Learning Generalized Features for Semantic Role Labeling

HAITONG YANG, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences
CHENGQING ZONG, National Laboratory of Pattern Recognition, Institute of Automation, CAS Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences

This article makes an effort to improve Semantic Role Labeling (SRL) through learning generalized features. The SRL task is usually treated as a supervised problem. Therefore, a huge set of features are crucial to the performance of SRL systems. But these features often lack generalization powers when predicting an unseen argument. This article proposes a simple approach to relieve the issue. A strong intuition is that arguments occurring in similar syntactic positions are likely to bear the same semantic role, and, analogously, arguments that are lexically similar are likely to represent the same semantic role. Therefore, it will be informative to SRL if syntactic or lexical similar arguments can activate the same feature. Inspired by this, we embed the information of lexicalization and syntax into a feature vector for each argument and then use K-means to make clustering for all feature vectors of training set. For an unseen argument to be predicted, it will belong to the same cluster as its similar arguments of training set. Therefore, the clusters can be thought of as a kind of generalized feature. We evaluate our method on several benchmarks. The experimental results show that our approach can significantly improve the SRL performance.

Categories and Subject Descriptors: G.4 [**Mathematics of Computing**]: Mathematical Software—*Algorithm design and analysis*; H.4.0 [**Information Systems Applications**]: General; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text analysis*

General Terms: Algorithms, Languages, Experimentation, Performance

Additional Key Words and Phrases: Semantic role labeling, generalized features, similar arguments, K-means

ACM Reference Format:
Haitong Yang and Chengqing Zong. 2016. Learning generalized features for semantic role labeling. ACM Trans. Asian Low-Resour. Lang. Inf. Process. 15, 4, Article 28 (April 2016), 16 pages.
DOI: http://dx.doi.org/10.1145/2890496

1. INTRODUCTION

Semantic Role Labeling (SRL) is a kind of shallow semantic parsing task and its goal is to recognize the related phrases and assign a joint structure (WHO did WHAT to WHOM, WHEN, WHERE, WHY, HOW) to each predicate of a sentence [Gildea and Juafsky 2002; Gildea and Palmer 2002]. SRL representations have many potential applications in natural language processing (NLP) and have been shown to benefit question and answering [Narayanan and Harabagiu 2004], information extraction

The research work has been supported by the Natural Science Foundation of China under Grant No. 61333018 and the Strategic Priority Research Program of the CAS under Grant No. XDB02070007.
Authors' addresses: H. Yang and C. Zong, Intelligence Building, No. 95, Zhongguancun East Road, Haidian District, Beijing, 100190, China; emails: {htyang, cqzong}@nlpr.ia.ac.cn.

DOI: http://dx.doi.org/10.1145/2890496

Table I. Some Examples to Show the Weak Generalized Power of the "Common" Features

Training Set	Test Set
in spring; in summer	in winter
in new york	in seattle

[Christensen et al. 2010; Surdeanu et al. 2003], and machine translation [Liu and Gildea 2010; Wu and Fung 2009; Xiong et al. 2012; Zhai et al. 2012, 2013], and so on.

The SRL task is usually treated as a supervised problem. Therefore, a huge set of features are crucial to the performance of SRL systems. The features often used include lexicalization features (e.g., *the last word of the argument*) and syntax features (e.g., *the syntax tag of the argument*). But these features, especially the lexicalization features, often do not generalize well. For instance, if we want to classify the argument "in winter" of Table I, the lexical information "*winter*" is crucial to discriminate the argument. Unfortunately, there is only the arguments like "in spring" and "in summer" in the training set except "in winter." In other words, the feature of the last word "*winter*" does not occur in the training set, which may cause the classifier to classify "in winter" incorrectly. The example shows the weak generalization power of the common features. However, intuitively, "in winter" is very similar to the arguments "in spring" and "in summer" in the training set and thus these arguments can activate some identical features. If we can learn these features, then they should have strong generalization powers and will be helpful to classify the arguments.

Previous works have shown that the out-of-vocabulary features like the above can be reduced by using word clusters, as has been done by Koo et al. [2008] and Deschacht and Moens [2009]. They introduced lexical intermediaries by learning word clusters from a large unannotated corpus. They evaluated their method in dependency parsing and found the substantial gains can be obtained. Similarly, Roth and Woodsend [2014] investigated distributed word representations for improving SRL. The advantage of the distributed word representations is that words with similar meanings will have similar representations, which can provide a more robust input signal to the classifier. Both word clusters and distributed representations can be thought as a generalization for the lexicalization features and they both produce positive effects to some extent. But, for the SRL task, an evident truth is that one word has a specific representation or cluster but this does not mean that it should be assigned a specific label. We show some examples to further illustrate. In the examples below, there are the same arguments "*the child*," but they are assigned different semantic roles because of different syntactic positions. Therefore, the lexicalization representations is insufficient for discriminating an argument. An important factor ignored by these methods is the syntactic information.

(a) *The child*[A0] broke the window.
(b) The mother blamed *the child*[A1] for breaking the window.

Differing from Koo et al. [2008], Roth and Woodsend [2014], and Fürstenau and Lapata [2009] addressed the out-of-vocabulary features with a semi-supervised learning. Their main idea is generating new training instances from an unlabeled corpus by automatically annotating. They first represented labeled and unlabeled sentences as graphs and formalized the search as a graph alignment problem in which graph alignment is scored using a function based on lexical and syntactic similarity. Then, they chose the top K similar sentences for each labeled sentence and automatically annotated these unlabeled sentences according to the labeled sentences. Although both the lexicalization and the syntactic information are incorporated in their method, the generalization power is also limited. For example, to correctly label "in winter" of

Table I, their method has to add training samples which could offer the feature of the last word "*winter*." Moreover, these training samples should be labeled as AM-TMP by automatically annotating. Therefore, these rigorous conditions limit the generalization power of their method.

