# Efficient Latent Structural Perceptron with Hybrid Trees for Semantic Parsing

**Junsheng Zhou**[1,2], **Juhong Xu**[1,2], **Weiguang Qu**[1, 2, 3]
[1]School of Computer Science and Technology, Nanjing Normal University, China
[2]Jiangsu Research Center of Information Security & Privacy Technology, China
[3]State Key Lab. for Novel Software Technology, Nanjing University, China.
zhoujs, wgqu@njnu.edu.cn, xujuhong_1987@126.com

## Abstract

Discriminative structured prediction models have been widely used in many natural language processing tasks, but it is challenging to apply the method to semantic parsing. In this paper, by introducing hybrid tree as a latent structure variable to close the gap between the input sentences and output representations, we formulate semantic parsing as a structured prediction problem, based on the latent variable perceptron model incorporated with a tree edit-distance loss as optimization criterion. The proposed approach maintains the advantage of a discriminative model in accommodating flexible combination of features and naturally incorporates an efficient decoding algorithm in learning and inference. Furthermore, in order to enhance the efficiency and accuracy of inference, we design an effective approach based on vector space model to extract a smaller subset of relevant MR productions for test examples. Experimental results on publicly available corpus show that our approach significantly outperforms previous systems.

## 1 Introduction

Semantic parsing is the task of mapping a natural language (NL) sentence into a complete, formal meaning representation (MR) in a meaning representation language (MRL), which is a formal unambiguous language that allows for automated inference and processing [Kate et al., 2005]. Semantic parsing distinguishes the task from related tasks such as semantic role labeling [Carreras and Marquez, 2004] and other forms of "shallow" semantic parsing which do not generate complete, formal representations.

Structured prediction model consists in learning a mapping from inputs to structured outputs. In the past few years, discriminative structured prediction models have become an important tool in many natural language processing tasks. Although they take much longer to train than generative models, they typically produce higher performing systems, in large part due to the ability to incorporate arbitrary features. Naturally, discriminative structured prediction model is well suited to formalizing the semantic parsing task. However, the semantic parsing task in this framework differs from traditional applications of discriminative structured prediction such as POS tagging or syntactic parsing. This is due to the fact that the correspondence between sentence substrings and parts of the meaning representation is unknown. To close the gap, in this paper, we develop a latent structural prediction approach for semantic parsing. Specifically, we use the latent variable perceptron as the discriminative model for semantic parsing, with the hybrid tree as the latent structure variable. The proposed approach maintains the advantage of a discriminative model in accommodating flexible combination of features and naturally incorporates an efficient decoding algorithm in learning and inference.

In practice, we observe that, in the testing phase, the learned latent structural perceptrons are able to attain higher efficiency and accuracy when given a smaller MR production subset containing relevant MR productions. Therefore, we design a simple yet effective approach based on vector space model to extract relevant MR productions for test examples.

Experiments on the publicly available corpus verify the effectiveness of the proposed approach, as the accuracy of semantic parsing can be improved significantly over other systems. Additionally, as some previous work, the proposed method does not require any prior knowledge of the NL syntax for training.

## 2 Related Work

Semantic parsing has attracted considerable interest in recent years. Previous work on learned semantic parsing falls under one of two approaches. One approach does semantic parsing based on syntax structure [Ge and Mooney, 2005; Ge and Mooney, 2006; Ge and Mooney, 2009]. Another major approach to semantic parsing is based on semantic grammars. Obviously, in contrast to syntax-based approach, the approach based on semantic grammar does not require any prior knowledge of the NL syntax for training and it is relatively easy to port to different NLs. In this paper, we focus on previous work related to the semantic grammars.

Among the discriminative learning-based methods to semantic parsing, WASP [Wong and Mooney, 2006; Wong

and Mooney, 2007] learns a synchronous context-free grammar (SCFG) for semantic parsing based on machine translation techniques. [Kate and Mooney, 2006] introduce a semantic parser called KRISP based on string kernels. For each semantic concept in a MR language, a string-kernel-based classifier is trained to capture a potentially infinite number of production rules.

