[2] For instance, in an input sentence "我要去人民广场不对外滩 I want to go to the People's Square no the Bund", the first location "人民广场 the People's Square" is of the type ShowRoute.[Route].[NEG_Dest].

The former two are lexical context features. The first feature is a context word that tends to indicate ShowRoute.[Route].[Dest]. The second one is a collocation that checks for the pattern "从(from)" and "到(to)" immediately before and after the concept [location] respectively, which tends to indicate ShowRoute.[Route].[Origin]. The third one is a slot context feature, which hints the target concept [location] is of type ShowRoute.[Route].[Dest]. In nature, these features are equivalent to the semantic grammar rules used by the robust rule-based parser. For example, the second feature has the same function as the semantic grammar rule "[Origin] → 从(from) [location] 到(to)". In order to handle the ill-formed input utterances, the robust rule-based parsers usually make use of hand-coded rules involving gaps or the skip parsing mechanism to achieve robustness. Motivated by this reason, we also learn the "rules" involving gaps, which are robust in the sense that they are effective in ignoring irrelevant words that do not help for filling a slot. For example, the first feature allows some noise words between "到(to)" and [location] to be skipped. However, the advantage of our approach is that we can automatically learn the semantic grammar "rules" from the training data rather than manually authoring them. Through the feature extraction, we get an exhaustive list of all the features founded in the training set. In the extraction process, statistics of the occurrence of the features are collected as well. We used a simple criteria to prune unreliable features: the features had fewer occurrence than a certain threshold were removed (in our experiments, the threshold of occurrence was set as 5).

After the features are extracted, the next problem is how to apply these features when predicting a new case since the multiple active features for this case may make opposite predictions. One simple and effective strategy was employed by the decision list (Rivest, 1987), i.e., always applying the strongest feature. 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 feature list and compares the features against the contexts of the target concept. The first matched feature is applied to make a prediction. Obviously, how to measure the confidence of features is a very important issue for the decision list. Yarowsky (1994) proposed a metric as follows:

$$confidence(f) = abs\left(log\left(\frac{Pr(ss_1|f)}{Pr(ss_2|f)}\right)\right) \tag{1}$$

This is for the case of two-classes classification (two slots ss_1 and ss_2). In our experiments, we used its identical metric for multiple-classes classification as described in Golding (1995). Provided that $Pr(ss_i|f) > 0$ for all i:

$$confidence(f) = \max_i Pr(ss_i|f) \tag{2}$$

This value measures the extent to which the context feature is unambiguously correlated with one particular slot. The probability $Pr(ss_i|f)$ is estimated by Maximum Likelihood Estimation (MLE) as follows: $Pr(ss_i|f) = \frac{count(f,ss_i)}{count(f)}$, where $count(f, ss_i)$ represents the number that the context feature f co-occurs with the slot ss_i and $count(f)$ is the total number of occurrence of f in the training corpus. To address the data sparseness problem, we apply a simple smoothing technique:

$$Pr(ss_i|f) = \frac{count(f, ss_i) + \beta}{count(f) + N_{ss}\beta} \tag{3}$$

where β is a small fixed number and N_{ss} is the total number of possible slots for the target concept.

3.2.4. Slot-value merging and slot re-classification

The slot-value merger is used 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 trained and run independently, it may result in inconsistent predictions. Considering the preprocessed word sequence "请问坐$[bus]$从$[location]_1$到$[location]_2$怎么走 (Please tell me how I can 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]. Fortunately, we can make use of the slot context features to relieve this problem. For example, the slot context feature "ShowRoute.[Route].[Origin] within the $\pm k$ windows" tends to imply ShowRoute.[Route].[Dest]. The lexical contexts only 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 slot classification module for re-classification. If the re-classification still results in clash, the dialog system will involve the user in an interactive dialog for clarity.

The idea of slot classification and re-classification can be understood as follows: it first finds the concept (or slot) islands (like partial parsing) and then bridge them together. This mechanism is well-suited for SLU since the spoken utterance usually consists of several phrases, between which are most often noises (restarts, repeats and filled pauses, etc) (Ward and Issar, 1994). Especially, this phenomena and the out-of-order structures are very frequent in the spoken Chinese utterances.

4. Weakly supervised training for two-stage classification based SLU

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

4.1. Related works about weakly supervised learning

Since it is always difficult to acquire a large amount of labeled training data, weakly supervised learning methods has been recently active in many areas. The aim of weakly supervised learning is to minimize the need for labeled examples while still achieving comparable or even better performance.

4.1.1. Active learning

One way to reduce the number of labeled examples is active learning, which has been applied in many domains (McCallum and Nigam, 1998; Tang et al., 2002; Tur et al., 2005). Traditionally, 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. In other words, this strategy always tries to select the examples, which will have the largest performance improvement, and hence minimizes the human labeling efforts 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. Herein we employ the uncertainty-based strategy for selective sampling. It is assumed that a small amount of labeled examples are initially available, which is used to train a base classifier. Then the classifier is applied to the unannotated examples. Typically the least confident 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.

4.1.2. Semi-supervised learning

Another alternative for reducing human labeling efforts is semi-supervised learning, which makes use of both labeled and unlabeled data for training. The acquisition of labeled data often requires expensive human labeling work, whereas the acquisition of unlabeled data is relatively easy. Therefore, it is a good idea to exploit unlabeled data to improve the performance of supervised learning algorithm and hence reduce human labeling efforts, which is the philosophy of semi-supervised learning.

Self-training is a commonly used semi-supervised learning approach. 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 and re-train the classifier. This procedure repeats for several iterations until the system performance converges. Self-training requires only one classifier without split of features.

Co-training (Blum and Mitchell, 1998) is another semi-supervised approach which attempts to improve the classification performance by exploiting large amount of unlabeled data. It assumes that features can be naturally split into two disjoint feature subsets (views): each feature subset is sufficient to train a good classifier and the two subsets are conditionally independent. Initially, two separate classifiers based on the two feature subsets are trained using a small amount of labeled data respectively. Then, each classifier tags the unlabeled data and adds new labeled examples that they feel most confident to the training set of the other classifier. Each classifier is retrained on the new training sets. This process repeats for several rounds. Co-training makes strong assumptions on the splitting of features, which is hard to hold in many real-world applications. Abney (2002) showed that the independence assumption could be relaxed and proposed a greedy algorithm to maximize agreement on unlabeled data. Nigam and Ghani (2000) investigated the effectiveness of co-training and proposed a semi-supervised learning paradigm named co-EM, which probabilistically tagged the entire unlabeled set instead of selecting a few most confident data. They also showed that co-training with artificial feature split (randomly divide the feature set into two subsets) still helped, though not as much as before.

Self-training and co-training are closely related. Their training processes run in a bootstrapping way: start with a set of labeled data and integrate examples from the unlabeled set into the labeled set iteratively. Therefore, in many literature (especially in the area of natural language processing), self-training and co-training are often referred as bootstrapping methods. The general bootstrapping process is as follows[3]:

(1) Given a small amount of labeled training set S_l (n examples) and a larger amount of unlabeled set S_u, and train the initial classifier C_i using S_l.
(2) Create a pool S_p of examples by randomly choosing n examples from S_u.
(3) While examples are available:
    (a) Use S_l to individually train the classifiers C_i, and label the examples in S_p.
    (b) For each classifier C_i, select m most confident examples and add them to S_l.
    (c) Refill S_p with the n random examples from S_u.

For co-training, the algorithm requires two classifiers with different views C_1 and C_2 that interact in the bootstrapping process. If there is only one classifier C_1 (with only one view), the bootstrapping process corresponds to self-training, in which a classifier learns from its own output. The underlying idea behind bootstrapping is to exploit the redundancy in the unlabeled data (Collins and Singer, 1999). The natural language processing tasks are suitable to be dealt with by the bootstrapping methods since natural language is highly redundant (Yarowsky, 1995). The bootstrapping methods have been applied to many natural language processing tasks such as word sense disambiguation (Yarowsky, 1995), web classification (Blum and Mitchell, 1998), named entity classification (Collins and Singer, 1999), statistical natural language parsing (Sarkar, 2001) and part-of-speech tagging (Clark et al., 2003).