Differing from the above methods, this article makes efforts to learn generalized features for SRL. The motivation of our work is that arguments that are lexically similar are likely to represent the same semantic role, and, similarly, arguments occurring in similar syntactic positions are likely to bear the same semantic role. Thus, the key idea of this work is to make a group of similar arguments activate one feature and another group of similar arguments activate another feature. In specific, we use a clustering method to reach the goal. We embed the information of lexicalization and syntax into a feature vector for each argument and then cluster all feature vectors of training set. For an unseen argument (e.g. "in winter"), it will go to the same cluster as the similar arguments (e.g. "in spring") in the training set. Therefore, compared with the common features such as "*the last word of the argument*," the learned clusters can be expected to be a kind of generalized feature that should be helpful for SRL.

We have conducted experiments on two standard benchmarks: Chinese PropBank and English PropBank. We evaluate our method under various conditions. The experimental results show that compared with the baseline system, the accuracy of our model improves significantly on Chinese PropBank and English PropBank.

The remainder of this article is organized as follows. Section II gives an introduction to the corpus used in our work and the baseline system. The proposed method is presented in Section III. The experiments and results are presented in Section IV. Section V discusses the related works. Finally, the conclusion is presented in Section VI.

2. SEMANTIC ROLE LABELING

This section introduces the corpora used in the experiments and the baseline system.

2.1. PropBank

Charles Fillmore proposed a theory of meaning called Frame Semantics in which abstract actions or common situations are defined as a frame and elements related with a frame are defined as semantic roles. Inspired by his theory, two handcrafted corpora, FrameNet [Baker et al. 1998] and PropBank [Gildea and Palmer 2002], have been constructed. Compared with FrameNet, PropBank provides unified roles for different predicates and has received the most attention from the research community, and thus we adopt it in this work. In the following, we use an example from the official anotation guidelines for English PropBank[1] to give a brief illustration about Propbank.

[*Mr. Bush*]$_{A0}$ **met** [*him*]$_{A1}$ [*privately*]$_{AM\text{-}MNR}$, [*in the White House*]$_{AM-LOC}$, [*on Thursday*]$_{AM\text{-}TMP}$.

The above sentence describes a frame scene of "*meet*" in which "*Mr. Bush,*" labeled as A0, is the agent, "*him,*" labeled as A1, is the patient and other roles provide adjuncts for the scene such as manner (MNR), locative (LOC), temporal (TMP). From the simple example, we can see that the unified annotation of Propbank can provide rich semantic information about a sentence in a concise way.

2.2. Baseline

An SRL system usually takes a parse tree (either handcrafted parse trees or auto parse trees generated by an automatic parser) as the input and assigns appropriate

[1]http://verbs.colorado.edu/mpalmer/projects/ace/EPB-annotation-guidelines.pdf.

semantic roles to the constituents in the parse tree, which are semantic arguments to the predicate in question. Most SRL systems work in stages, which consist of a pruning stage, an argument identification stage, and an argument classification stage. In the pruning stage, the goal is to eliminate the obvious non-argument candidates. The heuristic pruning method in Xue [2008] has been widely accepted by the SRL community and thus we implemented his method in this article. This article mainly focuses on argument classification. Argument classification, which assigns appropriate semantic roles to candidates identified in the argument identification stage, is a natural multi-category classification problem.

2.2.1. Classifier. In the literature, many classification techniques, such as SVM [Pradhan et al. 2005], Perceptron [Carreras et al. 2004], and Maximum Entropy [Xue and Palmer 2003; Xue 2008], have been successfully used to solve the tasks of argument identification and argument classification. In the work, we used a Maximum Entropy classifier with a tunable Gaussian prior. As a discriminative model, the Maximum Entropy classifier is scalable to handle a large set of features. Moreover, the Maximum Entropy classifier does multi-category classification naturally and thus can be straightforwardly applied in the problems here. Zhang Le's MaxEnt toolkit[2] is used for the specific implementation.

2.2.2. Features. Features are crucial to the performance of SRL systems. Many works [Xue and Palmer 2004; Xue 2008] have studied what features are discriminative for semantic role labeling. In their works, the authors experimentally proved that the stages of argument identification and classification require different features. Therefore, we follow their suggestions and utilize two groups of features: one for argument identification and the other for argument classification. The following lists describe the features in detail.

The features for argument identification are as follows:

—Predicate lemma
—Path from node to predicate
—Head word
—Head word's part-of-speech
—Verb class
—Predicate and Head word combination
—Predicate and Phrase type combination
—Verb class and Head word combination
—Verb class and Phrase type combination

All of the above features are also used in argument classification. In addition, there are some other features:

—Position: the relative position of the candidate argument compared with the predicate
—Subcat frame: the syntactic rule that expands the parent of the verb
—The first and the last word of the candidate
—Phrase type: the syntactic tag of the candidate argument
—Subcat frame+: the frame that consists of the NPs

3. THE PROPOSED METHOD

In this section, we will introduce our method in detail. Our main idea is that we first finely construct a feature vector for all samples of training set and then use *K*-means to cluster all feature vectors. Then the learned clusters are added into the classifier as

[2]http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html.

a new feature. When an unseen sample is to be classified, we first obtain the nearest cluster by computing the distances between it to the centers of all clusters and choose the cluster as the new feature.