Recently, [Lu et al., 2008] propose an algorithm for learning a generative model for semantic parsing. The proposed generative model coupled with a discriminative reranking technique, achieves state-of-the-art performance. But despite its apparent success, there remains a major drawback: this method suffers from the limited scope of the k-best list, which rules out many potentially good alternatives. Besides, [Jones et al., 2012] extend the generative models based on tree transformation and proposes using the tree transducer model, a formalism from automata theory, for semantic parsing, and introduces a variational Bayesian inference algorithm.

Additionally, some work has explored learning to map sentences to lambda-calculus meaning representations [Wong and Mooney, 2007; Zettlemoyer and Collins, 2005; Zettlemoyer and Collins, 2007; Kwiatkowski et al., 2010]. To some extent, our model is closely related to the work in [Zettlemoyer and Collins, 2007]. However, the model based on CCG grammar generally developed CCG grammar induction techniques where lexical items were proposed according to a set of hand-engineered lexical templates, which requires more supervision than our model.

In this paper, we restrict our meaning representation formalism to a variable free version as presented in [Wong and Mooney, 2006; Lu et al., 2008; Jones et al., 2012]. We propose a latent structural prediction model for semantic parsing, coupled with an extraction approach of relevant MR productions for inference. The benefits are two-fold: First, the approach provides a unified way to directly optimize the decoding phase of semantic parsing with the use of latent variable perceptron. Another benefit of our feature based discriminative structured prediction model is that it effortlessly allows smoothing over previously unseen MR productions.

# 3 Applying Latent Structural Perceptron to Semantic Parsing

In this section, we first present the latent structural prediction formulation of the problem of semantic parsing. Then we introduce in detail the model we use for solving the problem.

## 3.1 Discriminative Latent Structural Learning for Semantic Parsing

The task of semantic parsing is transforming a NL sentence into a formal meaning representation. It is natural to formulate the process as a structured prediction, which assigns a given NL sentence to a semantic representation. A general discriminative formulation of the problem is of the following form:

$$f(x) = \arg\max_{y}(w \cdot \Phi(x, y))$$

where $\Phi(x, y)$ is a vector of features and $w$ is the model parameter. However, it is unclear how $\Phi(x, y)$ can be computed. There exists no direct correspondence between the words in $x$ and the nodes in $y$.

To cope with this problem, we introduce a latent structure variable $h$ to close the gap between input sentences and output semantic representations. Then the above discriminative formulation of the problem can be rewritten as follows:

$$f(x) = \arg\max_{h,y}(w \cdot \Phi(x, h, y))$$

The challenges of successfully applying the latent variable model to this problem formulation are 1) how can we introduce an appropriate latent structure variable to model the correspondence between the input and the output; 2) how can we design a learning algorithm to learn the model parameter $w$ to directly optimize the maximization problem; 3) how can we solve the maximization efficiently without having to enumerate all candidates; 4) how can we design features to guarantee the correctness of the decoding algorithm. In the following subsections we introduce our solutions to these challenges in detail.

## 3.2 Hybrid Tree

We use the hybrid tree to model the correspondence between input sentences and output MR trees, because it provides a natural structure to express the correlations between the NL words and MR productions, and does not require any prior knowledge of the NL syntax for training. A hybrid tree is a tree consisting of NL words as leaves and MR productions as internal nodes [Lu et al., 2008]. An example hybrid tree is shown in Figure 1. The symbols $w_1, \ldots, w_9$ are NL word sequences, and $m_a, \ldots, m_g$ are MR productions. The hybrid tree was proposed under the simple assumption that the semantics conveyed by an MR tree is fully determined by the root MR production and its child MR subtrees and the semantics of the root MR production can be made to correspond with NL word sequences. In other words, the semantics of an MR tree is conveyed in the way the NL word sequences and the child MR productions are interleaved. Further, the semantics of the child MR subtrees are expressed in a recursive manner. Notice that the natural language words in a hybrid tree are in sequential order, and the children of an MR production can be reordered to match the natural language word sequence.

However, the correct correspondence between NL words and MR trees is unknown. Many possible derivations could reach the same N-M pair[1], where each such derivation forms

[1] An N-M pair is a contiguous NL word sequence paired with its corresponding MR tree.

a hybrid tree. But, for each hybrid tree, only one MR tree can be derived from it. Naturally, the hybrid tree is well-suited to be the latent structure variable in our framework.