4.1.3. Combining active learning and semi-supervised learning

Both active learning and semi-supervised learning aim at reducing the human labour. These two strategies are complementary in the sense that active learning selects the least confident examples for human labeling and semi-supervised learning makes use of the most confident machine-labeled examples. Therefore, it is quite natural to combine active learning and semi-supervised learning.

McCallum and Nigam (1998) combined the committee-based active learning with the Expectation Maximization (EM) learning to exploit the unlabeled examples in the task of text categorization. Muslea et al. (2002) proposed an effective combination strategy called CO-EMT, which introduces multiple views for both active learning and semi-supervised learning. In the literature of SLU, Tur et al. (2005) combined active learning and semi-supervised learning for call routing and demonstrated the effectiveness of reducing labeled examples.

[3] Excerpted from Mihalcea (2004).

4.2. Weakly supervised training of the topic classifier and topic-dependent slot classifiers

The weakly supervised training of the two kinds of classifiers is also 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 weakly supervised training (the combination of active learning and self-training). The resulting topic classifier is used to tag the remaining unlabeled training sentences with the corresponding topics. Then, we exploit all the sentences annotated against the semantic frame and the remaining training sentences machine-labeled only the topics to train the slot classifiers using weakly supervised training paradigms.

4.2.1. Weakly supervised training of the topic classifier

For weakly supervised training of the topic classifier, we employ the strategy of combining uncertainty-based active learning and self-training (Tur et al., 2005). The combination method is quite straightforward for pool-based training. The pool-based algorithm of combining active learning and self-training for the topic classifier is as follows:

(1) Given a small amount of human-labeled training set S_l and a larger amount of unlabeled set S_u.
(2) Create a pool S_p of sentences by randomly choosing n sentences from S_u.
(3) While labelers/sentences are available
  (a) Use S_l to train the topic classifier and label the sentences in S_p.
  (b) Select m sentences which are the least confident to the current classifier and manually label the selected sentences.
  (c) Add the m human-labeled sentences and the remaining machine-labeled sentences in S_p to S_l.
  (d) Refill S_p with the n random sentences from S_u.

At each iteration, the current classifier is applied to the unannotated examples in the pool S_p. The least confident 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, the two portions of labeled examples are added into the training set and used to re-train the classifier. A practical problem for this algorithm is how to compute the confidence score for the examples. For the topic classifier, we directly use the maximum class probability as the confidence score, which can be directly provided by the LIBSVM toolkit. Given a sentence s_i, its confidence score for topic classification is as follows:

$$Topic\_CS(s_i) = \max_j Pr(t_j|s_i) \quad (4)$$

where t_j is the jth topic.

4.2.2. Weakly supervised training of topic-dependent slot classifiers

Herein bootstrapping is also potential since the slot classification problem exhibits the redundancy. For instance, in the example “请问坐[*bus*]从[*location*]$_1$ 到[*location*]$_2$怎么走 (Please tell me how I can go from [*location*]$_1$ to [*location*]$_2$ by [bus])”, there are multiple lexical context features which all indicate that [*location*]$_1$ is of type ShowRoute.[Route].[Origin], such as:

(1) “从(from)” within the −1 windows;
(2) “从(from)” ___ “到(to)”;
(3) “到(to)” within the +1 windows.

Moreover, if the [*location*]$_2$ has already been recognized as ShowRoute.[Route].[Dest], the slot context feature “ShowRoute.[Route].[Dest] within the ±2 windows” is also a strong evidence that [*location*]$_1$ is of type ShowRoute.[Route].[Origin]. In other words, the lexical context and slot context features effectively overdetermine the slot of a concept in the input sentence. Especially, the lexical and slot context features can be seen as two natural “views” of an example from the respective of “co-training” (Blum and Mitchell, 1998). Given a few annotated seed sentences, our bootstrapping algorithm exploits the property of redundancy to incrementally identify the features for slot-filling.

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

(1) For each concept C_j in T_i ($1 \leqslant j \leqslant m$, m is the number of concepts appear in the training sentences with topic T_i), build the two initial classifiers based on the lexical and slot context features respectively using a small amount of labeled seed sentences.
(2) For each concept C_j in T_i, apply the current classifier based on the lexical context features to the remaining unlabeled concepts in the training sentences with the topic T_i. Keep those classified slots with confidence score above a threshold *th*.
(3) Check the consistency of the labeled slots in each training sentence. If some slots in a sentence clashed, keep the one with the highest confidence score among them and leave the others unlabeled.
(4) For each concept C_j in T_i, apply the current classifier based on the slot context features to the residual unlabeled concepts. Keep those classified slots with confidence score above a threshold *th* and Repeat Step 3.
(5) Augment the new classified cases into the training set and re-train the two classifiers based on the lexical and slot context features, respectively.
(6) If new slots are classified from the training data at the current iteration, return to Step 2. Otherwise, repeat Step 2–5 to label training data and keep all the new labeled slots regardless of the confidence score. Train the two final slot classifiers based on the lexical and slot context features, respectively using the final labeled training data.

This bootstrapping process makes use of both lexical and slot context features. If Steps 3 and 4 are canceled, it corresponds to the bootstrapping process of the slot classifier based on only the lexical context features. Note that our bootstrapping approach for slot classification is not co-training although it resembles co-training. Co-training could not be directly applied to the slot classification since initially the slot context features was not available. As an example, consider the preprocessed word sequence "Please tell me how I can go from $[location]_1$ to $[location]_2$ by [bus])", no slot context features can be extracted for slot classification. Therefore, in our bootstrapping approach, the slot classifiers based on the slot context features were only applied for slot re-classification to correct the slots, which are incorrectly tagged by the slot classifiers based on the lexical context features. Fig. 4 illustrates the bootstrapping process for weakly training slot classifiers, which can be seen as the visualization of the above bootstrapping algorithm. From this figure, we can see that our bootstrapping method for slot classification does not conform to the standard co-training paradigm and is not self-training only.

For this algorithm, a practical problem is how to set the value of the threshold *th* in Steps 3 and 5. A simple strategy is that *th* is set as a fixed optimal empirical value (often set as 0.5). However, it is observed that the cases added into the training set at the early iterations are unreliable since the performance of the early classifier is relatively poor. Motivated by this observation, at the early iterations, we can set the threshold as a relatively high value to prevent the noise data (incorrectly labeled examples) from being added into the training set, and then decrease the threshold as the performance of the classifier improves. This strategy is called bootstrapping with cautiousness. In the experiments reported in this paper, we set the initial threshold as 0.95 and made a reduction of 0.05 for each iteration until 0.5.

