Computing and Informatics, Vol. 34, 2015, 1257–1288

WORD SENSE DISAMBIGUATION: A STRUCTURED LEARNING PERSPECTIVE

Yun Zhou, Ting Wang*

School of Computer
National University of Defense Technology
Changsha 410073, China
e-mail: {zy_vxd, tingwang}@nudt.edu.cn

Zhiyuan Wang

State Key Laboratory of High Performance Computing
National University of Defense Technology
&
School of Computer
National University of Defense Technology
Changsha 410073, China
e-mail: wzy@nudt.edu.cn

Abstract. This paper explores the application of structured learning methods (SLMs) to word sense disambiguation (WSD). On one hand, the semantic dependencies between polysemous words in the sentence can be encoded in SLMs. On the other hand, SLMs obtained significant achievements in natural language processing, and so it is a natural idea to apply them to WSD. However, there are many theoretical and practical problems when SLMs are applied to WSD, due to characteristics of WSD. Beginning with the method based on hidden Markov model, this paper proposes for the first time a comprehensive and unified solution for WSD based on maximum entropy Markov model, conditional random field and tree-structured conditional random field, and reduces the time complexity and running time of the proposed methods to a reasonable level by beam search, approximate training, and parallel training. The update of models brings performance improvement, the introduction of one step dependency improves performance by 1–5 percent, the

* corresponding author

adoption of non-independent features improves performance by 2–3 percent, and the extension of underlying structure to dependency parsing tree improves performance by about 1 percent. On the English all-words WSD dataset of Senseval-2004, the method based on tree-structured conditional random field outperforms the best attendee system significantly. Nevertheless, almost all machine learning methods suffer from data sparseness due to the scarcity of sense tagged data, and so do SLMs. Besides improving structured learning methods according to the characteristics of WSD, another approach to improve disambiguation performance is to mine disambiguation knowledge from all kinds of sources, such as Wikipedia, parallel corpus, and to alleviate knowledge acquisition bottleneck of WSD.

Keywords: Word sense disambiguation, structured learning, hidden Markov model, conditional random field, parallelization, approximate training

Mathematics Subject Classification 2010: 68T50, 91F20

1 INTRODUCTION

Word sense disambiguation (WSD) is to determine the sense of polysemous words given in the context. WSD is regarded as the intermediate task of natural language processing (NLP), and plays a very important role in many applications, such as information retrieval, machine translation, semantic web and bioinformatics, cf. [1, 2]. For example, given query "bar", should the search engine return results about barroom, a piece of material, barrier, or anything else? It depends on the sense of "bar", and actually there are 13 senses for noun "bar" in WordNet[1]. In general, humans can easily recognize the sense of polysemous words in the context, but the difficulty of automatic word sense disambiguation is far beyond the estimation of people, even at present.

Polysemy is universal in natural language. For example, the noun "bar" may appear in "at the bar", "iron bar", "hit the bar", and so on, where the first refers to a room where alcoholic drinks are served, the second refers to a rigid piece of metal or wood, and the third refers to obstruction of the goal. The lemmas of polysemous words[2] account for 17.7 % of all lemmas in WordNet [3], while the occurrences of polysemous words accounts for 72.8 % of all occurrences in Brown Corpus [4]. It means that the more polysemous the word is, the more frequently it occurs in the real corpus, and over 70 % words need to be disambiguated [5].

At present, WSD systems usually label words independently in the sentence, without regard to the dependency among labels of different words. Given a sentence,

The man saw me looking at the iron bars.

[1] We use WordNet 1.7.1 in the paper.

[2] Refer to content words, i.e. noun, verb, adjective and adverb.

There are five content words, i.e. "man", "see", "look", "iron", "bar", to be disambiguated. Since the above five words have different label spaces, the state-of-the-art WSD systems usually build a classifier for each word. From the perspective of point-wise classifier, the sentence is divided into words.

The | **man** | **see** | me | **look** | at | the | **iron** | **bar** |.

As usual, all inflections of words are removed to relieve sparseness. The words in the sentence are disambiguated independently. If we want to disambiguate all the words in arbitrary text, thousands of classifiers are needed. Due to the computing power of modern computers, the number of classifiers does not constitute a challenge for computation. However, this point-wise paradigm does not take into account the label[3] dependency between words. For the example sentence, when point-wise WSD system disambiguates the word "iron", it cannot deal with the label of word "bar" simultaneously, while it is closely related to that of "iron". Similarly, the label of word "man" is also related to the label of word "see". In addition, the words that are not adjacent may also have label dependency, such as the word "look" and "bar", since "bar" is the object of "look".

In order to encode the label dependency between words, we resort to the structured learning methods (SLMs) [6] in this paper. SLMs are machine learning methods to predict structure consisting of multiple variables, such as sequence, tree or general graph. Meanwhile, the ordinary machine learning methods, such as naïve Bayes, support vector machine, and logistic regression, predict only one variable, called point-wise learning methods in this paper. SLMs obtained significant achievements in most subfields of NLP, such as Chinese word segmentation [7], POS tagging [8, 9], morphosyntactic tagging [10], chunking [11, 12], named entity recognition [13, 14], semantic role labeling [15], parsing [16] and dialogue act recognition [17]. However, SLMs seldom show their power in WSD, due to theoretical and practical obstacles when SLMs are applied to WSD.

The main idea of this paper is as follows. We first construct a graph for the sentence, where the vertex is the word in the sentence, and the edge represents the label dependency between words. The graph can be a simple one, such as the natural sequence of the sentence, or a more complicated one, such as the dependency parsing tree of the sentence. Then, we learn the model on the graph using various SLMs. We develop the models as well as features in an incremental fashion to better reflect the label dependency among words. The main contributions of this paper lie in:

1. Proposing a comprehensive and unified solution for WSD based on SLMs, including hidden Markov model (HMM), maximum entropy Markov model (MEMM), conditional random field (CRF) and tree-structured conditional random field (TCRF).
2. Applying a beam search to the methods based on HMM and MEMM, which reduces the time complexity of prediction from $O(TN^2)$ to $O(TR^2)$, where N is

[3] In the context of WSD, the label of the word refers to the sense of the word.

the number of different senses in the dictionary, at least tens of thousands, and R is the maximum number of senses of a word, about dozens.

3. Proposing a smoothing strategy to alleviate data sparseness in HMM and MEMM.
4. Proposing approximate training for CRF, which reduces the time complexity of training of CRF from $O(mLTN^2)$ to $O(mLTR^2)$.
5. Parallelizing training of CRF to reduce actual running time.
6. Generalizing disambiguation structure from line chain to dependency parsing tree.

2 WORD SENSE DISAMBIGUATION

The history of automatic WSD is almost as long as that of the computer. After over 60 years of research, WSD is still an open problem [1, 2].

2.1 Basics of Word Sense Disambiguation

The aim of automatic WSD is to enable computers, instead of humans, to disambiguate word sense automatically. The formal definition of WSD is, given dictionary D and a text T consisting of some words $(w_1, w_2, \ldots, w_n)$, WSD system determines senses of all words or some words in the text T, i.e. determines a mapping A from word to sense satisfying $A(i) \subseteq Sense_D(w_i)$, where $Sense_D(w_i)$ is the set consisting of all senses of word w_i in dictionary D, and $A(i)$ is the set consisting of proper senses of word w_i in given context. Mapping A can assign more than one sense for word w_i, although typically only one sense is assigned. This definition does not mention the part-of-speech (POS) of the target word, but the POS is usually provided before WSD. POS tagging is closely related to WSD, and POS tagging is a well-studied problem with accuracy over 95 %, thus the separation of POS tagging from WSD can fully expose the hardest core of WSD.

Evaluation is an important part of WSD research. Senseval/Semeval [18], starting from 1998, is the de facto standard evaluation in WSD field, which provides consistent dataset, dictionary and evaluation measure for comparing different WSD systems. Senseval/Semeval includes two types of WSD evaluation. One is lexical sample WSD, in which the system is required to disambiguate some words in a given text, and typically one sentence only contains one target word. The other is all-words WSD, in which the system is required to disambiguate all open words, i.e. noun, verb, adjective and adverb, in the text. The evaluation measures include coverage C, precision P, recall R and F_1 value, in which the most important measures are recall R and F_1 value.

Dictionary is the basis of WSD, and Senseval/Semeval adopts WordNet [3] in usual. In WSD, the candidate sense sets of different target words are usually different, and WordNet is often criticized to be too fine-grained for WSD, so it is very

difficult to label the corpus, even by humans. Inter-annotator agreement (IAA) is the measure to evaluate the quality of sense labeled corpus. For sense-labeled corpus with WordNet as the dictionary, IAA is usually between 0.67 and 0.80 [19, 20, 21]. Since the level of automatic WSD is much lower than human, IAA is usually regarded as the upper bound of WSD system, while the simplest methods, such as most frequent sense (MFS), are usually regarded as lower bound of WSD system.

2.2 Problem in Word Sense Disambiguation

WSD can be transformed to the classification problem in machine learning. In general classification problem, the candidate label sets of different targets are the same. However, the candidate sense set changes with target word in WSD, which results in specific difficulties. For example, according WordNet, the word "man" has two POSs, i.e. noun and verb. The noun "man" has 11 senses, and the verb "man" has 2 senses, as shown in Table 1. The senses of word "see" are shown in Table 2.