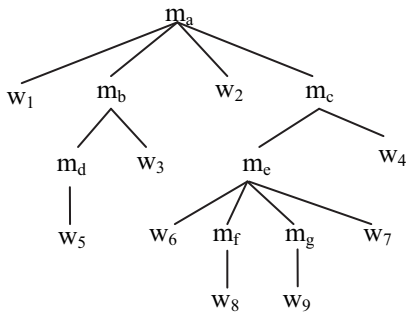


Figure 1: An example hybrid tree

### 3.3 Latent Structural Perceptron Learning

Based on the efficiency and convergence considerations [Collins, 2002; Sun et al., 2009], we employ the latent structural perceptron model for learning the discriminative model of semantic parsing. In Figure 2, we depict the proposed latent structural perceptron algorithm for the semantic parsing task. Like the structured perceptron, the latent structural perceptron is an online algorithm that iterates through the training set. This algorithm learns to predict hybrid trees that help to solve the parsing task. For each training instance, it performs two major steps: (i) a prediction for the given input using the current model; and (ii) a model update based on the large-margin principle.

Input: Training set $S = \{(x_i, y_i)\}_{i=1}^{N}$

$w^1 = 0$

for t = 1 to T do

  for i = 1 to N do

    $h' = \arg\max_{h,y}(w^t \cdot \Phi(x_i, y, h))$

    $y' = Proj(h')$

    if $y' \neq y_i$ then

      $h^* = \arg\max_h(w^t \cdot \Phi(x_i, y_i, h))$

      $w^{t+1} = update\ w^t\ \ according\ to\ (x_i, h^*, h')$

    else

      $w^{t+1} = w^t$

  end for

end for

Output: parameter vectors *w*

Figure 2: The training algorithm of the latent structural perceptron for semantic parsing

As shown in the algorithm, there exist two decoding tasks as follows:

$$h' = \arg\max_{h,y}(w \cdot \Phi(x_i, y, h))$$

$$h^* = \arg\max_h(w \cdot \Phi(x_i, y_i, h))$$

where $h^*$ denotes the constrained hybrid tree associated with the instance ($x_i$, $y_i$). Thus, we predict the constrained hybrid tree $h^*$ for the training instance ($x_i$, $y_i$) using a specialization of the latent predictor – the constrained latent predictor – that makes use of $y_i$. The constrained predictor finds the maximum scoring hybrid tree among all hybrid trees that cover all words in $x_i$ and only contain the MR productions occurring in the reference MR tree $y_i$. The constrained tree is used as the ground truth on each iteration. Additionally, the hybrid tree $h'$ can be predicted by the ordinary predictor, and then the predicted output $y'$ can be derived from the hybrid tree $h'$, which is denoted with the operation *Proj*(*h*).

Inspired by the online learning algorithm MIRA [Crammer et al., 2005], we update the parameter vector *w* based on the principle of large-margin. More specifically, we update the weight vector *w* by keeping the norm of the change in the weight vector as small as possible. Within this framework, we can formulate the optimization problem as follows [McDonald, 2006]:

$$w^{t+1} = \arg\min_w \left\| w - w^t \right\|$$

$$s.t.\ \ \forall h' \in best_k(x_i; w^t):$$

$$w^T \cdot \Phi(x, h^*, y_i) - w^T \cdot \Phi(x, h') \geq L(y_i, \Pr oj(h'))$$

where $best_k(x_i; w^t)$ represents a set of top k-best outputs for $x_i$ given the weight vector $w^t$, and $L(y_i, y')$ represents the loss function. In our implementation, the top *k*-best outputs are obtained with a straightforward *k*-best extension to the decoding algorithm described in subsection 3.5. The above quadratic programming (QP) problem can be solved using Hildreth's algorithm [Yair Censor, 1997].