At the same time, intuitively the strategy of combining uncertainty-based active learning and self-training can also be applied to train the slot classifiers. Unfortunately, it is a little difficult to employ active learning for training slot classifiers. A sentence often contains multiple concepts, which correspond to different slot classifiers. They are trained independently. For the ordinary active learning for slot classifiers, the concepts in a sentence are asynchronously human-labeled since each concept may be selected by active learning of the corresponding slot classifier. As an example, consider the preprocessed word sequence "请问坐$[bus]$从$[location]_1$到$[location]_2$怎么走 (Please tell me how I can go from $[location]_1$ to $[location]_2$ by [bus])". This sentence may be selected for human to label for at most three times since there are totally three concepts in this sentence, which will make the labeling job more tiresome, time-consuming and difficult to keep consistency. The human labeler prefers to label the sentences against the semantic frame, i.e., label all the

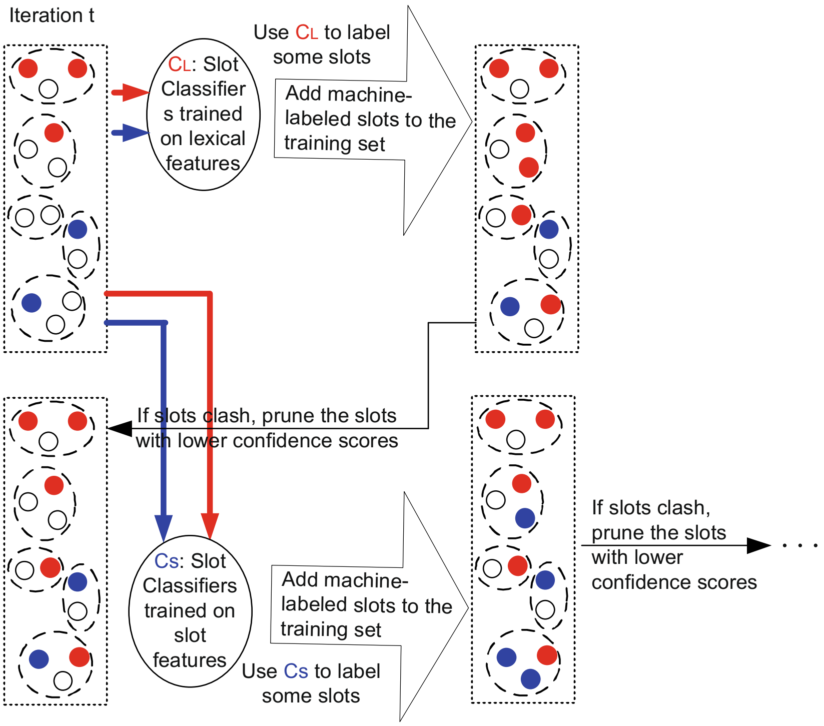
Iteration t
CL: Slot Classifiers trained on lexical features
Use CL to label some slots
Add machine-labeled slots to the training set
If slots clash, prune the slots with lower confidence scores
Cs: Slot Classifiers trained on slot features
Add machine-labeled slots to the training set
Use Cs to label some slots
If slots clash, prune the slots with lower confidence scores
. . .

Fig. 4. The bootstrapping process for weakly training slot classifiers. The dashed eclipses stand for the sentences containing several slots, which are denoted by the small inner circles. The red and blue small circles represent the slots labeled by the slot classifiers based on lexical features and slot features, respectively. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

concepts in a sentences at a time instead of labeling them asynchronously. To cater for this requirement, the pool-based algorithm of combining active learning and self-training in Section 4.2.1 needs to be modified as follows:

(1) Given a small amount of totally labeled training set S_l (labeled against the semantic frame) and a larger amount of partially unlabeled set S_u (labeled only the topic).
(2) Create a pool S_p of sentences by randomly choosing n sentences from S_u.
(3) While labelers/sentences are available.
  (a) Use S_l to train all the topic-dependent slot classifiers and label the sentences in S_p.
  (b) Compute the confidence score for each sentence in Sp and sort the sentences in the order of increasing confidence score.
  (c) Select m least confident sentences and manually label them against the semantic frame (if the topic is misclassified, correct it).
  (d) Add m human-labeled sentences and the remaining machine-labeled sentences in S_p to S_l.
  (e) Refill S_p with n random sentences from S_u.

Note that the algorithm transforms to pool-based active learning if only the human-labeled sentences are added into the training set in Step (d). Otherwise, if only the machine-labeled sentences are added, it is equal to pool-based simple self-training.

In this algorithm, the key issue is to measure the informativeness of a sentence with respect to all the related slot classifiers. One strategy is to use the minimum one of the confidence scores of all slots in a sentence s_i, which is illustrated by the equation as follows:

$$Slot\_CS(s_i) = \min_j Slot\_CS(ss_i^j) \tag{5}$$

where ss_i^j is a slot occurring in the sentence s_i, $Slot\_CS(ss_i^j)$ is determined by the strongest related feature $f_{max}(ss_i^j)$, i.e., $Slot\_CS(ss_i^j) = confidence(f_{max}(ss_i^j)$. Another strategy is to average the confidence scores of all slots in a sentence s_i as the following equation:

$$Slot\_CS(s_i) = \frac{\sum_j Slot\_CS(ss_i^j)}{n_{ss}^i} \tag{6}$$

where n_{ss}^i is the number of slots appearing in the sentence s_i.

5. Experiments and results

5.1. Data collection and experimental setting

Our experiments were carried out in two corpora. One is a Chinese corpus in the context of public transportation information inquiry domain. The other is the English DARPA Communicator Travel Data (Communicator Travel Data, 2004), which is related to air travel, hotel reservation, car rental, etc.

We collected two kinds of corpora for the Chinese transportation information inquiry domain in different ways. Firstly, a natural language corpus was collected through a specific website which simulated a dialog system. The user can conduct mixed-initiative conversational dialogues with the system by typing Chinese queries. We collected 2286 context-independent natural language utterances through this way. The context-independent utterances refer to the ones whose interpretation is independent of the dialog context (Meng and Siu, 2002). It was divided into two parts: the training set containing 1800 sentences (CTR), and the test set containing 486 sentences (CTS1). Also, a spoken language corpus was collected through the deployment of a preliminary version of telephone-based spoken dialogue system. In this system, the speech recognizer was based on the speaker-independent Chinese dictation system of IBM ViaVoice Telephony and the SLU component was a robust rule-based parser. The spoken utterances corpus contained 363 context-independent spoken utterances. Then we obtained two test sets from this corpus: one consists of the recognized text (CTS2); the other consists of the corresponding transcription (CTS3). Due to the unique challenges of Chinese speech recognition (e.g., 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 the concept error rate on CTS2 are 35.6% and 41.1%, respectively. We defined ten types of topic for the Chinese domain: ExplainLoc, IsServedLoc, IsServedTime, ListStop, ShowFare, ShowLocService, ShowRoute, ShowRouteTime, ShowStopPos. The first corpus covered all the ten topic types. The second corpus covered only four topic types: IsServedLoc, ListStop, ShowRoute, ShowRouteTime. Among the ten topics, ShowRoute occurred 71.1% of the time in the first corpus and 78.5% of the time in the second corpus. All the sentences were annotated against the semantic frame. In our experiments, the topic classifier and slot classifiers were trained on the natural language training set (CTR) and tested on three test sets (CTS1, CTS2 and CTS3). The vocabulary size (i.e., the total number of Chinese characters appear in unpreprocessed CTR set) was 923. After the preprocessing, the vocabulary size decreased to 397.

The DARPA Communicator Travel Data were collected in 461 days by University of Colorado (Communicator Travel Data, 2004), which are available to the public as open source download. This set contains 38408 utterance transcriptions and the corresponding semantic parsing results from the Phoenix parser (CU Pheonix Parser, 2003). We randomly selected the sentences in 101 days as the testing set and the reminder as the training set. We deleted the confirmed utterances such as “Yes.” and “No, thank you.”, the utterances with the parsing result of “No parse” and the dialog context dependent utterances such as the ones containing only a city name. After these cleaning steps, the training set (ETR) and the test set (ETS) contained 3724 and 1279 context-independent utterances respectively. We defined five types of topic for the English domain: RentCar, ReqFlight, ReqRoundFlight, ReqHotel, ReqReturnFlight. Among the five topics, ReqFlight occurs 67.4% of the time in the English corpus. The vocabulary size (i.e., the total number of English words appear in the unpreprocessed ETR set) was 839. After preprocessing, the vocabulary size decreased to 331. The

Table 2
The characteristics of utterance sets in the Chinese and English domains.