Man
Noun
1. an adult male person
2. a member of a military force
3. the generic use of the word to refer to any human being
...
Verb
1. take charge of a certain job
2. provide with men

Table 1. Senses of word "Man"

See
Noun
1. the seat within a bishop's diocese where his cathedral is located
Verb
1. perceive by sight or have the power to perceive by sight
2. perceive mentally, as of an idea
3. perceive or be contemporaneous with
...
Adverb
1. compare, used in texts to point the reader to another location

Table 2. Senses of word "See"

There are totally 111 223[4] senses[5] in WordNet. If viewed as a classification problem, WSD is to the determine the right sense for each word from 111 223 senses,

[4] WordNet 1.7.1.

[5] WordNet is organized by synonym set, synset for short, which is equivalent to sense.

which constitutes a challenge for any classifier. Thus, the mainstream WSD systems have to divide the sense space, and disambiguate each word independently. For the sentence "The man saw me looking at the iron bars .", we should build five classifiers for "man", "see", "look", "iron" and "bar", respectively, as shown in Figure 1.

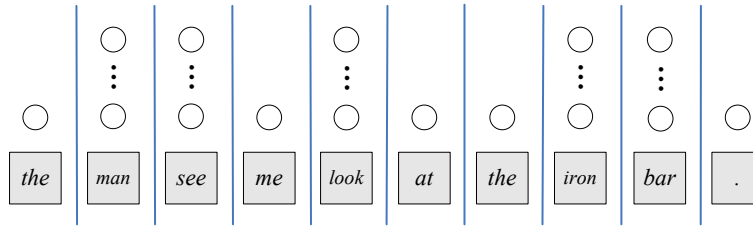

Figure 1. Point-wise classification

Figure 1 applies to any point-wise labeling problem in natural language, of course including WSD. The square represents observation, such as word in the sentence, and the circle represents label, such as sense, POS or anything else. The sense dictionary, such as WordNet, usually does not contain non-content words, but we also assign a pseudo sense to every non-content word for the sake of unified representation. Although the division of sense space results in feasibility of point-wise classification methods, it sacrifices the label dependency among words, which embodies the fluency of semantic expression and helps disambiguation.

For general NLP tasks other than WSD, researchers usually employ SLMs to encode the label dependency. Now we consider to POS tag words in the above sentence. The candidate POS set of "The" is $\{pronoun\}$, the set of "man" is $\{noun, verb\}$, the set of "see" is $\{noun, verb, adverb\}$, and so on. Although the candidate POS set also changes with target word, the universal POS set is not large, typically including tens of POSs. Therefore, we can take universal POS set as the candidate label set of all words in the sentence to maintain the homogeneity of label space, which is required by most unified statistical learning methods, such as HMM and CRF. The sequence classification are shown in Figure 2.

In Figure 2, the label dependencies are encoded directly in the model, which makes SLMs the state-of-the-art in many NLP tasks, such as POS tagging, named entity recognition and chunking. The success of SLMs is partly due to the relatively small number of labels in the domain. For example, POS tagging has dozens of labels, the named entity recognition has at most dozens of labels, and the BIO-fashioned chunking has only three labels.

Unfortunately, WSD has a large number of labels. As mentioned before, WordNet contains 111 223 senses, and even in SemCor [22] there are 25 846 different senses. Thus, the universal label set in Figure 2 contains at least tens of thousands elements. The large universal set results in extremely high complexity both in time and space, which is far beyond the capacity of modern computers. In addition, the large universal set causes serious data sparseness, which is not wanted

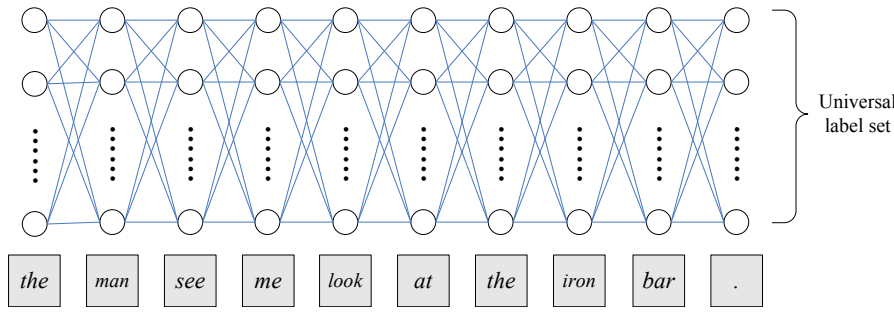

Figure 2. Sequence classification

in machine learning. Thus, SLMs are not directly available for WSD, and this paper explores the WSD methods based on SLMs, especially probabilistic graphical models.

3 RELATED WORK

Until now, there is little research on semantic disambiguation involving SLMs. Before the emergence of Senseval/Semeval, several scholars used HMM to disambiguate words. For example, Segond et al. [23] uses HMM to disambiguate semantic tags of WordNet, while WSD traditionally adopts senses, instead of semantics tags, as inventory, and thus [23] is not WSD in a usual sense. In addition, there are only 45 semantic tags, which are several orders of magnitude less than senses. Loupy et al. [24] disambiguates word sense using the linear combination of naïve Bayes model and first order HMM. Training dataset and test dataset are 95 % and 5 % of SemCor, respectively. Because above mentioned researches do not employ standard settings of WSD, their results are not comparable.

The system LIA-Sinequa-AllWords developed by Crestan et al. [25] attends the EAW of Senseval-2001, with recall 0.618 (ranked No. 3/22). This work continues the research of Loupy et al. [24], and method is divided into two steps. The first step labels words with semantic tags in WordNet according to lemma, using first order HMM and second order HMM, and the second step labels words with synsets in WordNet according to lemma and semantic tag obtained in first step, using naïve bayes model (zero order HMM). The experimental results show that the labeling with two steps does not improve the WSD performance compared with our directly modeled HMM, cf. Section 5 for the model and Section 9.2 for the experimental results.

Molina et al. [26, 27] proposes another method based on HMM to disambiguate word sense, the system [26] is evaluated on the data of EAW of Senseval-2001, with recall 0.602, and the system [27] attends EAW of Senseval-2004, with recall 0.609 (ranked No. 7/26). Molina first uses lex_sense as label to reduce the search space, and then maps the lex_sense to synset to accomplish WSD. In WordNet, there is one-to-

one correspondence between synset and sense_key, whose form is lemma%lex_sense, such as interest%1:09:00, where lex_sense is composed of several numbers separated by colons. Every sense of lemma corresponds to a lex_sense, but lex_sense of different words can be identical. The number of lex_sense is much smaller than synset, and thus the search space is reduced significantly. Lex_sense of word is obtained by prediction of HMM, then lemma and lex_sense is concatenated to get sense_key, and finally sense_key is mapped to synset. However, the commonness between same lex_sense of different lemmas is rather coincidental, without any linguistic support, and thus the performance of this method is only comparable with baseline system MFS.

Deschacht and Moens [28] exploits hypernym hierarchy in WordNet to disambiguate word sense using CRF. Since there are only several labels in each level, the label space is reduced efficiently. But this method is only applicable to noun and verb, because adjective and adverb have no hypernym. Duan et al. [29] labels word with two sememes in HowNet [30] using two-layered CRF, and two sememes are concatenated to get sense of the word. There are about 1 500 sememes in HowNet, which is much fewer than number of senses in HowNet, tens of thousands, and so label space is reduced. But, this method exploits the particular structure of HowNet, which is only applicable to Chinese, and thus this method cannot be generalized to other languages. Reichartz and Paass [31] labels noun and verb with 45 semantic tags in WordNet using CRF. Similar to [23], this method is not WSD in usual sense. Hatori et al. [32] builds the feature-forest model, which they claim to have the same power as CRF, for WSD, and the model is trained by a maximum entropy estimator [33, 34]. This method is evaluated on the data of EAW of Senseval-2004, with recall 0.655.

In a word, SLMs are not fully explored for WSD, and this paper tries to make some useful attempts in this field.

4 STRUCTURED LEARNING METHODS

Machine learning methods, referring to classification methods exclusively in this paper, are methods to learn a function $f : X \to Y$, which maps the element x of domain X to the element y of codomain Y. Codomain of point-wise learning methods can be -1, 1 (binary classification), multinomial value (multinomial classification) or real value (regression). Meanwhile, for SLMs, the element y in codomain Y has internal structure, such as linear chain, tree or general graph. SLMs can be divided into three kinds, the first kind of SLMs includes those based on probabilistic graphical models, such as Hidden Markov Model (HMM) [35], Maximum Entropy Markov Model (MEMM) [36], and Conditional Random Field (CRF) [8]; the second includes methods based on maximum margin, such as Maximum Margin Markov Network (M^3N) [37] and Support Vector Machine for Interdependent and Structured Output Spaces (SVM^{Struct}) [16]; the third includes other methods, such as Structured Perceptron (SP) [11] and Search-Based Structured Prediction (SEARN) [13].

Natural language is structured, while SLMs can reflect the structure in language directly by graph. Thus, there is a large number of works on NLP tasks using SLMs, and new developments in specific domains arise frequently, therefore an exhaustive survey is unlikely to be presented in this paper. Besides the classic works on NLP tasks using SLMs, we present some recent advances on this regard from two aspects.