Furthermore, the parameter update method is very flexible with respect to the loss function. Considering the output of semantic parsing corresponds to a MR tree, we introduce the tree edit distance to measure the loss between the predicted MR tree and reference MR tree. The tree edit distance between two trees is the minimum cost sequence of node edit operations (node deletion, node insertion and node rename) that transforms on tree into the other. The cost of a sequence $S$=$\{s_1, s_2, \ldots, s_n\}$ of edit operations is defined as:

$$\alpha(S) = \sum_{i=1}^{n} \alpha(s_i)$$

where $\alpha(s_i)$ is a cost function defined on each edit operation. In our implementation, we set a unit cost to each edit operation. The tree edit distance loss between the predicted parse tree and the reference one is efficiently computed using a dynamic programming algorithm [Zhang and Dennis, 1989].

As shown in [McDonald, 2006], parameter averaging can effectively avoid overfitting. The final weight vector *w* is the average of the weight vectors after each iteration.

### 3.4 Features

In the hybrid tree, the nodes are either NL words or MR productions, and generation of every NL word and child MR production depend on its direct parent MR production. In other words, all NL words and child MR productions in the hybrid tree are attached to their parent MR productions. To express the structural characteristics of a hybrid tree, we introduce four types of features: word features, production features, the mixture features of word and MR production, and hybrid pattern features. The former three types of features are used to capture the correlations between the parent MR production nodes and their child nodes. The last type describes the patterns the parent MR production nodes extend downward. Specifically, for a given MR production node in the hybrid tree, a hybrid pattern consists of NL word sequences below the node intermixed with the child MR productions. For simplicity, we assume that each MR production has at most two child semantic categories in its right hand side (RHS). Similar to [Lu et al., 2008], 21 possible hybrid patterns are considered in our model, as shown in Table 1. In Table 1, m is an MR production, X and Y are respectively the first and second child semantic category in m's RHS. The symbol w refers to a contiguous sequence of NL words.

The full set of features, along with an explanation of our notations, is listed in Table 2.

| #RHS | Hybrid Pattern | # Patterns |
|---|---|---|
| 0 | m→w | 1 |
| 1 | m→[w]X[w] | 4 |
| 2 | m→[w]X[w]Y[w] | 8 |
| | m→[w]Y[w]X[w] | 8 |

Table 1: A list of possible hybrid patterns. The notation [] denotes anything inside [] can be optionally omitted.

| | |
|---|---|
| Word features | par+w |
| | par+isConstant(w) |
| | predicate (par)+w |
| | par+$w_{-1}$+w |
| | predicate (par)+ $w_{-1}$+w |
| Production features | par+p |
| Mixture features | p+w / w+p |
| | Par+p+w / Par+w+p |
| hybrid pattern features | par + rule |

Table 2: Feature templates. w is the NL word and p the child MR production. $w_{-1}$ is the NL word to the left of w. par represents the direct parent MR production associated with a NL word or child MR production. rule represents the hybrid pattern. isConstant(w) checks whether w is a known constant, such as named entity. predicate(p) denotes the predicate drawn from the RHS of MR production p.

### 3.5 Decoding Algorithm

The essential decoding problem is to find the hybrid tree that maximizes the scoring function according to the model. For this purpose we design a dynamic programming algorithm to efficiently produce the optimal hybrid tree.

Input: Sentence *s* with *n* words, a set of candidate MR productions.
Algorithm:
for *len*=1 to *n* do
for *begin*=0 to *n* - *len* do
*end* = *begin* + *len*
for *m* as the next MR production in the set of candidate MR productions do
if *m* has no nonterminal in RHS then
if(*len*=1)
calculate directly the score of the subtree rooted by *m* and covering the only word *s*[*begin*].
else
calculate the score of the subtree rooted by *m* and covering the words *s*[*begin*..*end*] by combining the subtree covering previous (*len*-1) words and word *s*[*end*].
else if *m* has only one nonterminal in RHS then
calculate the highest-scored subtree rooted by *m* and covering the words *s*[*begin*..*end*] by decomposing the subtree according to the value of *len* and corresponding unary hybrid patterns.
else if *m* has two nonterminals in RHS then
calculate the highest-scored subtree rooted by *m* and covering the words *s*[*begin*..*end*] by decomposing the subtree according to the value of *len* and different segmentation of corresponding binary hybrid patterns.
end for
end for
end for
pick out the highest-scored hybrid tree rooted by a start MR production and covering all words in the NL sentence.
Output: the optimal hybrid tree corresponding to the sentence

Figure 3: A dynamic-programming algorithm for semantic parsing.