	The Chinese domain				The English domain	
	CTR	CTS1	CTS2	CTS3	ETR	ETS
#Sentences	1800	486	363	363	3724	1279
#Topic types	10	10	4	4	5	5
#Concept types	26	24	10	10	17	16
#Concepts	4999	1378	786	786	7132	2456
#Slot types	110	76	22	22	52	45
#Slots	4866	1358	767	767	7132	2456
%Character error rate	0.7	0.6	35.6	–	–	–
%Concept error rate	0.6	0.5	41.1	–	–	–

semantic frames of all the utterances were automatically extracted from the parsing results and hand corrected. Table 2 describes the characteristics of different utterance sets in the Chinese and English domains.

The performance of topic classification and slot classification are measured in terms of topic error rate and slot error rate respectively. The topic error rate is measured by comparing the topic of a sentence predicated by the topic classifier with the reference topic (i.e., the labeled frame type). The slot error rate is measured by counting the insertion, deletion and substitution errors between the slots generated by our system and those in the reference annotation. Note that, in a semantic frame, the topic error would cause all the corresponding slots to be incorrect. In addition, in order to validate to which extent the claim of an improvement can be supported by the observations on the test sets, we carried the statistical significance test (sign test) in most comparative experiments. We followed the common statistical practice that the significance p-values of 0.05 and 0.01 are required for claiming an improvement to be significant and very significant, respectively.

5.2. Supervised training experiments

5.2.1. The preprocessing performance

Firstly, it is necessary to evaluate the performance of the preprocessor since it is the basis of the whole system. We measure the preprocessing performance in term of the concept error rate, which is achieved by counting the insertion, deletion and substitution errors between the concepts generated by the preprocessor and those in the reference annotation. Table 3 shows the concept error rates of the preprocessor on the four test sets.[4] In the Chinese domain, we defined 75 rules for the local chart parser, some of which was potential to be reused in other application domains, for example, the rules for time expressions. Because of the difficulties of the Chinese speech recognition in the first domain as described before, the concept error rate of speech recognition on CTS2 was 41.1%.[5] Accordingly, the preprocessing performance on CTS2 was relatively poor. It is possible that the insertion concept errors resulting from the preprocessor are corrected by the latter component. For example, the slot classification may assign the insertion concepts with the dummy slots.

5.2.2. Supervised experiments for topic classification

For the topic classification, we compared the performance of the SVMs using various feature sets. The simplest features are all Chinese characters (or English words) in the training set. As mentioned in Section 3.2.2, we can also include semantic class labels as features. Using the preprocessor, we substituted those Chinese characters (or English words) with the corresponding semantic class labels. In addition, we also considered to use the N-grams (the Chinese character N-grams or English word N-grams) as features. The N-grams (up to length 3) were automatically acquired through high-frequency statistics and simple counting pruning. Tables 4 and 5 list the results of topic classification using the various kinds of features in the Chinese and English domains, respectively. The results show that the deep level feature (both N-gram and semantic class label) can improve the performance of topic classification. Note that, since the CTS2 (recognized utterance) covers

[4] For the ETS set, we directly extracted the preprocessed results from the parsing results of Phoenix Parser.

[5] Note that the concepts tagged by the preprocessor are on the higher level than those identified by the speech recognizer.

Table 3
The preprocessing performance.

CTS1 (%)	CTS2 (%)	CTS3 (%)	ETS (%)
2.5	43.2	1.3	2.0

Table 4
The topic error rates of the SVMs using various kinds of features in the Chinese domain.

Features	Topic error rate		
	CTS1 (%)	CTS2 (%)	CTS3 (%)
Chinese character	4.7	3.6	3.0
Chinese character N-gram	4.3	3.0	2.8
Chinese character and semantic class	2.9	2.2	1.4

Table 5
The topic error rates of the SVMs using various kinds of features in the English domain.

Feature	Topic error rate (%)
English word	3.5
English word N-gram	2.8
English word and semantic class	2.0

only four types of topic but CTS1 (typed utterance) covers ten topics, the topic error rate on CTS2 is lower than that on CTS1. We have also checked the statistical significance (sign test) of deep level features (semantic class labels) versus naive features (Chinese character or English word). The statistically significant improvements were obtained on all four test sets: CTS1(p-value = 0.01), CTS2(p-value = 0.041), CTS3(p-value = 0.035) and ETS(p-value = 0.0003).

5.2.3. Supervised experiments for slot classification

In order to investigate the impact of the slot re-classification, we first compared the performances of slot classification using only lexical context features and using both lexical and slot context features. Through feature extraction from the CTR set and feature pruning, we obtained 2259 lexical context features and 369 slot context features for 20 types of concepts in the Chinese domain. Similarly, we extracted 2361 lexical context features and 378 slot context features for 17 types of concepts in the English domain. Herein, the slot error rates are based on the identified topics by the best SVMs. If the topic of a sentence is incorrectly identified, each slot in this sentence is treated as a substitution error. Table 6 shows that slot re-classification considerably improves the performance. Due to the high concept error rate on recognized utterances, the performance of slot classification on the CTS2 is relatively poor. However, if considering only the correctly recognized concepts on CTS2, the slot error rate is 9.2%. In the slot classification experiments, the optimal value of β in Eq. (2) was chosen through the 10-fold cross-validation on the training set, which was set as 0.01 and 0.012, respectively in the Chinese and English domains. Also, we carried out the statistical significance test (sign test) of two-pass versus one-pass slot classification. The statistically significant improvements using the slot

Table 6
The slot error rates of decision lists.

	CTS1 (%)	CTS2 (%)	CTS3 (%)	ETS (%)
One-pass decision list	9.1	46.7	5.0	7.0
Two-pass decision list (+ re-classification)	8.4	45.6	4.5	6.6

Table 7
Performance comparison 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).

	CTS1		CTS2		CTS3		ETS	
	TER (%)	SER (%)	TER (%)	SER (%)	TER (%)	SER (%)	TER (%)	SER (%)
Rule-based parser	6.8	11.6	4.1	47.9	3.0	5.4	–	–
DL + SVM	4.9	11.1	3.6	47.4	2.5	4.9	2.6	8.4
SVM + DL	2.9	8.4	2.2	45.6	1.4	4.6	2.0	6.6

Table 8
The significance test results of performance comparison 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). “$\gg$” means p-value $\leqslant 0.01$; “$>$” means $0.01 < p$-value $\leqslant 0.05$; “$\sim$” means p-value >0.05.

	CTS1		CTS2		CTS3		ETS	
	TER	SER	TER	SER	TER	SER	TER	SER
(DL + SVM) vs. Rule-based parser	≫	>	∼	∼	∼	∼	–	–
(SVM + DL) vs. Rule-based parser	≫	≫	>	≫	>	>	–	–
(SVM + DL) vs. (DL + SVM)	≫	≫	>	≫	∼	∼	>	≫

re-classification were obtained on the CTS1 set (p-value = 0.002), the CTS2 set (p-value = 0.008) and the ETS set (p-value = 0.003). The improvement achieved on the CTS3 set was not statistically significant (p-value = 0.101).

We also compared our system with a rule-based robust semantic parser in the Chinese domain. The parsing algorithm of this parser is the same as that of the local chart parser used by the preprocessor. A linguistic expert spent one month to handcraft and tune the semantic grammar for this parser, which consists of 798 rules (except the lexical rules for named entities such as [loc_name]). In our SLU system, we first used the SVMs to identify the topic and then applied the decision list related to the identified topic to assign the slots to the concepts. The SVMs used the augmented binary features: the Chinese characters and semantic class labels in the Chinese domain, and the English words and semantic class labels in the English domain. A general developer independently annotated the CTR set against the semantic frame, which took only four days. The semantic frames of all the English utterances were automatically extracted from the parsing results and hand corrected. Table 7 shows that our SLU method (SVM + DL) performs better than the robust rule-based parser in both topic classification and slot identification. The results of statistical significance (sign test) of our SLU system (SVM + DL) versus rule-based parser are also shown in the Table 8.