The former line of research focuses on the improvement of the model while validating on NLP tasks. Gimpel and Smith [38] proposes softmax-margin CRF by incorporating a task specific cost function into CRF, which highlights the connections between CRF and max-margin learning, and their results are validated on named entity recognition. Berg-Kirkpatrick et al. [39] add features to generative model by turning the component multinomial into a miniature logistic regression model, which overcomes the shortcoming that generative SLMs can only exploit multinomial observations, and the results show that the proposed model outperforms significantly its basic counterparts in POS tagging, grammar induction, word alignment and word segmentation. Gimpel et al. [40] combines distributed computing and rapidly converging online algorithm to speedup SMLs for NLP, which is promising in large-scale NLP tasks. The latter line of research focuses on the application of structured learning methods to emerging NLP tasks, especially in social media. Social media and user-created web content are producing enormous quantities of text in electronic form. SLMs are widely used in NLP tasks related to social media, such as Twitter retrieval [41], POS tagging in Twitter [42], user profiling [43] and name disambiguation [44]. For the most recent literature on NLP tasks using structured learning, interested readers can refer to [45, 46].

4.1 Basics of Probabilistic Graphical Models

This paper focuses on application of SLMs to WSD. Specifically, they are HMM, MEMM, CRF and TCRF, and these models belong to the first category of SLMs, so we quickly go through the basics of probabilistic graphical models.

Probabilistic graphical models combine graph theory with probability theory, describe the dependency among variables, and define the decomposition of the joint probability of multiple variables in the graph. Probabilistic graphical model can be represented by $G = (V, E)$, where V is the set of all vertices in the graph, $E \subseteq V \times V$ is the set of all edges in the graph. Every vertex i in V is a random variable v_i, and every edge j in E represents the dependency between the two vertices connected by that edge. Probabilistic graphical model is typically divided into directed graphical model and undirected graphical model. Directed graphical model decomposes the joint probability of all variables into product of local conditional probabilities.

$$p(v) = \prod_{i \in V} p(v_i | v_{\pi(i)}), \tag{1}$$

where v represents all variables in V, $\pi(i)$ is the set consisting of parent nodes of i. Undirected graphical model decomposes the joint probability of all variables into product of potential functions on cliques.

$$p(v) = \frac{1}{Z} \prod_k \psi_k[C_k(v)], \tag{2}$$

where $C_k(v)$ is the clique in the undirected graphical model, ψ_k is the potential function on clique $C_k(v)$, Z is the normalization factor. Clique is the fully connected subgraph in the undirected graph model, and is typically maximal in the sense that there are no other variables whose inclusion preserves its fully connected property. Potential function is a non-negative real value function on the clique. HMM and MEMM belong to directed graphical model, while CRF and TCRF belong to undirected graphical model, and these models will be tailored for WSD in the following paper.

5 DISAMBIGUATION METHOD BASED ON HIDDEN MARKOV MODEL

5.1 Model

HMM [35] is a directed graphical model, as shown in Figure 3.

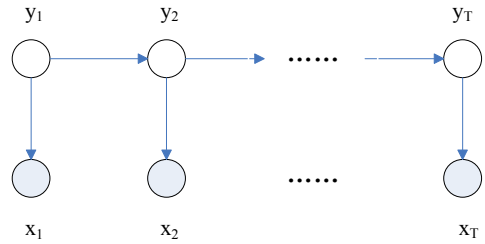

Figure 3. Directed graphical model for hidden Markov model

In directed graphical model, circle represents random variable, where circle without shadow is unobservable random variable, and circle with shadow is observable random variable. Edge represents the dependency between two random variables. According to Equation (1), joint probability of HMM can be decomposed as

$$p(x_1, \ldots, x_T, y_1, \ldots, y_T) = p(y_1)p(x_1|y_1)p(y_2|y_1)p(x_2|y_2) \ldots p(y_T|y_{T-1})p(x_T|y_T), \tag{3}$$

where x_t represents the observation of t^{th} word, and y_t represents the label of t^{th} word in the sentence. Usually, HMM can only employ multinomial observation, and there are several options, such as lemma, POS or the concatenation of lemma and POS. By preliminary experiments, we adopt the concatenation of lemma and POS,

since lemma and POS are the two most important features for WSD. There are also options for the label of word, such as semantic tags in the WordNet [23, 31], lex_sense [26, 27]. However, semantic tags is too coarse for WSD, and modern WSD systems are usually evaluated with WordNet, so we do not adopt semantic tags as the label. As mentioned in Section 3, the commonness between same lex_sense of different lemmas is rather coincidental, without any linguistic support, and so we do not adopt lex_sense either. Finally, we use sense(synset) in WordNet as the label of word. We train the HMM by supervised learning, i.e. calculating the probabilities in HMM by maximum likelihood estimation from the training corpus.

In general, the label spaces of different variables in HMM should be identical, so we have to merge all senses of all words into a universal sense set. Unfortunately, the universal sense set contains at least tens of thousands elements, which results in extremely high time complexity in prediction of HMM. The prediction of HMM is usually done with Viterbi algorithm, whose time complexity is $O(TN^2)$, where T is the length of the sentence, and N is the number of labels. Now N is at least tens of thousands, so the time complexity is far beyond the capacity of modern computers. We exploit the sparseness of the label space to reduce the time complexity as follows.

5.2 Beam Search

The prediction of HMM can be formulated as

$$\widehat{y_1}, \ldots, \widehat{y_T} = \underset{y_1,\ldots,y_T}{\operatorname{argmax}}\ p(y_1, \ldots, y_T | x_1, \ldots, x_T) = \underset{y_1,\ldots,y_T}{\operatorname{argmax}}\ p(x_1, \ldots, x_T, y_1, \ldots, y_T). \tag{4}$$

The idea of Viterbi algorithm is to find the most probable path in the label space, and the example label space is shown in Figure 2. The high time complexity comes from the large universal sense set. However, the label space is sparse, since a word has at most tens of senses, we can only search those actual senses of the word, without regard to the unrelated ones, which results in beam search Viterbi algorithm. We redraw the label space of beam search in Figure 4.

In Figure 4, the black block represents the actual senses of corresponding word, and we assign a pseudo sense to each non-content word to maintain the homogeneity of the label space, which is denoted by a black line. The beam search Viterbi algorithm is shown in Table 3.

The input of the algorithm contains two parts, one is the observation sequence, and the other is the parameters of HMM, i.e. S, V, A, B, Π, where

- $S = \{s_1, \ldots, s_N\}$ is the universal sense set,
- $V = \{v_1, \ldots, v_M\}$ is the set consisting of all lemma+POS in the training corpus,
- $A = \{a_{ij}\}$, $a_{ij} = p(y_{t+1} = s_j | y_t = s_i)$, $1 \leq i, j \leq N$ is the matrix of sense transition probability,
- $B = \{b_j(k)\}$, $b_j(k) = p(x_t = v_k | y_t = s_j)$, $1 \leq j \leq N$, $1 \leq k \leq M$ is the matrix of sense-lemma+POS emission probability, and

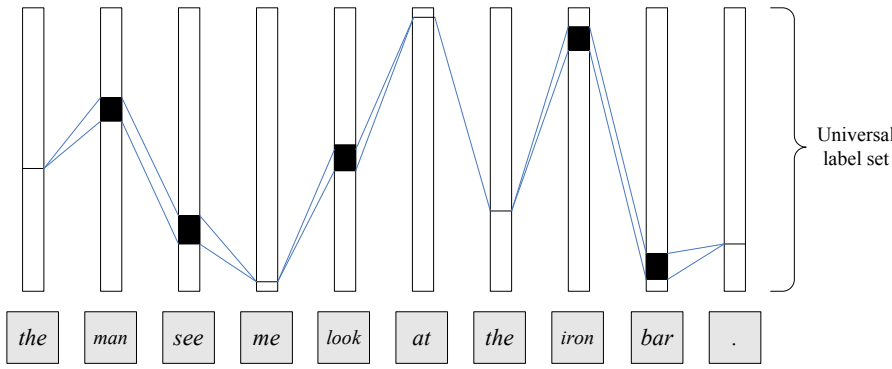

Figure 4. Beam search

Input:
Observation sequence $x_1, \ldots, x_T$
Parameters S, V, A, B, Π
Output:
Optimal label sequence $\widehat{y_1}, \ldots, \widehat{y_T}$
Algorithm:
01: FOR i FROM 1 TO N that satisfies $s_i \in S(x_1)$
02: $\delta_1(i) = \pi_i b_i(x_1)$
03: $\psi_1(i) = 0$
04: FOR t FROM 2 TO T
05: FOR j FROM 1 TO N that satisfies $s_j \in S(x_t)$
06: $\delta_t(j) = \max_{1 \le i \le N, s_i \in S(x_{t-1})}[\delta_{t-1}(i)a_{ij}]b_j(x_t)$
07: $\psi_t(j) = \operatorname{argmax}_{1 \le i \le N, s_i \in S(x_{t-1})}[\delta_{t-1}(i)a_{ij}]$
08: $\widehat{y_T} = \operatorname{argmax}_{1 \le i \le N, s_i \in S(x_T)}[\delta_T(i)]$
09: FOR t FROM $T-1$ TO 1
10: $\widehat{y_t} = \psi_{t+1}(\widehat{y_{t+1}})$
11: RETURN $\widehat{y_1}, \ldots, \widehat{y_T}$

Table 3. Beam search Viterbi algorithm for word sense disambiguation

- $\Pi = \{\pi_i\}$, $\pi_i = p(y_1 = s_i)$, $1 \le i \le N$ is the vector of initial sense probability. The output of the algorithm is the optimal sense sequence.