In the dynamic programming algorithm, let each subproblem correspond to a subtree rooted by some MR production in the hybrid tree and covering some words in the NL sentence. We decompose a subproblem according to the number of the words that the root MR production covers and all possible hybrid patterns associated with the root MR production. Then we can solve all subproblems in a bottom-up manner. The number of possible hybrid patterns in algorithm is more than 20, which results in the recursive expression in the dynamic programming being very complicated. Due to space limitation, a concise outline description of the decoding algorithm is shown in Figure 3. The time complexity of the above algorithm is $O(n^2T^2)$, where $n$ is the length of the sentence and $T$ is the number of candidate MR productions. It is quadratic in the length of sentence. However, the complexity of the algorithm also depends on the number of candidate MR productions.

# 4 Extraction of the Set of Relevant MR Productions

In the testing phase, in order to improve the efficiency and accuracy of inference, we propose an effective approach based on the vector space model to extract relevant MR productions for decoding, rather than using all possible MR productions as the candidates.

Motivated by the document ranking method, we cast the problem of relevant MR productions extraction as a MR productions ranking problem, with the use of vector space model [Lee et al., 1997]. However, different from document ranking problem, it is not trivial to represent every possible MR production as a vector. For each training instance, a reference MR tree containing all relevant MR productions is given. But each training instance contains multiple different relevant MR productions in the reference MR tree. To cope with the problem, we design a simple but efficient way to construct the vector for every MR production. Specifically, we first establish a vector for each training instance by extracting the unigrams, bigrams and trigrams of words in the NL sentence as meaning bearing terms in the vector. Then, for each MR production, a summation vector is formed by summing all vectors of training instances containing the same MR production all together, which will be viewed as the vector representing the MR production. The reason is as follows. The words or phrases closely related to some MR production may occur in more instances containing the MR production in the reference MR tree, which may result in a relatively higher term frequency in the summation vector corresponding to the MR production.

Next, an important problem in establishing a vector for each training instance is how to set the weights of terms in MR production vector. Due to the fact that there exist too many noise terms in the summation vector for every MR production formed based on above method, we introduce a modified tf-idf weighting scheme. Instead, we compute the relative term frequency, not traditional term frequency, because the relative frequency can better reflect how important a term is to a MR production.

When given a test example, we first construct a vector in the same manner, and then we can pick out the $n$-best MR productions[2] based on cosine similarity calculation, which are regarded as relevant MR productions for the test example. In the development experiments, we found that the average recall of relevant MR productions in the reference MR trees for test examples can achieve up to about 97%, but the precision is relatively low, attaining about 60%. Obviously, in contrast to using all possible MR productions as the set of candidate MR productions, the low precision doesn't hurt the performance of decoding.

# 5 Experimental Evaluation

## 5.1 Data sets and evaluation method

Our Experimental evaluations were performed on GeoQuery，a publicly available corpus [Wong and Mooney, 2006]. GeoQuery contains 880 English questions and database queries about United States geography. Recently, [Jones et al., 2012] presented the translations of the full 880 sentences into German, Greek, and Thai.

To make our system directly comparable to previous systems, we conduct two sets of experiments. Firstly, we run standard 10-fold cross validations based on identical training and test data splits as reported in [Wong and Mooney, 2006], and the micro-averaged results are presented in this section. Secondly, following from [Jones et al., 2012], we reserve 280 sentences for test and train on the remaining 600.

For a test example, an output MR is considered correct if and only if the resulting query retrieves the same answer as the correct MR when submitted to the database. Performance was measured in terms of precision, recall and F-measure.

## 5.2 Overall Results

Two systems, with and without the extraction phase of relevant MR productions for test examples, were run over two different experimental setups to test the efficiency and accuracy of our model. The results, in term of test decoding time, precision, recall and F-measure are shown in Table 3. As we can see, the process of extracting relevant MR productions is shown to be quite effective. The use of extracting phase greatly improves the efficiency of decoding. Addtionally, we observe a consistent improvement in both precision and recall after employing the extracting phase for test examples. We also notice that the recall always equals to the precision, which means that, for every test example, an output MR tree can be generated by our systems. The likely reason is that our feature based discriminative structured prediction model can effortlessly allow smoothing over previously unseen MR productions.