Another alternative for our SLU 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. 2, the slot (e.g., [Route].[Origin]) of each concept (e.g., [location]) in the preprocessed sequence is recognized before topic classification. Therefore, the slots like [Route].[Origin] can also be included as features for topic classification, which is deeper than the concepts like [location] and potential to achieve better topic classification performance. This strategy was adopted in some previous works (He and Young, 2003; Wutiwiwatchai and Furui, 2003). Table 7 also compares our SLU systems with different processing orders (i.e., SVM + DL or DL + SVM). The statistical significance (sign test) of our SLU systems with different processing orders versus rule-based parser have been checked. The sign test results are showed in Table 8. These results indicate that, at least in our two-stage classification framework, the strategy of identifying the topic before assigning the slots to the concepts is more optimal. According to our error analysis, the unsatisfactory performance of the reversed two-stage classification system can be explained as follows: (1) since the slot classification is performed on all topics, the search space is much bigger and the ambiguities increase. These deteriorate the performance of slot classification. (2) In the case that the slots and Chinese characters (or English words) are both included as features, the topic classifier relies heavily on the slot features. Then, the errors of slot classification have serious negative effect on the topic classification.

5.3. Weakly supervised training experiments

5.3.1. Weakly supervised training experiments for topic classification

First, we evaluated the performance of active learning on the CTS1 and ETS sets. Herein active learning is pool-based. The pool sizes of the Chinese and English domains were set as 200 and 300, respectively, i.e., 200/300 sentences are randomly selected from the unlabeled set at each iteration. Active learning chose 50/75 least confident sentences from the pool for manually labeling at each iteration. All the experiments were repeated ten times with different randomly selected seed sentences and the results were averaged. Fig. 5 depicts the learning curves of active learning on the CTS1 and ETS sets. 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 CTS1, but only 600 actively selected examples (a saving of 62.5%).

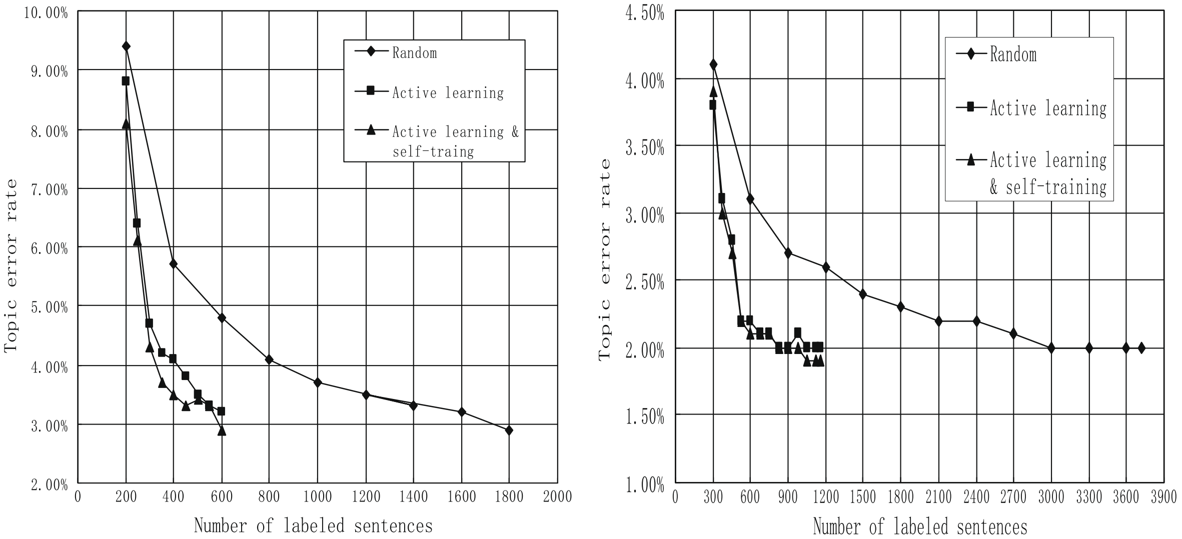

Fig. 5. Learning curves using different sampling strategies in the CTS1 (left) and ETS (right) set. In both figures, the topmost baseline curves are obtained using random sampling on the training sets. The downmost curves are achieved using active learning and self-training.

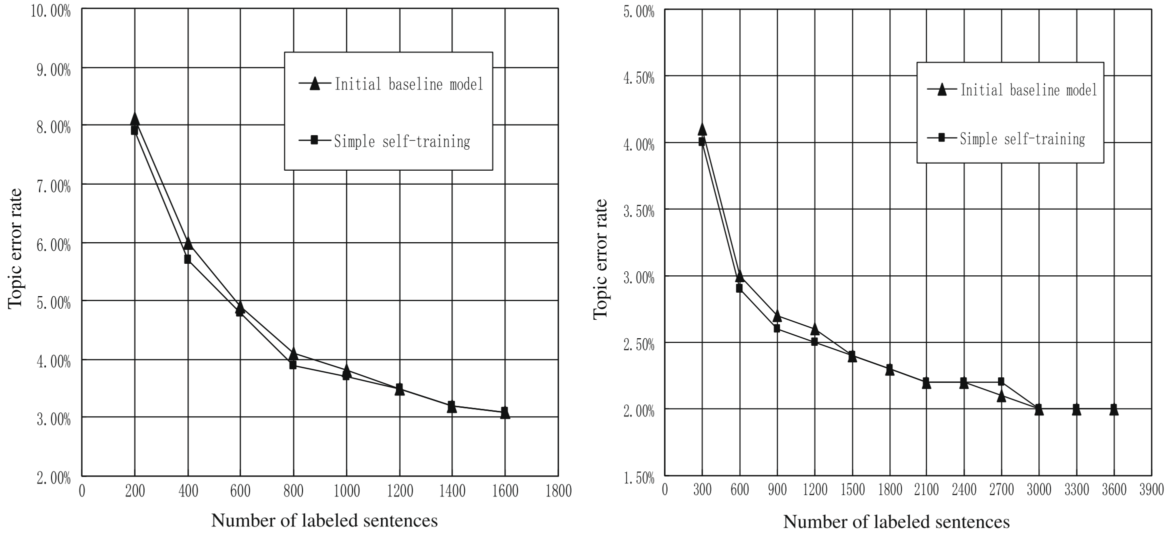

Fig. 6. Learning curves using the simple self-training strategy in the CTS1 (left) and ETS (right) set. In both figures, the topmost baseline curves are obtained using supervised training. The curves below are achieved using simple self-training.

Table 9
The topic error rate using pool-based active learning and self-training on the four test sets (AL: active learning).

	The Chinese domain				The English domain	
	CTS1 (%)	CTS2 (%)	CTS3 (%)	Labeled Sent.	ETS (%)	Labeled Sent.
Random	2.9	2.2	1.4	1800	2.0	3000
AL	3.2	2.5	1.7	600	2.0	1050
AL and self-training	2.9	2.5	1.4	600	1.9	1050

We also evaluated self-training methods on the same corpus. Fig. 6 gives the learning curves of simple self-training on the CTS1 and ETS sets. It shows that simple self-training can achieve slight performance improvement when the number of manually labeled sentences is less than 1200 and 1500, respectively in the Chinese and English domains. However, no improvement is gained when the amount of manually labeled data exceeds a certain threshold. This is coincident with the intuition: when the initial model is good enough, the examples automatically labeled by the classifier is less useful for the performance improvement.