$\delta_t(i)$ is the forward local optimal value for label i at step t, $\psi_t(i)$ is backtracking pointer for label i at step t, and $S(x_t)$ represents the set of actual senses of observation x_t. Lines 01–03 initialize the forward local optimal value $\delta_1(i)$ and backtracking pointer $\psi_1(i)$. Lines 04–07 calculate $\delta_t(i)$ recursively from beginning to end, and the search space is restricted in the possible senses of the corresponding word. Line 08 gets the optimal sense of last word. Lines 09–10 get all other optimal senses by backtracking. The time complexity of Viterbi algorithm is dominated by Lines 04–07, and it is $O(TN^2)$ in general. After adopting beam search, it bcecomes $O(TR^2)$,

where $R = \max_{x_t \in V} |S(x_t)|$, the number of senses of the most polysemous word, is about dozens, while N is at least tens of thousands.

In speech recognition field, similar techniques [47, 48] have been used to keep only a subset of promising candidates at every step, instead of retaining all candidates. However, this is the first time that beam search is applied in WSD explicitly.

5.3 Smoothing Strategy

In the HMM for WSD, the matrix of sense transition probability and the matrix of sense-word emission probability are both sparse. Viterbi algorithm calculates recursively local optimal value $\delta_t(j)$ according to Line 06 in Table 3, and $\delta_t(j)$ is the product of a number of transition probabilities and emission probabilities. If any of these probabilities is zero, then $\delta_t(i)$ is zero, which makes the path impossible.

After beam search is applied, emission probability is not zero any more, but transition probability still could be zero. That is to say, beam search reduces the search space to a denser one, but the "beam" itself is still sparse. Now, we smooth the reduced search space – beam, instead of reducing it further. If the bisense $s_i s_j$ does not occur in training set, due to data sparseness of training data, i.e. $a_{ij} = 0$ by maximum likelihood estimation, then we assign a_{ij} a very small value using smoothing strategy as follows.

$$a_{ij}^* = \begin{cases} \gamma \frac{C(s_i s_j)}{\sum_{s:s\in S(x_t), C(s_i s)>0} C(s_i s)} & \text{if } s_i \in S(x_{t-1}),\ s_j \in S(x_t) \text{ and } C(s_i s_j) > 0, \\ (1-\gamma) \frac{F(s_j)}{\sum_{s:s\in S(x_t), C(s_i s)=0} F(s)} & \text{if } s_i \in S(x_{t-1}),\ s_j \in S(x_t) \text{ and } C(s_i s_j) = 0, \\ 0 & \text{if } s_i \notin S(x_{t-1}) \text{ or } s_j \notin S(x_t), \end{cases} \tag{5}$$

where $C(s_i s_j)$ represents the number of occurrences of $s_i s_j$ in the training set, and $F(s_j)$ represents the frequency of sense in dictionary, such as WordNet. This smoothing strategy is a kind of back-off smoothing, which is similar to Katz's back-off [49]. When the bisense does not occur in the training corpus, we resort to the unisense model. However, rather than using the unigram model from original corpus as in Katz's back-off, we use the statistics in the dictionary, such as WordNet, to better reflect the distribution of senses.

When $s_i \notin S(x_{t-1})$ or $s_j \notin S(x_t)$, the transition probability a_{ij}^* is not used by beam search, and so it is set to be zero. When $s_i \in S(x_{t-1})$ and $s_j \in S(x_t)$, there are two cases, one is that $s_i s_j$ does occur in the training corpus, i.e. $C(s_i s_j) > 0$, and the other is that $s_i s_j$ does not occur in the training corpus, i.e. $C(s_i s_j) = 0$. We normalize $C(s_i s_j)$ with statistics from train corpus for the former case, and $F(s_j)$ with frequency in WordNet for the latter case. All a_{ij}^* should sum up to one, and so we introduce an empirical parameter γ to balance the importance of the model from training corpus and the back-off model from WordNet. The former reflects the real sense transition, so γ should be a real number close to 1. In preliminary experiments, we try 0.9, 0.95, 0.99, 0.999 and find that the final result is not sensitive to γ. Finally, we set $\gamma = 0.999$ in the experiments.

6 DISAMBIGUATION METHOD BASED ON MAXIMUM ENTROPY MARKOV MODEL

The disambiguation method based on HMM embodies the one step dependency between senses of consecutive words, but the observation of HMM can only be multinomial value, for the example sentence, its observation sequence is

the_DT man_NN see_VB me_PRP look_VB at_IN the_DT iron_NN bar_NN ._PUNC

However, overlapping non-independent observations (features), such as local collocation and bag-of-words, are very important for WSD [50, 51, 52]. MEMM [36] compensates the disadvantage of HMM in this regard, and so we propose the disambiguation method based on MEMM.

6.1 Model and Solution

HMM embodies dependency between labels, maximum entropy model (ME) [53] can exploit non-independent features, while MEMM combines the merits of HMM and ME. MEMM employs ME to represent the conditional probability of current label given current observation and previous label, i.e. $p_{y_{t-1}}(y_t|x_t)$, where observation x_t is the feature vector, which can include non-independent features. MEMM can be treated as directed graphical model, as shown in Figure 5.

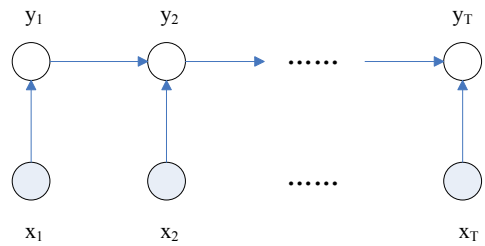

Figure 5. Directed graphical model for maximum entropy Markov model

According to Equation (1), the joint probability of MEMM can be decomposed into

$$p(x_1, \ldots, x_T, y_1, \ldots, y_T) = p(x_1)p(y_1|x_1)p(x_2)p(y_2|x_2, y_1) \ldots p(x_T)p(y_T|x_T, y_{T-1}). \quad (6)$$

MEMM for WSD includes four parameters S, V, M, Π, where

- $S = \{s_1, \ldots, s_N\}$ is the value set of label y_t, i.e. universal sense set in WSD,
- V is the vector space of observation (feature vector) x_t, and the features are defined in Section 6.2,

- $M = \{p_{y_{t-1}}(y_t|x_t) : y_{t-1} \in S\}$ is the set of ME models, and
- $\Pi = p_{Begin}(y_1|x_1)$ is the initial ME model.

ME model is formalized as

$$p_{y_{t-1}}(y_t|x_t) = \frac{1}{Z(x_t)} \exp\left\{\sum_i \theta_i f_i(y_t, x_t)\right\}, \tag{7}$$

where $Z(x_t)$ is the local normalization factor for step t, and $f_i(y_t, x_t)$ is the local feature, and θ_i is the weight of $f_i(y_t, x_t)$. Different from HMM, MEMM is a sort of machine learning framework, instead of a unified model. The joint probability is the product of multiple separate ME models, and each ME model is trained independently. ME model $p_{y_{t-1}}(y_t|x_t)$ is obtained by first collecting all y_t, x_t (label-observation pairs) immediately after label y_{t-1} and then training with learning algorithm of ME, such general iterative scaling [54]. The prediction of MEMM is formalized as

$$\begin{aligned}
\widehat{y_1}, \ldots, \widehat{y_T} &= \operatorname*{argmax}_{y_1,\ldots,y_T} p(y_1, \ldots, y_T|x_1, \ldots, x_T) = \operatorname*{argmax}_{y_1,\ldots,y_T} p(x_1, \ldots, x_T, y_1, \ldots, y_T) \\
&= \operatorname*{argmax}_{y_1,\ldots,y_T} p(y_1|x_1)p(y_2|x_2, y_1)\ldots p(y_T|x_T, y_{T-1}) = \operatorname*{argmax}_{y_1,\ldots,y_T} \prod_t p_{y_{t-1}}(y_t|x_t) \\
&= \operatorname*{argmax}_{y_1,\ldots,y_T} \prod_t \frac{1}{Z(x_t)} \exp\left\{\sum_i \theta_i f_i(y_t, x_t)\right\}.
\end{aligned} \tag{8}$$

The prediction of MEMM can be solved by Viterbi algorithm, so the search space of MEMM is similar to that of HMM, cf. Figure 2, which also results in high time complexity. This is again solved by beam search, which is similar to that of HMM, cf. Section 5.2. The details are omitted here. Moreover, data sparseness also exists in disambiguation method based on MEMM, since the ME model cannot calculate $p_{y_{t-1}}(y_t|x_t)$ if y_t has never occurred immediately after y_{t-1} in the training corpus. This problem is also solved by smoothing strategy similar to HMM, and its equation is presented as follows.

$$p^*_{s_i}(s_j|x_t) = \begin{cases}
\gamma \frac{p_{s_i}(s_j|x_t)}{\sum_{s:s\in S(x_t), C(s_i s)>0} p_s(s_j|x_t)} & \text{if } s_i \in S(x_{t-1}),\ s_j \in S(x_t) \text{ and } C(s_i s_j) > 0, \\
(1-\gamma) \frac{F(s_j)}{\sum_{s:s\in S(x_t), C(s_i s)=0} F(s)} & \text{if } s_i \in S(x_{t-1}),\ s_j \in S(x_t) \text{ and } C(s_i s_j) = 0, \\
0 & \text{if } s_i \notin S(x_{t-1}) \text{ or } s_j \notin S(x_t).
\end{cases} \tag{9}$$

6.2 Features

The main advantage of MEMM over HMM is the introduction of overlapping non-independent features, which reflect our prior knowledge about WSD and play a very important role in WSD. We design six types of features based on [52, 55].