3.1. Feature Vector

Semantic Role Labeling is a kind of shallow semantic parsing for natural sentences. For a sentence, all its roles construct the skeleton of its semantics. But how to represent a role of a sentence using a feature vector is an open and hard problem. A feasible way is to use all features in Section II to represent a role but most features such as the first and the last word of the argument are very sparse. If we represent a role in this way, then the length of the feature vector will be up to 100,000. It is also very inefficient to cluster vectors with such a high dimension if no special technique is done. Moreover, the feature vector is too sparse to learn good cluster information. Intuitively, lexicalization and syntax information are two key elements for representing a role. In the following section, we will introduce how to use the two elements to represent the roles of a sentence.

3.1.1. Lexicalization Information. We represent the lexicalization information of an argument using the technique of distributed word representation. Distributed representations represent words in some low-dimensional space, where each dimension might correspond to some syntactic or semantic property of a word. Distributed word representation recently has caught much attention and has wide applications. These word representations can be used to create novel features [Miller et al. 2004; Koo et al. 2008; Nguyen and Grishman 2014; Roth and Woodsend 2014; Sun et al. 2011; Turian et al. 2010] and can also be treated as model parameters to build representations for higher-level structures in some compositional embedding models [Collobert et al. 2011; Collobert 2011; Hermann et al. 2014; Socher et al. 2012; Socher et al. 2013]. In practical applications, distributed word representation have boosted the performance of many NLP tasks, such as syntax parsing [Collobert 2011; Turian et al. 2010], semantics parsing [Hermann et al. 2014; Socher et al. 2012; Socher et al. 2013], question answering [Bordes et al. 2014], and machine translation [Devlin et al. 2014].

While an argument usually consists of multiple words, in our method, we compute additive compositional representations of all words of an argument. This is the simplest method of Mitchell and Lapata [2010], where the composed representation is the uniformly weighted sum of each single representation.

Although compositional models aim to learn the higher-level structure representations, composition of distributed representations alone may not capture the important syntactic or semantic patterns. Most research on distributed representations evaluate their methods just by conducting word-level experiments such as word similarity, which only show the power of distributed representations at generalizing the lexicalization information. Therefore, there is still not sufficient evidence to demonstrate its advantages at incorporating syntax or semantics which is crucial to SRL. We take the examples of "the child" in Section I to further illustrate. The two arguments "*the child*" have the same lexicon but their roles differ. In this case, the syntactic information is crucial to correctly classifying the two arguments. If only lexicalization information is considered, the classifier may classify these arguments incorrectly. The example clearly shows the deficiency of using lexicalization information alone at representing an argument.

3.1.2. Syntactic Information. Just as the above states, syntactic information is an important factor for representing an argument. But what kind of syntax is the most effective to represent an argument of a sentence is still controversial. For example, the syntactic path is a kind of syntactic information to represent an argument, but the syntactic path

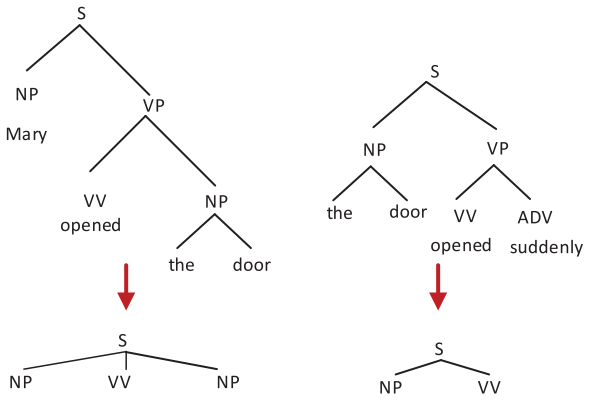

Fig. 1. Two flat trees.

is too sparse to learn helpful information. For an unseen argument, its syntactic path may not be found in the training set. Moreover, the syntactic path is also redundant, for example, there are many inessential components like "NP^NP," which means that the parent of the argument is also an NP. In the work, we experimentally choose the flat tree to represent the syntactic information. Our method is a variant of *Argument Keys* in Titov and Klementiev [2012]. In the following, we will illustrate how to represent an argument using a flat tree.

A flat tree, as opposite to a hierarchical tree, reflects the syntactic structure of a sentence in a flat way. Compared with a hierarchical tree, a flat tree provides a more compact way to understand a sentence. The flat tree used in the work slightly differs from the common one and it consists of syntactic tags of the core arguments and the predicate in a sentence. The flatting process of a syntactic tree is as follows:

(a) For a given predicate, collect all core arguments and the predicate itself.
(b) Arrange the syntactic tag of the above nodes into a line according to the position order.

We take some examples to give a further description. There are two sentences in Figure 1. After flattening, the structure of the left sentence becomes "NP-VV-NP" and the right sentence becomes "NP-VV." It is worth noting that the adjunct arguments like the argument "suddenly" in the right sentence are removed from the flat tree. There are three reasons for removing adjunct arguments. First, the core arguments like "'A0" and "A1" are key elements to convey the meanings of a sentence while adjunct arguments are used to reveal the peripheral information such as Time, Location, Manner, and Extent. Second, in most cases, the adjunct arguments like "suddenly" can be correctly classified just using the lexicalization information. Third, removing the adjunct arguments helps control the dimensionality of syntactic information.

After the whole flat tree is obtained, we incorporate the flat tree with the position of the argument in question to represent the argument. For the arguments "Mary" and "the door" in the left sentence of Figure 1, we represent them by using "NP^-VV-NP" and "NP-VV-NP^," respectively, in which the symbol "^" indicates the argument in question. We can see that this kind of representation can capture all core elements related to the predicate which suggests some global information can be encoded in the representation. Moreover, compared with other syntax representations like syntactic path, this kind of representation is very concise and compact.