Finally, the strategy of combining active learning and self-training is evaluated on the same corpus. For the combination of pool-based active learning and self-training, at each iteration, one fourth of least confident sentences were selected from the pool for manually labeling and the remaining sentences in the pool were automatically labeled by the current classifier. Fig. 5 also plots the learning curves of active learning and self-training on the CTS1 and ETS sets (the downmost curves). It shows that, given the same amount of labeled data, the strategy of combing active learning and self-training can further improve the performance of topic classification than active learning only.

Moreover, we evaluated the overall performance of topic classification using pool-based active learning and self-training on the four test sets. Table 9 shows that the combination of active learning and self-training achieves almost the same performance on four test sets as random sampling does, but requires only about one third labeled data.

5.3.2. Weakly supervised training experiments for topic-dependent slot classification

We first evaluated the bootstrapping method for training the topic-dependent slot classifiers, which is based on the topic classifier trained through the combination of active learning and self-training. In this scenario, the sentences selected by active learning of the topic classifier were manually labeled the semantic frames. Thus, the bootstrapping procedure of the slot classifiers began with a small amount of sentences annotated against the semantic frame, which were either the initial seed sentences or human-labeled by active learning of the topic classifier. The topics of the remaining training sentences were machine-labeled by the resulting topic classifier. In this weakly supervised training scenario, the pool sizes in the Chinese and English domains were set as 200 and 300, respectively. Therefore, the active learning and self-training procedure for topic classification need to run 8/12 iterations. At each iteration, one fourth of least confident sentences were selected by active learning of the topic classifier. The number of initial seed sentences was also 200/300. So the total number of sentences human-labeled the semantic frames was 600/1156.

We compared our bootstrapping methods with supervised training for slot classification. We tested two bootstrapping methods: using only lexical context features, and using both lexical and slot context features. We also investigated the effect of the modification of cautiousness on the threshold setting. All the experiments ran ten times with different labeled sentences and the results were averaged. Figs. 7 and 8 plot the learning curves of various bootstrapping methods and supervised training with different number of labeled sentences on the CTS1 and ETS sets. The results indicate that bootstrapping methods can effectively make use of the unlabeled data to improve the slot classification performance. Bootstrapping with mixed features achieves considerable improvement over bootstrapping with only lexical features. 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 slot classifiers using only lexical context features or provides new cases. Moreover, the figures show the cautiousness strategy improves the bootstrapping performance in most of time, even slightly in the English domain. The cautiousness strategy could to a certain extent prevent the machine-labeled noise data from being added into the training set.

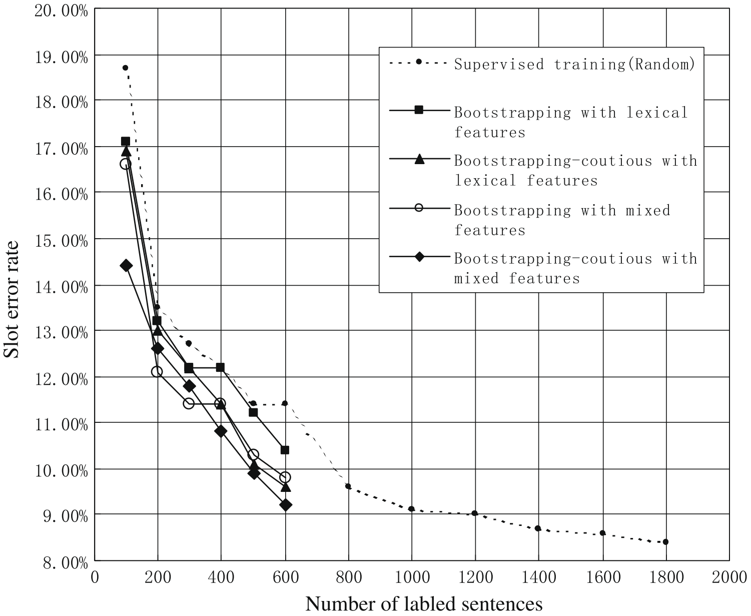

Fig. 7. Learning curves of various bootstrapping methods for slot classification on the CTS1 set.

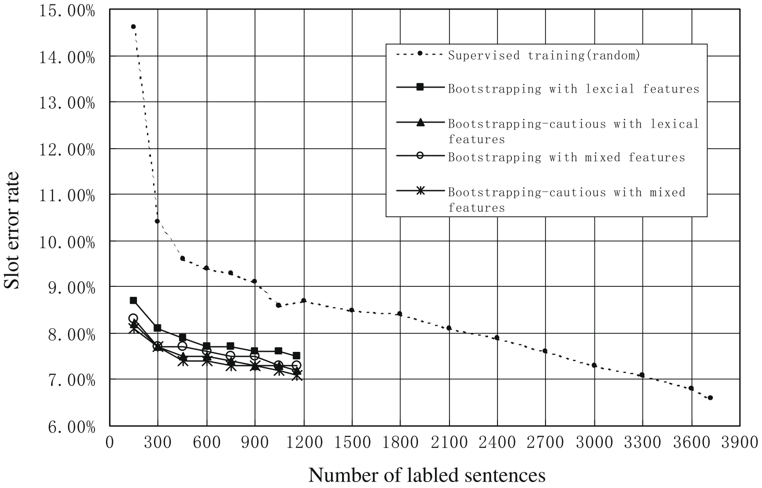

Fig. 8. Learning curves of various bootstrapping methods for slot classification on the ETS set.

We also evaluated the strategy of combining active learning and self-training for slot classification, prior to which the topic classifier has been already trained through the combination of active learning and self-training. In this paradigm, the sentences selected by active learning for topic classification and slot classification are manually labeled only the topics and the whole semantic frames, respectively. Thus, the combination of active learning and self-training of the slot classifiers begins with a small amount of the initial seed sentences annotated against the semantic frame and the remaining training sentences whose topics were human-labeled by active learning for topic classification or machine-labeled by the resulting topic classifier. At each iteration, active learning for slot classification select one fourth of least confident sentences for human to label. However, only one fourth of most confident sentences are automatically labeled in order to avoid the incorrectly labeled slots to some extent. Similarly, the pool sizes in the Chinese and English domains were both set as 200 and 300, respectively. The number of seed sentences is the same as the pool size in both domains. Therefore, besides 600/1156 sentences human-labeled the topics, the same amount of sentences were manually labeled the semantic frames.

We compared the strategies of supervised training, active learning only, combining active learning and self-training for slot classification. For active learning of slot classification, we evaluated two confidence metrics: one is the minimum confidence score of all slots in a sentence (AL-minscore), the other is the mean value of confidence scores of all slots in a sentence (AL-meanscore). Figs. 9 and 10 depict the learning curves of these weakly supervised strategies and supervised training with different number of labeled sentences on the CTS1 and ETS sets. It shows that active learning only, combining active learning and self-training can both effectively reduce the number of labeled sentences. AL-minscore demonstrates better performance than AL-meanscore in most of the time. This is consistent with the intuition. When AL-minscore selects a sentence for human labeling, this sentence contains at least one slot with the lowest confidence score. On the contrary, when AL-meanscore selects a sentence, the slots in this sentence may have medium confidence scores although

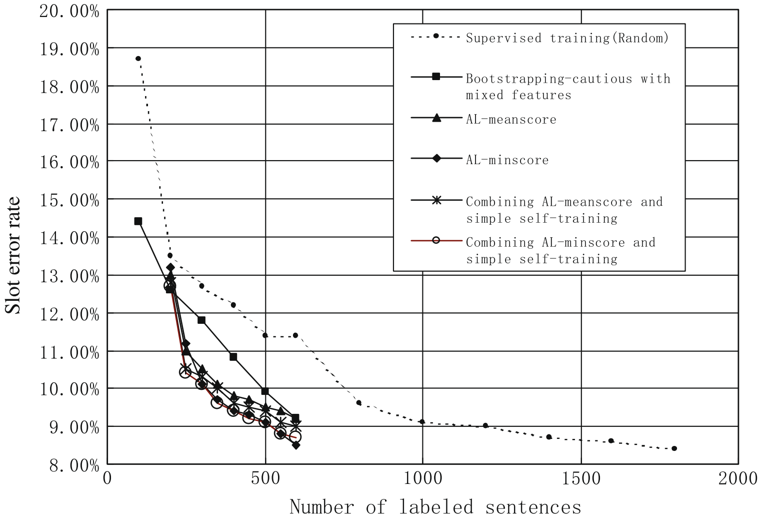

Fig. 9. Learning curves of various weakly supervised training strategies and supervised training for slot classification on the CTS1 set.