1. POSs of neighbor words. P_i (P_{-i}) represents the POS of i^{th} word to the right (left) of the target word w. The employed features include P_{-3}, P_{-2}, P_{-1}, P_0, P_1, P_2, P_3. For example, in the lemmatized and POS tagged sentence "the/DT man/NN see/VB me/PRP look/VB at/IN the/DT iron/NN bar/NN ./PUNC", the target word is "bar", and the features are $\langle \text{IN}, \text{DT}, \text{NN}, \text{NN}, ., \epsilon, \epsilon \rangle$, where ϵ indicates that the position is out of the sentence.
2. Local collocation. C_{ij} represents the concatenation of lemmas of i^{th} word to j^{th} word, and the employed features include $C_{-1,-1}$, $C_{1,1}$, $C_{-2,-2}$, $C_{2,2}$, $C_{-2,-1}$, $C_{-1,1}$, $C_{1,2}$, $C_{-3,-1}$, $C_{-2,1}$, $C_{-1,2}$, $C_{1,3}$. For the example sentence, the $C_{-2,1}$ feature of target word "bar" is the_iron_bar.
3. Hypernym links of neighbor words. y is the hypernym of x, means that x is a kind of y. For example, fruit is the hypernym of banana. Moreover, the hypernym may also have its hypernym, and thus a hypernym link can be constructed. We take the hypernym links of the most frequent sense of three neighbors before and after the target word as features, so as to generalize lemma and alleviate data sparseness. For the example sentence, the words having hypernym in neighbors of target word "look" are "man", "see" and "iron", and their hypernym links are $\langle$man, male, person, organism, living thing, object, entity, causal agent, entity$\rangle$, $\langle$see, perceive$\rangle$ and $\langle$iron, metallic element, chemical element, substance, entity$\rangle$, respectively.
4. Semantic tags of neighbor words. Open words, i.e. adjective, adverb, noun and verb, are divided into more refined categories in WordNet. These categories are usually called semantic tags, and there are totally 45 semantic tags in WordNet, including 26 semantic tags for noun, 15 for verb, 3 for adjective, and 1 for adverb. We take the semantic tags of the most frequent sense of three neighbors before and after target word as features, also to alleviate data sparseness. For the example sentence, the words having hypernym in neighbors of target word "look" are "man", "see" and "iron", and their semantic tags are $\langle$noun.person$\rangle$, $\langle$verb.perception$\rangle$ and $\langle$noun.substance$\rangle$, respectively.
5. Verb frames of neighbor words. Verb frame is the type of sentence in which the word can be used, and we take the verb frames of the most frequent sense of verb in three neighbors before and after target word as features. For the example sentence, the verbs in neighbors of target word "look" are "saw", and its verb frames are $\langle 2, 8, 9 \rangle$, where 2^{nd} verb frame is "Somebody –s", 8^{th} verb frame is "Somebody –s something", and 9^{th} verb frame is "Somebody –s somebody".
6. Bag of words. We take all words in the sentence as features.

7 DISAMBIGUATION METHOD BASED ON CONDITIONAL RANDOM FIELD

Compared with HMM, MEMM can integrate overlapping non-independent features, which contributes to performance improvement of WSD, as shown in experiments

(cf. Section 9.2). However, MEMM normalizes the probabilities locally, which incurs "label bias" [56]. Label bias makes the state with low entropy tend to ignore context, which is in conflict with WSD that relies on the context, and limits the performance of WSD system. CRF [8] overcomes label bias by global normalization, but with very high time complexity. After adopting universal sense set, the training of CRF cannot be accomplished on standalone workstation. We reduce time complexity of CRF by approximate training, and further reduce actual running time from 145 days to 8 days by parallelizing CRF training algorithm.

7.1 Label Bias in the Disambiguation Method Based on MEMM

Many probabilistic models with independently trained classifiers suffer from label bias, such as classical probabilistic automata [57], discriminative Markov model [56], and MEMM. We explain label bias problem with the example sentence in the disambiguation method based on MEMM, as shown in Figure 6.

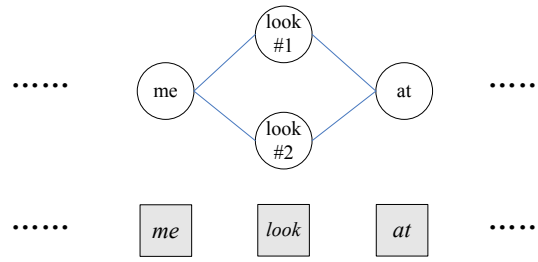

Figure 6. Label bias in disambiguation method based on maximum entropy Markov model

For simplicity, we only show the reduced label space of an excerpt from the example sentence. Suppose the verb "look" has only two senses[6], represented by look#1 and look#2 in the circle, respectively. Look#1 means perceiving with attention, for example, "Look at your child". Look#2 means having a certain outward aspect, for example, "The kids make me look happy.". Obviously, look#1 is the correct sense for the target word "look" in the example sentence. The "me" and "at" in the circle are the pseudo senses of words "me" and "at".

When sense "me" receives the word "me", there are two outgoing probabilities from sense "me", which embody the sense transition me–look#1 and me–look#2, respectively, and we denote them by p_1 and p_2. When sense look#1 and sense look#2 receives the word "look", both of them have only one outgoing transition, so sense look#1 and sense look#2 have no choice but to pass all their probability mass to next sense. This is because the discriminative model, such as MEMM, is conditioned on the observation, instead of generating it. It means that the transition probabilities look#1–at and look#2–at are always 1.0, without regard to next word "at", i.e. the

[6] Actually, the verb "look" has 10 senses in WordNet.

senses with a single outgoing transition effectively ignore their observation. However, sense look#1 has seen sense "at" often in the training corpus, while sense look#2 has almost never seen sense "at". The probability of upper path is $p_1 * 1.0 = p_1$, and the lower is $p_2 * 1.0 = p_2$. Therefore, the disambiguation result totally depends on the number of occurrences of sense transitions me–look#1 and me–look#2, and the sense transitions look#1–at and look#2–at are totally discarded, but it is not reasonable. In addition, the number of occurrences of me-look#1 may be smaller than that of me–look#2, since there are a lot of sentences like "... make me look ..." in English. So look#2 is likely to be chosen in the example sentence when MEMM is applied, but it is wrong.

Generally speaking, label bias makes the state with low entropy tend to ignore observation due to local normalization, and CRF overcomes this shortcoming by global normalization.

7.2 Model

CRF is a typical undirected graphical model, which is usually represented by factor graph, as shown in Figure 7.

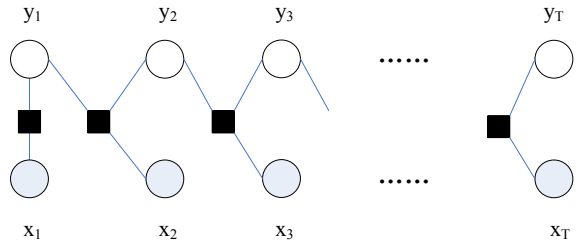

Figure 7. Factor graph of conditional random field

In factor graph, circle represents random variable, where circle without shadow is unobservable variable, and circle with shadow is observable variable. Black square represents factor, i.e. potential function. According to Equation (2), joint probability over labels given observations of CRF can be decomposed into

$$p(y_1, \ldots, y_T | x_1, \ldots, x_T) = \frac{1}{Z(x_1, \ldots, x_T)} \exp\left\{ \sum_i \theta_i F_i(y_1, \ldots, y_T, x_1, \ldots, x_T) \right\}, \quad (10)$$

where $Z(x_1, \ldots, x_T)$ is global normalization factor, $F_i(y_1, \ldots, y_T, x_1, \ldots, x_T)$ is global feature, and θ_i is weight of $F_i(y_1, \ldots, y_T, x_1, \ldots, x_T)$. Compared with local normalization of MEMM, cf. Equation (8), CRF alleviates label bias by global normalization. Training of CRF is to learn parameter $\theta_1, \ldots, \theta_K$. Given training data

$\left\{x_1^{(i)}, \ldots, x_T^{(i)}, y_1^{(i)}, \ldots, y_T^{(i)}\right\}$, solve

$$\begin{aligned}\widehat{\theta_1}, \ldots, \widehat{\theta_K} &= \underset{\theta_1,\ldots,\theta_K}{\operatorname{argmax}} \prod_i p_{\theta_1,\ldots,\theta_K}\left(y_1^{(i)}, \ldots, y_T^{(i)} | x_1^{(i)}, \ldots, x_T^{(i)}\right) \\ &= \underset{\theta_1,\ldots,\theta_K}{\operatorname{argmax}} \sum_i \left\{ \sum_j \theta_j F_j\left(y_1^{(i)}, \ldots, y_T^{(i)} | x_1^{(i)}, \ldots, x_T^{(i)}\right) \right. \\ &\quad \left. - \log Z\left(x_1^{(i)}, \ldots x_T^{(i)}\right) \right\}. \end{aligned} \tag{11}$$

We denote the objective function, i.e. the right part of Equation (11), by $L(\theta_1, \ldots, \theta_K)$. This is an unconstrained optimization problem, which can be solved by L-BGFS algorithm [58], one kind of quasi-Newton method, while L-BGFS requires the partial derivative of target function, and it is

$$\begin{aligned}\frac{\partial L(\theta_1, \ldots, \theta_K)}{\partial \theta_j} &= \sum_i \sum_t \sum_{y_{t-1}, y_t \in S} p_{\theta_1,\ldots,\theta_K}\left(y_{t-1}, y_t | x_1^{(i)}, \ldots, x_T^{(i)}\right) f_j\left(y_{t-1}, y_t, x_t^{(i)}\right) \\ &\quad - \sum_i \sum_t f_j\left(y_{t-1}^{(i)}, y_t^{(i)}, x_t^{(i)}\right). \end{aligned} \tag{12}$$

Equation (12) determines time complexity of CRF training $O(mLTN^2)$, where m is the number of iterations of L-BGFS, L is the number of training samples, T is the average length of training samples. Actually, L, T, N correspond to the three sums in the right hand of Equation (12). For WSD, N is the number of elements in the universal sense set, at least tens of thousands, thus the training complexity is by far out of the capacity of the standalone workstation.