[2] $n$ is set to be the number of words in the NL sentence in our experiments.

| | 10-fold cross-val | | | |
|---|---|---|---|---|
| | Time | Prec. | Rec. | F |
| *LVP* | 2:51 | 89.2 | 89.2 | 89.2 |
| *LVP+EXT* | 0:48 | 90.9 | 90.9 | 90.9 |
| | 600 train/280 test | | | |
| | Time | Prec. | Rec. | F |
| *LVP* | 0:42 | 85.0 | 85.0 | 85.0 |
| *LVP+EXT* | 0:08 | 87.5 | 87.5 | 87.5 |

Table 3: Results on two different experimental setups; columns are testing time (m:s), precision/recall/F-measure on test data. *LVP* denotes the system using all possible MR productions for decoding; *LVP+EXT* denotes the system employing the additional extraction phase of relevant MR productions.

### 5.3 Comparison with Other Models

In this section, we give a comparison between our model and other state-of-the-art learned models for semantic parsing. Among all the previous models, *WASP* [Wong and Mooney, 2006], *LU* [Lu et al., 2008] and *tsVB-hand* [Jones et al., 2012] are chosen, because they require the same amount of supervision as our system, and are directly comparable to our model. tsVB-hand represents the best performing version of the model based on tree transducer using hyper-parameters manually tuned on the training data.

Performance of our model and some of the best results from the state-of-the-art systems are summarized in Table 4, where the performance scores for the previous systems are taken from [Lu et al., 2008] and [Jones et al., 2012]. Some entries that are not available in other publications are indicated with a "–" symbol. Observing the results in Table 4, we can see that our model achieves significant performance improvement over those state-of-the-art systems on all metrics. In terms of F-measure, we gain a 5.7% absolute improvement, and a 6.5% absolute improvement over the best results on the two different experimental setups, which lead to an error reduction rate of 38.5% and 34.2%, respectively.

| | 10-fold cross-val | | | 600 train/280 test | | |
|---|---|---|---|---|---|---|
| | Prec. | Rec. | F | Prec. | Rec. | F. |
| *WASP* | 87.2 | 74.8 | 80.5 | 85.6 | 71.1 | 77.7 |
| *LU* | 89.3 | 81.5 | 85.2 | 85.7 | 76.8 | 81.0 |
| *tsVB-hand* | – | – | – | 79.3 | 79.3 | 79.3 |
| *LVP+EXT* | **90.9** | **90.9** | **90.9** | **87.5** | **87.5** | **87.5** |

Table 4: Performance comparison with other directly comparable systems over English corpus.

### 5.4 Performance on Other Languages

Since our learning algorithm does not use any natural language specific knowledge, it is directly applicable to other natural languages. We illustrate this by evaluating our system on the multilingual section of GeoQuery. The experimental results are shown in Table 5. These results used the same experimental setup as English, training on 600 examples, and testing on 280 examples. As we can see from Table 5, compared with state-of-the-art systems, our system achieves better performance on German and Greek, and attains comparable performance on Thai.

| | German | | | Greek | | | Thai | | |
|---|---|---|---|---|---|---|---|---|---|
| | Prec. | Rec. | F | Prec. | Rec. | F | Prec. | Rec. | F |
| *WASP* | **87.1** | 65.7 | 74.9 | **88.5** | 70.7 | 78.6 | 79.0 | 71.4 | 75.0 |
| *LU* | 76.4 | 62.1 | 68.5 | 80.8 | 69.3 | 74.6 | 80.1 | 73.6 | 76.7 |
| *tsVB-hand* | 74.6 | 74.6 | 74.6 | 75.4 | 75.4 | 75.4 | 78.2 | **78.2** | **78.2** |
| *LVP+EXT* | 78.6 | **78.6** | **78.6** | 80.3 | **80.3** | **80.3** | 76.4 | 76.4 | 76.4 |

Table 5: Performance comparison among models on the multilingual section of GeoQuery.