The two sentences of Figure 1 are both in the active voice, but some sentences in the passive voice may also occur. We find that the roles in the sentences with different

voices, especially for A0 and A1, have different preference for the position in the flat tree. For instance, there are two sentences below and one is in the passive voice.

(a) *Mary*[A0] opened the door.
(b) The door was opened by *Mary*[A0].

The two sentences have the same meaning, which suggests that the same roles of the two sentences should have the same representation. But for "Mary" in sentence (a) the representation is "NP^-VV-NP" and for "Mary" in sentence (b) the representation is "NP-VV-NP^." Therefore, we design two rules to handle the sentences in passive voice. These two rules are designed as follows:

(i) The subject[3] of the predicate in the passive voice is put after the predicate.
(ii) If a prepositional phrase leaded by "by" directly follows the predicate, then it will be put in front of the predicate.

After processing the flat tree of the sentence (b) by using the two rules, we obtain the same representations for the corresponding roles like "Mary" in the two sentences.

The representation constructed above is discrete. We then transform it into a feature vector by using a *one-hot* representation. We collect the flat representations for all arguments in the training set into a list. The feature vector has the same length as the list and only one dimension is on. It is worth noting that, according to the above illustration, the adjunct arguments do not have the syntax part and thus its syntax part is all off.

3.2. *K*-Means Clustering

According to the above description, we can obtain the feature vectors in the lexicalization part and the syntax part for an argument. The feature vector in the lexicalization part is denoted F_{lex} and the feature vector in the syntax part is denoted F_{syn}. Then we concatenate F_{lex} and F_{syn} into a single vector F to represent this argument. Intuitively, the arguments with the same roles have similarity in the lexicalization and the syntax, which indicates that their feature vectors F should be very similar and close. Therefore, the idea of our method is to cluster all feature vectors of arguments in training set. The clusters learned should generalize better than the common features.

In particular, we use the K-means algorithm to implement clustering. There are no special requirements for the clustering algorithm, so other clustering algorithms can also be adopted. But K-means is a very simple and efficient clustering algorithm, and thus it is adopted here. The K-means algorithm consists of the following four steps:

(i) Select K vectors as the initial centroids for K clusters.
(ii) Assign each vector to the cluster with the nearest distance.
(iii) Compute each cluster's centroids by averaging all its vectors.
(iv) Repeat Step (ii) and (iii) until convergence.

In Step (ii), a distance function is required to measure the distance of a vector to the centroid of a cluster. The Euclidean distance is adopted as the distance function. Its definition is as follows:

$$\begin{aligned} dis(F, O) &= dis_{lex} + \lambda * dis_{syn} \\ &= \frac{F_{lex} \cdot O_{lex}}{|F_{lex}| * |O_{lex}|} + \lambda * \frac{F_{syn} \cdot O_{syn}}{|F_{syn}| * |O_{syn}|}, \end{aligned} \tag{1}$$

[3]The subject tag cannot be obtained directly and thus here we handle the nearest NP before the predicate instead of the subject node.

in which F and O denote the vector representations for an argument and a cluster's centroid, and the subscripts *lex* and *syn* denote the lexicalization part and the syntax part of F and O, respectively. In Equation (1), the distance of F and O consists of two sub-distances, dis_{lex} and dis_{syn}, which denote the distance of F and O in the lexicalization and syntax parts, respectively. A hyper-parameter λ is added to trade off the distances in the lexicalization and syntax.

We extract the feature vectors of all arguments in the training set and implement K-means clustering on them following the above steps. After K-means clustering is done, we obtain the centroids of K clusters. We number all clusters for distinction. Each argument of training set will be assigned to the nearest cluster and the number of the cluster will be added into the classifier as a new feature. For an argument of test set, we extract its feature vector, and the number of its nearest cluster is the new feature.

4. EXPERIMENTS

4.1. Experimental Settings

To evaluate the performance of our approach, we have conducted experiments on two standard benchmarks: Chinese PropBank and English PropBank. The experimental settings for Chinese and English are given as follows:

Chinese: We use Chinese Proposition Bank 1.0 as the evaluation corpus in the Chinese SRL system. All data are divided into three groups, whereby 648 files (from chtb_081.fid to chtb_899.fid) are used as the training set and 40 files (from chtb_041.fid to chtb_080.fid) constitute the development set. The test set consists of 72 files (chtb_001.fid to chtb_040.fid and chtb_900.fid to chtb_931.fid). This data setting is the same as in Xue [2008]. We adopt Berkeley Parser[4] to carry out auto parsing for SRL and the parser is retrained on the training set.

English: We choose English Propbank as the evaluation corpus in the English SRL system. According to the traditional partition, the training set consists of the annotations in Sections 2 to 21, the development set is Section 24, and the test set is Section 23. This data setting is the same as in Toutanova et al. [2008]. We adopt Charniak Parser[5] to carry out auto parsing for SRL and the parser is retrained on the training set.

Distributed word representation

To construct vector representations of words, we run the public word2vec[6] tool on a public large corpus the Xinhua portions of English Gigaword (LDC2003T05) and Chinese Gigaword corpus (LDC2003T09), using the CBOW model with a window of 10 and hierarchical softmax. The dimension size of word vectors is set 500.

Evaluation metric

For fair comparison, we use the widely accepted criteria: F_{score} ($\mathbf{F_1}$). Following Zhang et al. [2004], we conducted the statistical significance tests with 1,000 resampling size and 95% confidence interval. In the following tables, an asterisk is used to indicate that the difference between the proposed approach and the corresponding baseline system was statistically significant.