7.3 Solution for High Complexity Training

The traversal in the square of label space results in high complexity, which corresponds to the third sum in the right hand of Equation (12).

7.3.1 Approximate Training

The solution in CRF is inspired by beam search for Viterbi algorithm, cf. Section 5.2. The pruning of search space is in the training phase of CRF, while that is in the prediction phase of HMM. The word only has a few number of senses, denoted by $S(x_t)$. When $y_{t-1} \notin S(x_{t-1}^{(i)})$ or $y_t \notin S(x_t^{(i)})$, the occurrence probability of observation

pair $y_{t-1}y_t$ is almost zero, i.e. $p_{\theta_1,\ldots,\theta_K}(y_{t-1}, y_t|x_1^{(i)}, \ldots, x_T^{(i)}) \approx 0$, and so we obtain

$$\frac{\partial L(\theta_1, \ldots, \theta_K)}{\partial \theta_j} \approx \sum_i \sum_t \sum_{\substack{y_{t-1} \in S(x_{t-1}^i) \\ y_t \in S(x_t^{(i)})}} p_{\theta_1,\ldots,\theta_K}\left(y_{t-1}, y_t|x_1^{(i)}, \ldots, x_T^{(i)}\right) f_j\left(y_{t-1}, y_t, x_t^{(i)}\right) - \sum_i \sum_t f_j\left(y_{t-1}^{(i)}, y_t^{(i)}, x_t^{(i)}\right). \quad (13)$$

Thus, the time complexity of CRF training is reduced from $O(mLTN^2)$ to $O(mLTR^2)$, where $R = \max_{x_t \in V} |S(x_t)|$, the number of most polysemous word. R is about tens, while N is at least tens of thousands.

7.3.2 Parallel Training

Although approximate training reduces time complexity of CRF significantly, it is still out of capacity of a standalone workstation when there is a large number of samples. In experiments, one iteration of L-BGFS for CRF approximate training on one CPU core costs 34.7 hours, and 100 iterations should cost about 145 days by estimation. The objective function of CRF training, i.e. the right part of Equation (11), is the sum over index of samples $1 \leq i \leq L$. Thus, we can split the training set into multiple subsets, calculate the objective function of subset on each CPU core, and then calculate the sum of target function of subsets to get that of whole training set [59]. The calculation of partial derivative of objective function can also be parallelized. The non-parallelizable part of the algorithm is only the optimization part of L-BGFS, and so the training of CRF has a high degree of parallelization, which is expected to obtain approximately linear speedup. The algorithm of parallel training of CRF is shown in Table 4. The time complexity of prediction in CRF can also be reduced from $O(TN^2)$ to $O(TR^2)$, which is similar to HMM and MEMM.

8 DISAMBIGUATION METHOD BASED ON TREE-STRUCTURED CONDITIONAL RANDOM FIELD

Given the example sentence, disambiguation method based on vanilla CRF treats the sentence as a linear chain, as shown in Figure 8.

However, the vanilla (linear-chain) CRF can only encode the dependency between adjacent words, which results in limitations. On one hand, the "content word – non-content word" pair or even the "non-content word – non-content word", such as the look–at or at–the pair, is less important for CRF, since the non-content word has only one pseudo sense. On the other hand, some long-distance dependencies between content words, such as the look–bar pair, are not reflected in the linear-chain CRF, since there is no direct relation between these two words in linear word sequence, as shown in Figure 8. However, the word "bar" is the object of the word "look", so their disambiguation results should be related.

Input:
Training data $D = \left\{x_1^{(i)}, \ldots, x_T^{(i)}, y_1^{(i)}, \ldots, y_T^{(i)}\right\}$
Number of CPU cores P, number of iterations m
Output:
Optimized parameters $\widehat{\theta_1}, \ldots, \widehat{\theta_K}$
Algorithm:
01: Generate initial parameter $\theta_1^{(0)}, \ldots, \theta_K^{(0)}$, and let $l = 0$
02: (Broadcast) Root process passes $\theta_1^{(l)}, \ldots, \theta_K^{(l)}$ to all processes
03: Each process calculates target function $L^p\left(\theta_1^{(l)}, \ldots, \theta_K^{(l)}\right)$ and its partial derivative $\frac{\partial L^p\left(\theta_1^{(l)}, \ldots, \theta_K^{(l)}\right)}{\partial \theta_j}$ on its training subset D^p
04: (Reduction) Root process calculates the sum of $L^p\left(\theta_1^{(l)}, \ldots, \theta_K^{(l)}\right)$ to get $L\left(\theta_1^{(l)}, \ldots, \theta_K^{(l)}\right)$ and the sum of $\frac{\partial L^p\left(\theta_1^{(l)}, \ldots, \theta_K^{(l)}\right)}{\partial \theta_j}$ to get $\frac{\partial L\left(\theta_1^{(l)}, \ldots, \theta_K^{(l)}\right)}{\partial \theta_j}$
05: Root process obtains $\theta_1^{(l+1)}, \ldots, \theta_K^{(l+1)}$ by optimization
06: Let $l = l + 1$. If $l = m$, then terminate, else goto 02

Table 4. Parallel training of conditional random field

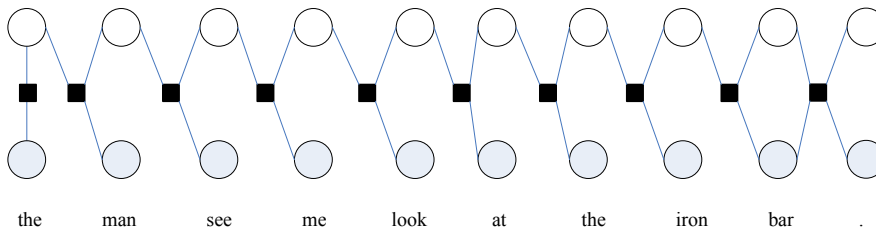

Figure 8. Factor graph of conditional random field with linear chain

Besides the surface linear chain structure, the sentence also has non-linear structure, which may reflect the long-distance dependency between words. In dependency grammar [60], syntactic structure is composed of asymmetric binary relations, which is call dependency relation. The subordinate word is call dependent, and the other is call head. The dependency tree of the example sentence is shown in Figure 9.

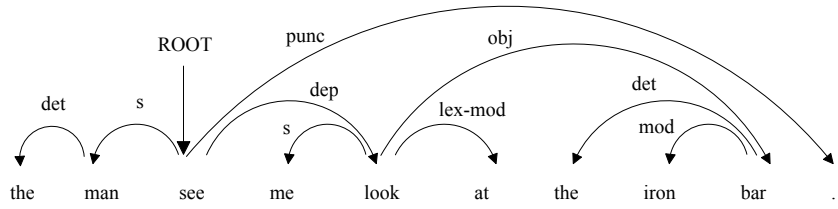

Figure 9. Dependency tree

We replace the linear chain structure in CRF with dependency tree, and the resulting probabilistic graphical model is call tree-structured conditional random field (TCRF), as shown in Figure 10.

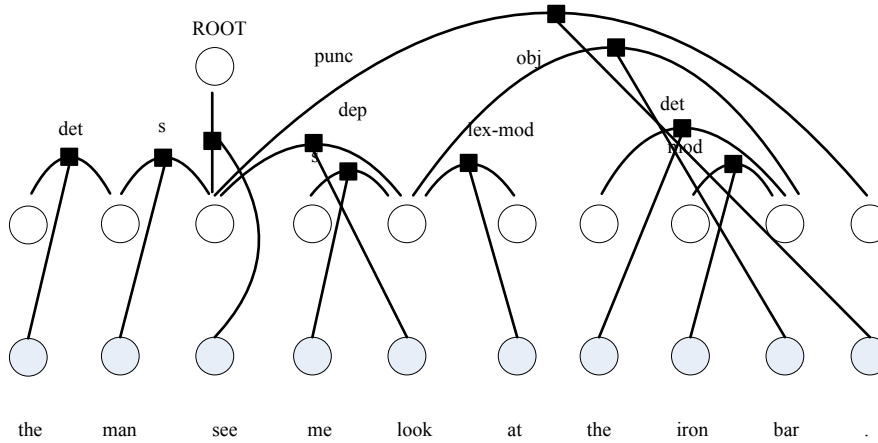

Figure 10. Factor graph of conditional random field with tree structure

In TCRF, the labels of "look" and "bar" are connected directly, reflecting the long-distance dependency between them. The training and prediction of TCRF is similar to CRF, with some extensions. In addition, TCRF also employs approximate training and parallelization for training, and beam search for prediction, which are similar to CRF.

9 EXPERIMENTS

SLMs label all elements in the structure simultaneously, which prefers to be evaluated by all-words WSD, rather than lexical sample WSD. English all-words WSD tasks (EAW) in Senseval-2001 [61] and Semeval-2004 [20] are two fully-explored and general-purpose evaluations with many attendees, and thus their datasets are used in this paper, whose information is shown in Table 5.