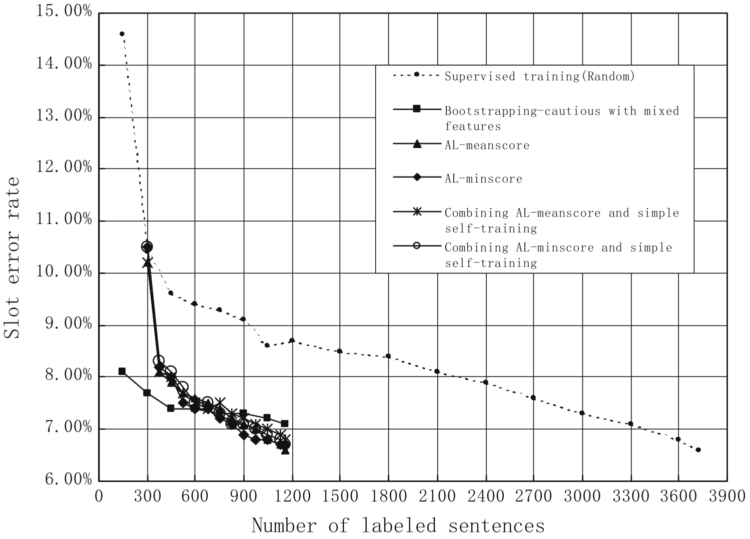

Fig. 10. Learning curves of various weakly supervised training strategies and supervised training for slot classification on the ETS set.

Table 10
Performance comparison of three SLU systems using supervised training and weakly supervised training on the four test sets (TER: topic error rate; SER: slot error rate).

	The Chinese domain							The English domain		
	CTS1 (%)		CTS2 (%)		CTS3 (%)		Labeled sent.	ETS (%)		Labeled sent.
	TER	SER	TER	SER	TER	SER		TER	SER	
Supervised	2.9	8.4	2.2	45.6	1.4	4.6	1800	2.0	6.6	3724
Weakly supervised 1	3.2	9.2	2.5	45.4	1.7	5.7	600	2.0	7.1	1156
Weakly supervised 2	3.2	8.5	2.5	45.5	1.7	5.1	600 (600)	2.0	6.7	1156 (1156)

Table 11
The significance test results of performance comparison of the SLU systems using supervised training and weakly supervised training on the four test sets (TER: topic error rate; SER: slot error rate). “$\gg$” means p-value $\leqslant 0.01$; “$>$” means $0.01 < p$-value $\leqslant 0.05$; “$\sim$” means p-value > 0.05.

	The Chinese domain						The English domain	
	CTS1		CTS2		CTS3		ETS	
	TER	SER	TER	SER	TER	SER	TER	SER
Supervised vs. weakly supervised 1	∼	>	∼	∼	∼	>	∼	>
Supervised vs. weakly supervised 2	∼	∼	∼	∼	∼	∼	∼	∼

their mean score is low. However, the combination of active learning and simple self-training did not achieve improvement over active learning only. This is due to that different slot classifiers share the same confidence threshold and hence the machine-labeled noise examples are possibly introduced. Figs. 9 and 10 also show active learning only or combining active learning and self-training achieves better performance than the best bootstrapping method at the expense of more labeled job, i.e., manually label the topics of the sentences selected by active learning of the topic classifier.

Finally, we compared two SLU systems through supervised training and weakly supervised training respectively in both application domains. The supervised systems were trained using all the annotated sentences in the CTR (1800 sentences) and ETR (3724 sentences) sets. In the first weakly supervised training scenario (weakly supervised 1), the topic classifier was trained using the pool-based active learning and self-training, and the slot classifiers were trained through the cautious bootstrapping methods with the lexical and slot context features. Since the pool sizes in the Chinese and English domains were set as 200 and 300, the weakly supervised training of the topic and slot classifiers used only 600 and 1156 labeled sentences respectively. In the second weakly supervised training scenario (weakly supervised 2), the topic classifier was also trained using the pool-based active learning and self-training, and the slot classifiers were trained using pool-based active learning (AL-minscore). Similarly, with the pool sizes of 200 and 300 in the Chinese and English domains respectively, the weakly supervised training of topic classifier required 600/1156 sentences labeled the topics. The weakly supervised training of slot classifiers required 600/1156 sentences labeled the semantic frame. Table 10 shows that the weakly supervised scenarios achieve comparable performance with the supervised one,[6] but require only about 33.3% labeled data. Furthermore, the significance test results in Table 11 allow us to conclude as follows: (1) for topic classification, the performance difference between the topic classifiers using supervised training and the combination of active learning and self-training is not statistically significant; (2) for slot classification, the performance difference between the slot classifiers using supervised training and pool-based active learning (AL-minscore) is not statistically significant, however the difference between supervised training and bootstrapping is statistically significant (p-value > 0.05).

[6] The number in the braces in the third line is the amount of sentences which are selected by active learning of topic classification and manually labeled the topics.

6. Conclusion and future work

We have presented a new SLU framework based on two-staged classification. The proposed framework exhibits the advantages as follows.

- It has good robustness on processing spoken language: (1) the preprocessor provides 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 concept or slot islands and then linking them is suitable for processing spoken language.
- It also keeps the understanding deepness: (1) the class of slot classification is the slot name, which inherits the hierarchy from the domain model. (2) The slot 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. More importantly, our proposed SLU framework allows the employment of weakly supervised strategies for training the two kinds of classifiers, which can significantly reduce the cost of annotating labeled sentences.

The experimental results show that the performance of our SLU system based on two-stage classification is better than traditional robust rule-based parser and comparable to other new data-driven SLU systems. However, since it can be trained using the weakly supervised training methods, our system requires less labelled data and hence significantly reduce the development cost.

The future work includes employing high level knowledge, such as the dialog context, as the features of topic and slot classifiers. Moreover, like most current practices, the slot frames are also 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 semantic frame design.

References

Abney, S., 2002. Bootstrapping. In: Proceedings of ACL, Philadelphia, PA, pp. 360–367.

Allen, J. et al., 1995. The TRAINS project: a case study in building a conversational planning agent. Journal of Experimental and Theoretical Artificial Intelligence (7), 7–48.

Blomberg, M., Carlson, R., Elenius, K, Granstrom, B, Gustafson, J., Hunnicutt, S., Lindell, R., Neovius, L., 1993. An experimental dialogue system: WAXHOLM. In: Proceedings of EUROSPEECH.

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

Communicator Travel Data, 2004. University of Corlorado at Boulder, URL: <http://communicator.colorado.edu/phoenix>.

CU Pheonix Parser, 2003. University of Colorado at Boulder, URL: <http://communicator.colorado.edu/phoenix>.

Carpenter, B., Chu-Carroll, J., 1998. Natural language call routing: a robust, self-organizing approach. In: Proceedings of ICSLP.

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

Clark, S., Curran, J., Osborne, M., 2003. Bootstrapping POS-taggers using unlabeled data. In: Proceedings of Computational Natural Language Learning (CoNLL-03), Edmonton, Canada.

Cohn, D., Atlas, L., Ladner, R., 1994. Improving generalization with active learning. Machine Learning 15, 201–221.

Collins, M., Singer, Y., 1999. Unsupervised models for named entity classification. In: Proceedings of EMNLP.