Hyper-parameters tuning

There are two hyper-parameters, K and λ, in our method and we tune them on the development set.

[4]http://code.google.com/p/berkeleyparser/.
[5]https://github.com/BLLIP/bllip-parser.
[6]http://word2vec.googlecode.com/svn/trunk/.

Table II. Results

		Auto	Gold	Known	All-Known
Ch	Baseline	74.04	90.25	92.78	92.78
	Roth	74.29	90.52	93.15	93.15
	Koo	74.34	90.50	93.20	93.20
	Lex	74.24	90.45	93.00	93.00
	Syn	74.35	90.70	93.42*	93.80*
	Lex+Syn	**74.51***	**91.10***	**93.63***	**94.30***
En	Baseline	76.15	87.45	92.37	92.37
	Roth	76.38	87.75	92.62	92.62
	Koo	76.30	87.82	92.74	92.74
	Lex	76.32	87.60	92.55	92.55
	Syn	76.28	87.78	92.88*	93.30*
	Lex+Syn	**76.53***	**88.00***	**92.91***	**94.00***

4.2. Results and Discussion

We evaluate our method on the test set of Chinese Propbank and English Propbank under various experimental conditions. The conditions are set as follows:

Auto: This condition adopts completely auto modules in all stages of SRL, which includes auto parse trees as the input, the argument identification stage, and the stage that separates the core arguments from the adjunct arguments. Because the flat tree defined in the work consists of the syntax tags of core arguments, it is necessary to judge whether an argument is a core one. Here, we use the argument classification module of the baseline system to automatically perform judgement.

Gold: The difference of this condition with **Auto** is that it adopts the gold syntax trees as the input. In this condition, we also determine automatically whether an argument is a core one.

Known: The difference of this condition with **Gold** is that it ignores the argument identification stage. In other words, this condition assumes that all argument candidates are known and it focuses on the argument classification stage. In this condition, we also determine automatically whether an argument is a core one.

All-Known: The difference of this condition with **Known** is that it assumes whether an argument is a core one is also known.

Table II shows the results of contrast systems and our method in various conditions. The contrast systems contain *Baseline*, which is described in Subsection 2.2; *Roth*, in which we implement the method of Roth and Woodsend (2014) in our Baseline system; and *Koo*, in which we implement the method of Koo et al. (2008) in our Baseline system. Our method contains three systems: "Lex," "Syn," and "Lex+Syn." "Lex" denotes the system where the clusters are learned only from lexicalization information, "Syn" denotes the system where the clusters are learned only from syntactic information, and "Lex+Syn" denotes the system where the clusters are learned from both.

From the results of Table II, we can get the following observations:

—Compared with Baseline, "Roth," "Koo," and "Lex" improve the performance of SRL systems because they could learn helpful information from lexicalization information of large unlabelled data. However, the improvement is very limited, only about 0.2%–0.3% under the Auto condition, which indicates that learning informative features only from lexicalization information is insufficient for discriminating semantic roles.
—Compared with other systems, both Chinese and English SRL systems with "Lex+Syn" get better performance, especially in the condition of "All-Known," where the F_1 score improves significantly by 1.5 points and 1.6 points on Chinese PropBank and English PropBank, respectively.

Table III. Evaluating the Importance of a Single Feature

Feature	Strategy I	Strategy II
All	94.30	94.30
Position	92.28	95.05
Predicate	94.27	94.54
Predicate class	93.85	94.59
Subcat	94.13	94.63
Syntax tag	94.25	94.84
First word	94.50	94.61
Last word	94.30	94.54
Path	94.20	94.52
Head	94.25	94.50
Head's tag	94.29	94.86
Predicate&Head	94.24	94.45
Predicate&SyntaxTag	94.18	94.40
PredicateClass&Head	94.22	94.52
PredicateClass&HeadTag	93.90	94.62
Subcat Frame+	92.80	94.66
Lex+Syn	**90.97**	**92.78**
Greedy	**94.60**	**95.20**

—Our method get better performance in the condition of "All-Known" than in other conditions. The reason is that there are some argument identification errors and core argument identification errors in other conditions. These errors can generate cascading errors in constructing the flat tree, which will cause negative effects on the overall performance.

In the "Lex+Syn" system, we further investigate every feature's importance to the performance by removing each feature independently and the experimental condition is "All-Known" of the Chinese SRL system. We adopt two strategies to remove features: (I) Do not train a new model and directly remove one feature from the system, and (II) differing from the strategy I, the strategy retrains the classifier after removing one feature. The results are shown in Table III.

In Strategy I, the model has been trained by using all features and when testing we remove one feature at a time to investigate its importance to the whole performance. We find that, after being removed, most features cause performance drop while the proposed feature, "Lex+Syn," causes the most severe drop, which indicates that the proposed feature is crucial to the trained model. An interesting point is that "First word" and "Last word" do not cause any performance drop but slight improvement. We think the reason is these lexicalization features can be covered by the proposed feature since the lexicalization information has been encoded into the proposed feature.

In Strategy II, after removing one feature, we retrain the classifier. To our surprise, after being removed, almost every feature helped improve the performance. However, after the proposed feature is removed, the performance drops 1.6 points to 92.78%, which indicates that the common features can harm the performance in some cases due to the weak generalization power but the proposed feature is indispensable for argument classification.