Year	Dictionary	MFS	Best result	IAA	No. words
Senseval-2001	WordNet 1.7	0.570	0.690	N/A	2 473
Senseval-2004	WordNet 1.7.1	0.609	0.651	0.725	2 037

Table 5. Test datasets of all-words WSD in Senseval/Semeval

EAW are also conducted in Semeval-2007 and Semeval-2010. There are two versions of EAW in Semeval-2007, and they are fine-grained EAW [62] (denoted by EAW if there is no specific explanation) and coarse-grained EAW [63]. However, EAW in

Semeval-2007 is much less influential than that of Senseval-2001 and Senseval-2004, due to its fewer test instances (only 465 test instances), fewer attendee systems and lower performance (recall 0.591) in similar settings. Coarse-grained EAW is interesting, but this paper focuses on the fine-grained one. EAW in Semeval-2010 [64] focuses on WSD in specific domain. So these datasets are not adopted in this paper.

EAW only provides a test dataset, but not a training dataset. At present, the largest sense labeled corpus is SemCor [22], and we take the subsets Brown1 and Brown2 in SemCor as the training set. These two subsets include 359 732 words, in which all contents words (192 639) are labeled with senses in WordNet. For the word not existing in the training set, we label it with most frequent sense (MFS) in WordNet. The most important measures in WSD evaluation are recall R and F_1 value. When coverage $C = 1$, recall R, precision P and F_1 value are all equal. $C = 1$ holds in all methods proposed in this paper, and so we only take R as measure.

9.1 Experimental Settings

With the beam search, the time complexity of prediction in HMM and MEMM is reduced significantly, and we run them on a standalone workstation. Smoothing parameter γ is set to 0.999 in HMM and MEMM, which has been mentioned in Section 5.3. General iterative scaling [54] is employed to train ME model in MEMM with 100 iterations. The observation of HMM can only be multinomial value, and lemma+POS is adopted as observation, while MEMM, CRF and TCRF can employ overlapping non-independent features, cf. Section 6.2.

Training time complexity of CRF and TCRF is very high, which cannot be accomplished on a standalone workstation with a general purpose training algorithm. Even with approximate training, the training of CRF costs 145 days by estimation. Thus, we use an enterprise-class parallel server, i.e. HP ProLiant DL 580 G7, to train CRF, and the configuration of the server is shown in Table 6.

Hardware		Software	
CPU	Xeron X7560 2.33 GHz	Operating system	Ubuntu 9.10 64-bit
No. CPU	4	Compiler	GNU C++ 4.4.2
No. cores per CPU	8	Parallel computing environment	MPICH2-1.3.2p1
Total No. cores	32		
Memory	64 GB		

Table 6. Configuration of parallel server

20 cores are used to train CRF, and 100 iterations cost 200.7 hours (about 8 days). Meanwhile, one approximate training iteration costs 34.7 hours on one core, and 100 iterations will cost 3 473.3 hours (about 145 days) by estimation. Thus, the speedup is $3\,473.3/200.7 = 17.3$, which is approaching linear speedup. If there are 200 cores, the training of CRF is expected to be done in one day. The number of iterations of CRF training is an empirical parameter, which is determined by validation set in this experiment. 10 % of Brown1 and Brown2 are selected

randomly as validation set, and the remaining 90 % as training set. The learning curve of CRF approximate training is shown in Figure 11.

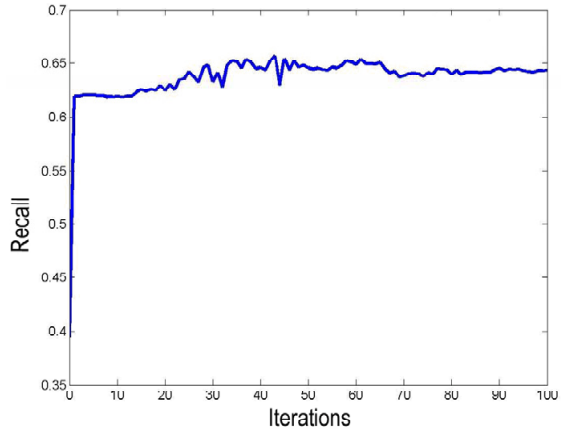

Figure 11. Learning curve of CRF approximate training

In Figure 11, the recall rises from 0.394 to 0.620 after only one iteration, and then enters a plateau. From 15th iteration, the recall rises slowly, it reaches maximum 0.655 at 41st iteration, and declines after 41st iteration, due to overfitting. In testing, the number of iterations is set to be 41.

9.2 Experimental Results

The main results of our methods on EAW of Senseval-2001 and Senseval-2004 are shown in Table 7. Performance of all methods proposed in this paper is between baseline MFS and IAA. Baseline MFS takes the most frequent sense as label of word, without regard to the context. Actually, MFS can be treated as naïve Bayes model (NB) with observation being lemma + POS, and the method based on HMM is sequential version of NB, with the same observation. Compared with NB, HMM introduces one step label dependency, improving WSD performance by 5.2 percent on EAW of Senseval-2001, by 1.6 percent on EAW of Senseval-2004.

The disambiguation method based on MEMM improves WSD performance further, by 2.6 percent on EAW of Senseval-2001, by 2.9 percent on EAW of Senseval-2004, which credits to the introduction of overlapping non-independent linguistic features. Except the rank 1st of EAW of Senseval-2001, the method based on MEMM outperforms all attendee systems in EAW of Senseval-2001 and Senseval-2004. The rank 1st in EAW of Senseval-2001 is SMUaw [65], which obtains initial sense labeled samples from SemCor and definition of WordNet, and then bootstraps on unlabeled data on Internet, which is a sort of semi-supervised learning method. It is well known that semi-supervised learning is an effective way to improve the performance

Method	Senseval-2001	Senseval-2004
HMM	0.622	0.625
MEMM	0.648	0.654
CRF	0.650	0.657
TCRF	0.664	0.668
Rank 1^{st}	0.690	0.651
Rank 2^{nd}	0.636	0.642
MFS	0.570	0.609
IAA	N/A	0.725

Table 7. Recall of systems on EAW of Senseval-2001 and Senseval-2004

of machine learning methods, and it can also be applied to SLMs, but it is not the point of this paper. Currently, the methods based on MEMM, as well as CRF and TCRF, only employ a part of labeled samples in SemCor, and they are expected to achieve better performance if bootstrapping on unlabeled data.

The method based on CRF is only slightly better than MEMM, reflecting that label bias does not constitute a serious problem in WSD. After adopting dependency tree as the underlying structure, the method based on TCRF improves performance by 1.1 percent, due to direct embodiment of non-linear dependency between words.

Paired t-test is conducted to see if one system is significantly better than another, and the system output of Rank 1^{st} and Rank 2^{nd} in EAW of Senseval-2004 are from Rada Mihalcea's webpage[7].

$$MFS < HMM \ll Rank2 \approx Rank1 \approx MEMM \approx CRF < TCRF. \quad (14)$$

In the above formula, systems are sorted by recall in ascending order, "$\approx$" means p-value > 0.05, "$<$" means $0.01 <$ p-value ≤ 0.05, and "$\ll$" means $p < 0.01$. A larger p-value indicates that the two systems are not significantly different from each other.

Also explored is the performance of proposed methods on different POSs in EAW of Senseval-2004, as shown in Figure 12. Adverb is the easiest to be disambiguated, and is also the hardest to be improved. Adjective and noun have the similar performance, and the hardest to disambiguate is verb. Except adverb, the performance improvements of different POSs have similar trends, and the disambiguation method based on TCRF achieves best performance on all POSs.

10 CONCLUSIONS

SLMs can encode the dependency among labels in structure, and this paper explores application of SLMs in WSD systematically. Due to the homogeneity of label space of SLMs, we have to take the universal sense set as label space, which results in extremely high time complexity. It is usually impossible to solve directly the original

[7] http://www.cse.unt.edu/∼rada/senseval/senseval3/data/Systems.Senseval3.tar.gz

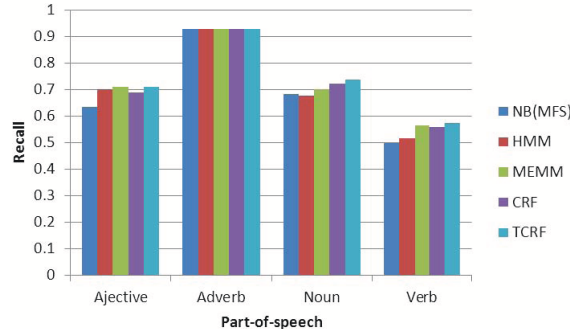

Figure 12. Performance on different part-of-speech in EAW of Senseval-2004

model based on SLMs on a standalone workstation, and this possibly accounts for few researches on WSD using SLMs.

Beginning with the method based on HMM, this paper proposes for the first time the complete solution for WSD based on MEMM, CRF and TCRF in an incremental fashion, as sketched in Figure 13. The performance increases with the updates of models, while the time complexity also increases dramatically. We propose beam search, approximate training to reduce time complexity, to employ parallel training to reduce actual running time, and to alleviate data sparseness with smoothing strategy. After great effort, these models can be solved in reasonable time.