## 6 Conclusion and Future Work

In this paper, we have presented a discriminative structured prediction approach to semantic parsing using the hybrid tree as the latent structure variable. Furthermore, we design an efficient approach based on vector space model to extract relevant MR productions for test examples, improving the efficiency and accuracy of inference. The experimental results show that our approach performs substantially better than previous systems.

In future, we plan to improve the decoding algorithm by employing cubing pruning algorithm to allow the use of more effective non-local features, which may result in greater performance gain. Additionally, it will be very interesting to investigate applying other structure as the latent variable in the proposed model.

## Acknowledgments

This research is supported by projects 61073119, 61272221 under the National Natural Science Foundation of China, project BK2010547 under the Jiangsu Natural Science Foundation of China and project 12YYA002 under the Jiangsu Social Science Foundation of China. We would also like to thank the excellent and insightful comments from the three anonymous reviewers.

## References

[Carreras and Marquez, 2004] Xavier Carreras and Luis Marquez. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of the Eighth*

*Conference on Computational Natural Language Learning (CoNLL)*, pages 89-97, 2004.

[Collins, 2002] Michael Collins. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceeding of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2002.

[Crammer et al., 2005] Koby Crammer, Ryan McDonald, and Fernando Pereira. Scalable large-margin online learning for structured classification. In *NIPS Workshop on Learning With Structured Outputs*, 2005.

[Ge and Mooney, 2005] Ruifang Ge and Raymond J. Mooney. A statistical semantic parser that integrates syntax and semantics. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL)*, pages 9-16, 2005.

[Ge and Mooney, 2006] R. Ge and R. J. Mooney. Discriminative reranking for semantic parsing. In Proce*edings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL)*, pages 263-270, 2006.

[Ge and Mooney, 2009] Ruifang Ge and Raymond J. Mooney. Learning a compositional semantic parser using an existing syntactic parser. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP)*, pages. 611-619, 2009.

[Jones et al., 2012] Bevan Keeley Jones, Mark Johnson and Sharon Goldwater. Semantic Parsing with Bayesian Tree Transducers. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 488-496, 2012.

[Kate and Mooney, 2006] R. J. Kate and R. J. Mooney. Using string-kernels for learning semantic parsers. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL)*, pages 913-920, 2006.

[Kate et al., 2005] R. J. Kate, Y. W. Wong, and R. J. Mooney. Learning to transform natural to formal languages. In *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI)*, pages 1062-1068, 2005.

[Kwiatkowski et al., 2010] Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater and Mark Steedman. Inducing probabilistic CCG grammars from logical form with higher-order unification. In *Proceeding of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1223-1233, 2010.

[Lee et al., 1997] D.L. Lee, H. Chuang and K. Seamons. Document Ranking and the Vector-Space Model. *IEEE Software*, 14(2): 67-75, 1997.

[Lu et al., 2008] Wei Lu, Hwee Tou Ng, Wee Sun Lee and Luke S. Zettlemoyer. A Generative Model for Parsing Natural Language to Meaning Representations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 913-920, 2008.

[McDonald, 2006] Ryan McDonald. Discriminative Training and Spanning Tree Algorithms for Dependency Parsing. University of Pennsylvania, *PhD Thesis*, 2006.

[Sun et al., 2009] Xu Sun, Takuya Matsuzaki, Daisuke Okanohara Jun'ichi Tsujii. Latent Variable Perceptron Algorithm for Structured Classification. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1236-1242, 2009.

[Wong and Mooney, 2006] Y. W. Wong and R. J. Mooney. Learning for semantic parsing with statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, pages 439-446, 2006.

[Wong and Mooney, 2007] Y. W. Wong. Learning for Semantic Parsing and Natural Language Generation Using Statistical Machine Translation Techniques. *Ph.D. thesis*, The University of Texas at Austin.

[Yair Censor, 1997] Stavros A. Zenios Yair Censor. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, 1997.

[Zettlemoyer and Collins, 2007] L. S. Zettlemoyer and M. Collins. Online learning of relaxed CCG grammars for parsing to logical form. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 678-687, 2007

[Zhang and Dennis, 1989] Kaizhong Zhang and Shasha Dennis. Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing*, 18(6):1245-1262, 1989.