The results in Table III also show that more features do not always bring better performance and feature selection is important. Here, we adopt a simple greedy strategy for feature selection. Although the best solution may not be found, this strategy possibly can produce a good local optional solution. Each time we choose one of the feature templates and remove it from the system. In one iteration, the one, after which

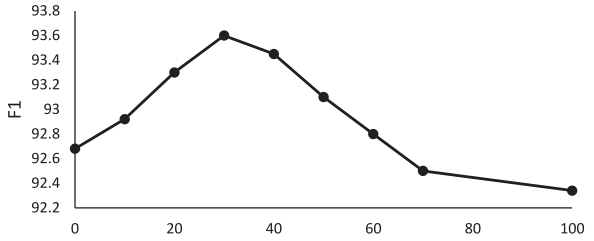

Fig. 2. Results of varying the max clustering number K on the development set.

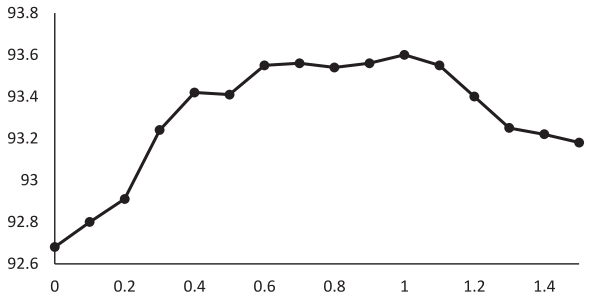

Fig. 3. Results of varying the max clustering number λ on the development set.

it is removed, the performance is the highest, will be removed. Then we continue the above iteration process until the performance on the development set does not improve. Through the greedy feature selection, the results of Strategy I and Strategy II achieve 94.60% and 95.20%, which are higher than the results of all features. Moreover, we find that the removed features in the greedy selection are lexical features like "Predicate," "First word," "Last word," and so on, which often generalize poorly.

4.3. Effects of Hyper-Parameter *K*

We tune the hyper-parameter K of the K-means algorithm on the development set. The experimental condition is "All-Known" of the Chinese SRL system and the hyper-parameter λ is set to a default value 1.0. Figure 2 shows the results of varying K on the development data. We can see that the proposed method performs better than the baseline system when K is less than 60. The best results are achieved when K is 30. But if K is bigger than 60, the performance of our method is lower than the baseline system, which suggests that a large clustering number causes the drop of our feature's generalization power. Thus, the K in the experiments on the Chinese PropBank is set to 30. The K of the experiments on the English PropBank is also tuned by the same method.

4.4. Effects of Hyper-Parameter λ

We tune the hyper-parameter λ of K-means algorithm on the development set. The experimental condition is "All-Known" of Chinese SRL system and the hyper-parameter K is set to 30 according to the above subsection. In our method, λ is introduced to trade off between the lexicalization information and syntactic information in weighing the similarity of two arguments. When λ is assigned 0, this means that only the lexicalization information is utilized to represent an argument and we can see that the performance of our method under this case is slightly better than the baseline system. With the growth of λ, the syntactic information takes more weight in computing the similarity of two arguments and our method performs much better than the case of λ equalling to 0, which again demonstrates that the syntactic information is crucial to

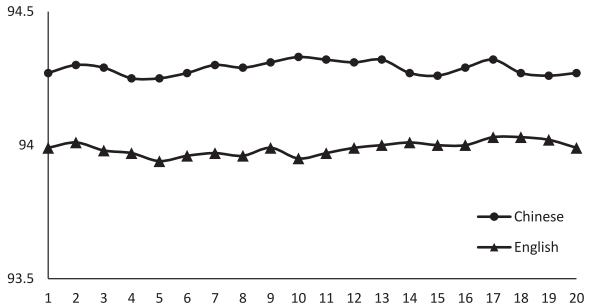

Fig. 4. Evaluating the stability of the proposed method.

Table IV. Results on Brown Test Set

	Auto	Gold	Known	All-Known
Baseline	62.63	77.19	81.25	81.25
Ours	**63.36***	**77.84***	**82.37***	**84.80***

representing an argument. However, the performance drops slightly if λ is bigger than 1.0. Thus, the K in the experiments on the Chinese PropBank is set to 1.0. The λ of the experiments on the English PropBank is also tuned by the same method.

4.5. The Stability of Our Method

K-means clustering is usually thought of as an unstable algorithm and the clustering results are easily influenced by the initial values. Here, we investigate whether the unstable clustering results will influence the performance of the SRL system. The experimental condition is "All-Known" of the Chinese SRL system and the English SRL system. Figure 4 shows the results of running 20 times K-means clustering separately. We can see that, although the clustering results are unstable, the results of SRL systems are stable and the gap between the maximum value and the minimum value is lower than 0.1 points.

4.6. The Out-of-Domain Evaluation

An SRL system often suffers severe performance drops on out-of-domain test data due to the diversity of features of different domains. Intuitively, the common features learned from in-domain data generalize more poorly in out-of-domain test data than in in-domain data. We further investigate whether the proposed method helps relieve the problem of domain adaptation. The results for the out-of-domain experiments are summarized in Table IV. The evaluation corpus utilized is the Brown part of the English PropBank. From the results of Table IV, we can see that the performance of baseline is lower by about 10 points in out-of-domain data than in in-domain. However, after the generalized features are added, the accuracy improves significantly by 1.1 points and 3.6 points under Known and All-Known, respectively. Moreover, the absolute gains are slightly higher than in the in-domain setting. Therefore, the proposed features help relieve the diversity of different domains.

4.7. Integrate the Generalized Features into the State-of-the-Art SRL System

The publicly available system *Mate*[7] ranks at the top in the SRL-only closed challenge of the Conference on Computational Natural Language Learning (CoNLL) 2009 shared tasks and represents the current state of the art for SRL. Thus, we integrate the

[7]http://code.google.com/p/mate-tools/.