Dowding, J., Gawron, J., Appelt, D., Bear, J., Cherny, L., Moore, R., Moran, D., 1993. GEMINI: a natural language system for spoken language understanding. In: Proceedings of ACL, Columbus, Ohio, pp. 54–61.

Fu, K.S., Booth, T.R., 1975. Grammatical inference: introduction and survey, Parts I and II. IEEE Trans. Systems, Man, and Cybernetics, SMC-5(1) and (4), pp. 95–111 and pp. 409–423.

Gao, Y., Gu, L., Kuo, H., 2005. Portability challenges in developing interactive dialogue systems. In: Proceedings of ICASSP, pp. 1017–1020.

Golding, R., 1995. A Bayesian hybrid method for context-sensitive spelling correction. In: Proceedings of Third Workshop on Very Large Corpora, Boston, MA.

Gorin, A., Riccardi, G., Wright, J., 1997. How may I help you. Speech Communication 23, 113–127.

Gupta, N., Tur, G., Hakkani-Tür, D., Bangalore, S., Riccardi, G., Rahim, M., 2006. The AT&T spoken language understanding system. IEEE Transactions on Speech and Audio Processing. 4 (1), 213–222.

He, Y., Young, S., 2003. A data-driven spoken language understanding system. In: Proceedings of IEEE ASRU Workshop, US Virgin Islands.

He, Y., Young, S., 2005. Semantic processing using the hidden vector state model. Computer Speech and Language 19 (1), 85–106.

Kuhn, R., De Mori, R., 1995. The application of semantic classification trees to natural language understanding. IEEE Transaction on Pattern Analysis and Machine Intelligence 17, 449–460.

Lamel, L., Bennacef, S., Rosset, S., Devillers, L., Foukia, S., Gangolf, J., Gauvain, J., 1997. The LIMSI RailTel system: field trial of a telephone service for rail travel Information. Speech Communication 23 (1–2), 67–82.

Macherey, K., Och, F., Ney, H., 2001. Natural language understanding using statistical machine translation. In: Proceedings of EUROSPEECH.

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

Meng, H., Siu, K., 2002. Semiautomatic acquisition of semantic structures for understanding domain-specific natural language queries. IEEE Transactions on Knowledge and Data Engineering 14 (1), 172–181.

Mihalcea, R., 2004. Co-training and self-training for word sense disambiguation. In: Proceedings of the Conference on Computational Natural Language Learning (CoNLL-2004).

Miller, S., Bobrow, R., Ingria, R., Schwartz, R., 1994. Hidden understanding models of natural language. In: Proceedings of ACL, pp. 25-32.

Minker, W., Bennacef, S., Gauvain, J., 1996. A stochastic case frame approach for natural language understanding. In: Proceedings of ICSLP, Philadelphia, USA, pp. 1013–1016.

Muslea, I., Minton, S., Knoblock, C.A., 2002. Active + semi-supervised learning = robust multi-view learning. In: Proceedings International Conference on Machine Learning (ICML), Sydney, Australia.

Nigam, K., Ghani, R., 2000. Analyzing the effectiveness and applicability of co-training. In: Proceedings International Conference on Information and Knowledge Management (CIKM), McLean, VA.

Pargellis, A., Fosler-Lussier, E., Potamianos, A., Lee, C., 2001. Metrics for measuring domain independence of semantic classes. In: Proceedings of EUROSPEECH, Aalborg, Denmark.

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

Pietra, S., Epstein, M., Roukos, S., Ward, T., 1997. Fertility models for statistical natural language understanding. In: Proceedings of ACL, Madrid, Spain, pp. 168–173.

Price, P., 1990. Evaluation of spoken language systems: the ATIS domain, In: Proceedings of DARPA Speech and Natural Language Workshop, pp. 91–95.

Rivest, R., 1987. Learning decision lists. Machine Learning 2 (3), 229–246.

Rochery, M., Schapire, R., Rahim, M., Gupta, N., Riccardi, G., Bangalore, S., Alshawi, H., Douglas, S., 2002. Combining prior knowledge and boosting for call classification in spoken language dialogue. In: Proceedings of ICASSP, Orlando, USA.

Sarkar, A., 2001. Applying co-training methods to statistical parsing, In: Proceedings of NAACL, pp. 175–182.

Schohn, G., Cohn, D., 2000. Less is more: active learning with support vector machines. In: Proceedings of ICML, pp. 839–846.

Seneff, S., 1992. TINA: A natural language system for spoken language applications. Computational Linguistics 18 (1), 61–86.

Stolcke, A., Omohundro, S., 1994. Inducing probabilistic grammars by bayesian model merging. In: Proceedings of International Conference on Grammatical Inference and Applications, LNAI, vol. 862, Springer-Verlag, pp. 106–118.

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

Tong, S., Koller, D., 2000. Support vector machine active learning with applications to text classification. In: Proceedings of ICML, pp. 999–1006.

Tur, G., Hakkani-Tür, D., Schapire, R., 2005. Combining active and semi-supervised learning for spoken language understanding. Speech Communication 45 (2), 171–186.

Vapnik, V., 1998. Statistical Learning Theory. Wiley, New York.

Walker, M., Litman, D., Kamm C., Abella A., 1997. PARADISE: a framework for evaluating spoken dialogue agents. In: Proceedings of ACL.

Wang, Y., Acero, A., 2001. Grammar learning for spoken language understanding. In: Proceedings of IEEE ASRU Workshop, Madonna di Campiglio, Italy.

Wang, Y., Waibel, A., 1998. Modeling with structures in statistical machine translation. In: Proceedings of ACL-COLING, Montreal, Que, Canada.

Wang, Y., Acero, A., Chelba, C., Frey, B., Wong, L., 2002. Combination of statistical and rule-based approaches for spoken language understanding. In: Proceedings of ICSLP, Denver, Colorado.

Wang, Y., Deng, L., Acero, A., 2005. Spoken language understanding – an introduction to the statistical framework. IEEE Signal Processing Magazine 27 (5).

Wang, Y., Acero, A., Mahajan, M., Lee, J., 2006. Combining statistical and knowledge-based spoken language understanding in conditional models. In: Proceedings of ACL-COLING.

Wang, Y., 1999. A robust parser for spoken language understanding. In: Proceedings of EUROSPEECH, Budapest, Hungary.

Ward, W., Issar, S., 1994. Recent improvements in the CMU spoken language understanding system. In: Proceedings of ARPA Workshop on HLT.

Wu, W., Duan, J., Lu, R., Gao, F., 2005. Embedded machine learning systems for robust spoken language parsing. In: Proceedings of IEEE NLP-KE, Wuhan, China.

Wu, W., Lu, R., Liu, H., Gao, F., 2006a. A spoken language understanding approach using successive learners. In: Proceedings of ICSLP, Pittsburgh, PA, USA.

Wu, W., Lu, R., Duan, J., Liu, H., Gao, F., Chen, Y., 2006b. A weakly supervised learning approach for spoken language understanding. In: Proceedings of EMNLP, Sydney, Australia.

Wutiwiwatchai, C., Furui, S., 2003. Combination of finite state automata and neural network for spoken language understanding. In: Proceedings of EUROSPEECH, Geneva, Switzerland.

Yarowsky, D., 1994. Decision lists for lexical ambiguity resolution: application to accent restoration in Spanish and French. In: Proceedings of ACL, pp. 88–95.

Yarowsky, D., 1995. Unsupervised word sense disambiguation rivaling supervised methods. In: Proceedings of ACL, Cambridge, MA, pp.189–196.

Zue, V., Seneff, S., Glass, J., Polifroni, J., Pao, C., Hzen, T., Hetherington, L., 2000. JUPITER: a telephone-based conversational interface for weather information. IEEE Transaction on Speech Audio Processing 8 (1), 85–96.