Table V. Comparison with the State-of-the-Art System

		Auto	Gold	Known	All-Known
Ch	Mate	78.40	87.24	93.45	93.45
	Ours	**78.72***	**87.69***	**94.32***	**95.29***
En	Mate	85.38	88.97	94.38	94.38
	Ours	**85.64***	**89.42***	**95.20***	**95.86***

generalized features into *Mate* to further evaluate our method. In the experiments, we evaluate our methods on the Chinese and English portions of CoNLL 2009 shared tasks, and the same training, development, and test sets provided in the CoNLL shared tasks are utilized for a fair comparison. We report labelled semantic F_1 score as computed by the official scorer.[8] The results of in-domain tests are shown in Table V. We can see that our method helps improve the F_1 scores of both the Chinese SRL system and the English SRL system, especially in the condition of "All-Known," where the F_1 score improves significantly by 1.8 points and 1.5 points on the Chinese and English portions, respectively. However, the improvement in the condition of "Auto" is very limited, which is caused by the cascading errors in constructing the flat tree. In the future, we will exploit more robust syntactic representation of an argument.

5. RELATED WORK

The annotated corpuses Framenet and Propbank have greatly boosted the development of Semantic Role Labeling. Gildea and Jurafsky [2002] first presented an auto SRL system based on a statistical classifier that is trained on a hand-annotated corpora FrameNet. In their pioneering work, they used a gold or autoparsed syntax tree as the input and then extracted various lexical and syntactic features to identify the semantic roles for a given predicate. After Gildea and Jurafsky [2002], there have been a large number of works on automatic semantic role labeling. Based on a basic discriminative model, Punyakanok et al. [2004] constructed an integer linear programming architecture in which the dependency relations among arguments are implied in the constraint conditions. Toutanova et al. [2008] proposed a joint model to explore relations of all arguments of the same predicate. In addition, there have been many extensions in machine learning models [Moschitti et al. 2008], feature engineering [Xue and Palmer 2004], and inference procedures [Punyakanok et al. 2004; Toutanova et al. 2008; Yang and Zong 2014; Yang et al. 2015; Yang et al. 2016; Zhuang and Zong 2010a; Zhuang and Zong 2010b].

This article focuses on how to boost the generalization power of an SRL system. There have been some related works. Koo et al. [2008] showed that the out-of-vocabulary features can be reduced by using word clusters and in dependency parsing substantial gains can be obtained by using their method. Roth and Woodsend [2014] investigated distributed word representations that can provide a more robust input to the classifier. However, the syntactic information is ignored by these methods because the syntactic information is crucial to classifying an argument. Fürstenau and Lapata [2009] proposed a semi-supervised method in which they automatically annotated most similar unlabeled data for a labeled data and a new classifier is retrained on all these data. But, the generalization power of their method is limited because only some lexical features can be learned by using their method. Differing from these methods, we induce the generalized feature from similar arguments which are helpful for SRL.

[8]http://ufal.mff.cuni.cz/conll2009-st/eval09.pl.

6. CONCLUSION

This article proposes a simple method to learn generalized features for SRL. In current systems for SRL, people usually define a large set of features to achieve good performance. However, most of these features, like "first word" and "last word," generalize poorly when predicting an unseen argument. To relieve the problem, some researchers have tried adding auto labeled data or using distributed word representation learned from unlabeled data and they reported better performance. But the generalization power of these methods is limited. Intuitively, arguments occurring in similar syntactic positions are likely to bear the same semantic role, and, analogously, arguments that are lexically similar are likely to represent the same semantic role. Therefore, differing from previous works, we propose a simple method to induce generalized features from similar arguments. In specific, we embed the lexicalization and syntax information of each argument into a feature vector, and then the K-means algorithm is utilized to make clustering for all feature vectors. This makes the similar arguments go to the same cluster, and thus the learned clusters can be thought as a kind of generalized feature. We evaluate our method on two standard benchmarks: Chinese Propbank and English Propbank. The results demonstrate that our method can significantly improve the SRL performance. Meanwhile, the proposed method can help relieve the problem of domain adaptation.

ACKNOWLEDGMENTS

We thank the three anonymous reviewers for their helpful comments and suggestions. We also thank Liu Zhanyi and Wu Wenquan of Baidu Co. for insightful discussions.

REFERENCES

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (ACL'98)*.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014. Question answering with subgraph embeddings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*. Association for Computational Linguistics, 615–620.

Xavier Carreras, Lluís Màrquez, and Grzegorz Chrupała. 2004. Hierarchical recognition of propositional arguments with perceptrons. In *HLT-NAACL 2004 Workshop: 8th Conference on Computational Natural Language Learning (CoNLL'04)*.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2010. Semantic role labeling for open information extraction. In *Proceedings of the NAACL HLT 2010 1st International Workshop on Formalisms and Methodology for Learning by Reading*.

Ronan Collobert. 2011. Deep learning for efficient discriminative parsing. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS-11)*, Geoffrey J. Gordon and David B. Dunson (Eds.), Vol. 15. Journal of Machine Learning Research—Workshop and Conference Proceedings, 224–232.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.* 12 (Nov. 2011), 2493–2537.

Koen Deschacht and Marie-Francine Moens. 2009. Semi-supervised semantic role labeling using the latent words language model. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 21–29.

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. 2014. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 1370–1380.

Hagen Fürstenau and Mirella Lapata. 2009. Graph alignment for semi-supervised semantic role labeling. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11–20.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling for semantic roles. *Comput. Linguist.* 28, 3 (2002), 245–288.

Daniel Gildea and Martha Palmer. 2002. The necessity of parsing for predicate argument recognition. In *Proceedings of the Joint Conference of the 40th Annual Meeting of the ACL (ACL'02)*.

Karl Moritz Hermann, Dipanjan Das, Jason Weston, and Kuzman Ganchev. 2014. Semantic frame identification with distributed word representations. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 1448–1458.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*. Association for Computational Linguistics, 595–603.

Ding Liu and Daniel Gildea. 2010. Semantic role features for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING'10)*.

Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative Training. In *HLT-NAACL 2004: Main Proceedings*, Daniel Marcu Susan Dumais and Salim Roukos (Eds.). Association for Computational Linguistics, 337–342.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cogn. Sci.* 34, 8 (2010), 1388–1429. DOI:http://dx.doi.org/10.1111/j.1551-6709.2010.01106.x

Alessandro Moschitti, Daniele Pighin, and Roberto Basili. 2008. Tree kernels for semantic role labeling. *Comput. Linguist.* 34, 2 (June 2008), 193–224. DOI:http://dx.doi.org/10.1162/coli.2008.34.2.193

Srini Narayanan and Sanda Harabagiu. 2004. Question answering based on semantic structures. In *Proceedings of the 17th International Conference on Computational Linguistics (COLING'04)*.

Thien Huu Nguyen and Ralph Grishman. 2014. Employing word representations and regularization for domain adaptation of relation extraction. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, 68–74.

Sameer Pradhan, Kadri Hacioglu, Valerie Krugler, Wayne Ward, James H. Martin, and Daniel Jurafsky. 2005. Support vector learning for semantic argument classification. *Mach. Learn.* 60, 1–3 (Sept. 2005), 11–39. DOI:http://dx.doi.org/10.1007/s10994-005-0912-2

Vasin Punyakanok, Dan Roth, Wen-tau Yih, and Dav Zimak. 2004. Semantic role labeling via integer linear programming inference. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING'04)*. DOI:http://dx.doi.org/10.3115/1220355.1220552

Michael Roth and Kristian Woodsend. 2014. Composition of word representations improves semantic role labelling. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 407–413.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, 1201–1211.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 631–1642.

Ang Sun, Ralph Grishman, and Satoshi Sekine. 2011. Semi-supervised relation extraction with large-scale word clustering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 521–529.

Weiwei Sun, Zhifang Sui, Meng Wang, and Xin Wang. 2009. Chinese semantic role labeling with shallow parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1475–1483.

Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. 2003. Using predicate-argument structures for information extraction. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics (ACL'03)*. DOI:http://dx.doi.org/10.3115/1075096.1075098

Mihai Surdeanu and Jordi Turmo. 2005. Semantic role labeling using complete syntactic analysis. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*. Association for Computational Linguistics, 221–224.

Ivan Titov and Alexandre Klementiev. 2012. A bayesian approach to unsupervised semantic role induction. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Avignon, France, 12–22.

Kristina Toutanova, Aria Haghighi, and Christopher Manning. 2005. Joint learning improves semantic role labeling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. Association for Computational Linguistics, Ann Arbor, MI, 589–596. DOI:http://dx.doi.org/10.3115/1219840.1219913

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. 2008. A global joint model for semantic role labeling. *Comput. Linguist.* 34, 2 (June 2008), 161–191. DOI: http://dx.doi.org/10.1162/coli.2008.34.2.161

Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Uppsala, Sweden, 384–394.

Dekai Wu and Pascale Fung. 2009. Can semantic role labeling improve SMT?. In *Proceedings of the 13th Annual Conference of European Association for Machine Translation (EAMT'09)*.

Deyi Xiong, Min Zhang, and Haizhou Li. 2012. Modeling the translation of predicate-argument structure for SMT. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL'12)*.

Nianwen Xue. 2008. Labeling chinese predicates with semantic roles. *Comput. Linguist.* 34, 2 (June 2008), 225–255. DOI: http://dx.doi.org/10.1162/coli.2008.34.2.225

Nianwen Xue and Martha Palmer. 2003. Annotating the propositions in the penn chinese treebank. In *Proceedings of the Second SIGHAN Workshop on Chinese Language Processing*. Association for Computational Linguistics, 47–54. DOI: http://dx.doi.org/10.3115/1119250.1119257

Nianwen Xue and Martha Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'04)*.

Haitong Yang, Yu Zhou, and Chengqing Zong. 2016. Bilingual semantic role labeling inference via dual decomposition. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 15, 3, Article 15 (Jan. 2016), 21 pages. DOI: http://dx.doi.org/10.1145/2835493

Haitong Yang, Tao Zhuang, and Chengqing Zong. 2015. Domain adaptation for syntactic and semantic dependency parsing using deep belief networks. *Transactions of the Association for Computational Linguistics* 3 (2015), 271–282.

Haitong Yang and Chengqing Zong. 2014. Multi-predicate semantic role labeling. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*.

Feifei Zhai, Jiajun Zhang, Yu Zhou, and Chengqing Zong. 2012. Machine translation by modeling predicate-argument structure transformation. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING'12)*.

Feifei Zhai, Jiajun Zhang, Yu Zhou, and Chengqing Zong. 2013. Handling ambiguities of bilingual predicate-argument structures for statistical machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL'13)*.

Ying Zhang, Stephan Vogel, and Alex Waibel. 2004. Interpreting bleu/NIST scores: How much improvement do we need to have a better system?. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC'04)*. 2051–2054.

Tao Zhuang and Chengqing Zong. 2010a. Joint inference for bilingual semantic role labeling. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP'10)*.

Tao Zhuang and Chengqing Zong. 2010b. A minimum error weighting combination strategy for chinese semantic role labeling. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING'10)*.

Received August 2015; revised January 2016; accepted January 2016