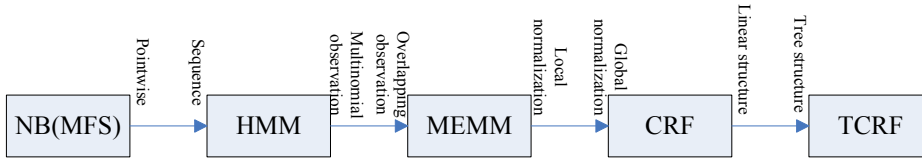

Figure 13. Relation of structured learning methods for word sense disambiguation

From HMM to TCRF, the series of improvements in models contribute to an increase in the performance, and three of them are significant:

1. the introduction of one step label dependency improves the performance by 1–5 percent which corresponds to extension from NB(MFS) to HMM;
2. the introduction of overlapping non-independent features improves the performance by 2–3 percent, reflecting the importance of linguistic knowledge to WSD which corresponds to extension from HMM to MEMM;
3. the introduction of tree structure improves the performance by about 1 percent, reflecting the importance of non-linear dependency among words which corresponds to extension from CRF to TCRF.

Although the improvements in models bring a performance increase, the results are not so exciting as expected. We give a possible reason for this. As stated before, point-wise learning usually disambiguates polysemous words in the sentence independently, and SLMs aim to reflect the label dependency among words. Meanwhile, the dependency is based on statistics, whether in point-wise learning or SLMs. Sense labeling by hand is a very hard labor, and there is no language with "adequate" sense-labeled corpus. Moreover, the label space of each polysemous word is different, and so the sense-labeled data for one word cannot be used by other words, resulting in a serious data sparseness. In the largest sense labeled corpus SemCor, only 83 words occur more than 100 times, and only 7 % bigram occur more than once. In this situation, no machine learning can take a full advantage of "statistics", and even exhibits spurious effects [66].

In the future, we shall further explore SLMs for WSD from the following directions. The first is to apply the proposed models to coarse grained WSD. WordNet has long been criticized for its too fine-grained senses, which is probably not required by some practical NLP tasks. The coarse-grained inventory benefits not only practical NLP tasks, but also the SLMs, since the label space is reduced compared with the fine-grained one. The second is to conduct detailed feature analysis. We still need to figure out the contribution of different kinds of features in the model, and to evaluate the proposed models and the state-of-the-art algorithms with the same set of features, as it is done in [67], which helps to better understand the nature of SLMs. The third is to employ other SLMs. For example, every decision in one-step MEMM only depends on the previous label, if one of the previous labels is wrong, then the performance of following classifiers will reduce significantly [68], even they are nearly equal to be selected randomly, due to error accumulation. That is to say, MEMM only exploits the optimal path, while SEArch-based structured leaRNing, SEARN [13], exploits sub-optimal paths while training, which may improve the disambiguation performance. The fourth is to solve the knowledge bottleneck of WSD by using all possible language resources, such as Wikipedia, search engine and parallel corpus.

Acknowledgement

This work is supported by the National Natural Science Foundation of China under Grants No. 61170156 and No. 61303068. We appreciate editors and anonymous reviewers for their valuable advice.

REFERENCES

[1] Agirre, E.—Edmonds, P.: Word Sense Disambiguation: Algorithms and Applications. Springer, 2007.

[2] Navigli, R.: Word Sense Disambiguation: A Survey. ACM Computing Surveys (CSUR), Vol. 41, 2009, No. 2, Art. No. 10.

[3] Miller, G.—Beckwith, R.—Fellbaum, Ch.—Gross, D.—Miller, K.: Introduction to WordNet: An On-line Lexical Database. International Journal of Lexicography, Vol. 3, 1990, No. 4, pp. 235–244.

[4] Kucera, H.—Francis, W. N.—Caroll, J. B.—Twadell, W. F.: Computational Analysis of Present-Day American English. Brown University Press, 1967.

[5] Miller, G. A.—Chodorow, M.—Landes, S.—Leacock, C.—Thomas, R.: Using a Semantic Concordance for Sense Identification. Proceedings of the Workshop on Human Language Technology, 1994, pp. 240–243.

[6] Bakir, G.—Hofmann, T.—Schölkopf, B.—Smola, A. J.—Taskar, B.—Vishwanathan, S. V. N.: Predicting Structured Data. The MIT Press, 2007.

[7] Peng, F.—Feng, F.—McCallum, A.: Chinese Segmentation and New Word Detection Using Conditional Random Fields. Proceedings of the 20th International Conference on Computational Linguistics (COLING '04), 2004, Art. No. 562.

[8] Lafferty, J. D.—McCallum, A.—Pereira, F. C. N.: Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. Proceedings of the Eighteenth International Conference on Machine Learning (ICML '01), 2001, pp. 282–289.

[9] Cutting, D.—Kupiec, J.—Pedersen, J.—Sibun, P.: A Practical Part-of-Speech Tagger. Proceedings of the Third Conference on Applied Natural Language Processing (ANLP '92), Trento, 1992, pp. 133–140.

[10] Kuta, M.—Chrzaszcz, P.—Kitowski, J.: A Case Study of Algorithms for Morphosyntactic Tagging of Polish Language. Computing and Informatics, Vol. 26, 2007, No. 6, pp. 627–647.

[11] Collins, M.: Discriminative Training methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing (EMNLP '02), 2002, Vol. 10, pp. 1–8.

[12] Sha, F.—Pereira, F.: Shallow Parsing with Conditional Random Fields. Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL '03), 2003, Vol. 1, pp. 134–141.

[13] Daumé III, H.—Langford, J.—Marcu, D.: Search-Based Structured Prediction. Machine Learning, Vol. 75, 2009, No. 3, pp. 297–325.

[14] McCallum, A.—Li, W.: Early Results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons. Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 (CONLL '03), 2003, Vol. 4, pp. 188–191.

[15] Cohn, T.—Blunsom, P.: Semantic Role Labelling with Tree Conditional Random Fields. Proceedings of the Ninth Conference on Computational Natural Language Learning (CONLL '05), 2005, pp. 169–172.

[16] Tsochantaridis, I.—Joachims, T.—Hofmann, T.—Altun, Y.: Large Margin Methods for Structured and Interdependent Output Variables. The Journal of Machine Learning Research, Vol. 6, 2005, pp. 1453–1484.

[17] Kral, P.—Cerisara, Ch.: Dialogue Act Recognition Approaches. Computing and Informatics, Vol. 29, 2010, No. 2, pp. 227–250.

[18] Senseval Offical Website. `http://www.senseval.org/`, 2012/09/20.

[19] Chklovski, T.—Mihalcea, R.: Exploiting Agreement and Disagreement of Human Annotators for Word Sense Disambiguation. Recent Advances in Natural Language Processing Conference (RANLP), Borovetz, Bulgaria, 2003.

[20] Snyder, B.—Palmer, M.: The English All-Words Task. Proceedings of the 3rd International Workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL-3), Barcelona, Spain, 2004, pp. 41–43.

[21] Palmer, M.—Dang, H. T.—Fellbaum, Ch.: Making Fine-Grained and Coarse-Grained Sense Distinctions, Both Manually and Automatically. Journal of Natural Language Engineering, Vol. 13, 2007, No. 2, pp. 137–163.

[22] Miller, G. A.—Leacock, C.—Tengi, R.—Bunker, R.: A Semantic Concordance. Proceedings of the Workshop on Human Language Technology (HLT '93), 1993, pp. 303–308.

[23] Segond, F.—Schiller, A.—Grefenstette, G.—Chanod, J.-P.: An Experiment in Semantic Tagging Using Hidden Markov Model Tagging. Proceedings of the ACL Workshop on Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications, 1997, pp. 78–81.

[24] de Loupy, C.—El-Beze, M.—Marteau, P.-F.: Word Sense Disambiguation Using HMM Tagger. Proceedings of the 1st International Conference on Language Resources and Evaluation (LREC), Granada, Spain, 1998, pp. 1255–1258.

[25] Crestan, E.—El-Beze, M.—de Loupy, C.: Improving WSD with Multi-Level View of Context Monitored by Similarity Measure. The Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems (SENSEVAL-2), 2001, pp. 67–70.

[26] Molina, A.—Pla, F.—Segarra, E.: A Hidden Markov Model Approach to Word Sense Disambiguation. Proceedings of the 8th Ibero-American Conference on AI: Advances in Artificial Intelligence, LNAI, Vol. 2527, 2002, pp. 655–663.

[27] Molina, A.—Pla, F.—Segarra, E.: WSD System Based on Specialized Hidden Markov Model. Proceedings of the 3rd International Workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL-3), Barcelona, Spain, 2004.

[28] Deschacht, K.—Moens, M.-F.: Efficient Hierarchical Entity Classifier Using Conditional Random Fields. Proceedings of the Second Workshop on Ontology Learning and Population, 2006, pp. 33–40.

[29] Duan, X.—Zhao, J.—Xu, B.: Word Sense Disambiguation through Sememe Labeling. Proceedings of the 20th International Joint Conference on Artifical Intelligence (IJCAI '07), 2007, pp. 1594–1599.

[30] Dong, Z.—Dong, Q.: HowNet and the Computation of Meaning. World Scientific Publishing Co. Pte. Ltd., 2006.

[31] Reichartz, F.—Paass, G.: Estimating Supersenses with Conditional Random Fields. Proceedings of Workshop on High-Level Information Extraction, ECML-PKDD, 2008.

[32] Hatori, J.—Miyao, Y.—Tsujii, J.: Word Sense Disambiguation for All Words Using Tree-Structured Conditional Random Fields. COLING, 2008, Companion Volume – Posters and Demonstrations, pp. 43–46.

[33] Miyao, Y.—Tsujii, J.: Maximum Entropy Estimation for Feature Forests. Proceedings of the Second International Conference on Human Language Technology Research (HLT '02), pp. 292–297.

[34] Hatori, J.: Personal Communication. 2010.

[35] Rabiner, L.: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of IEEE, Vol. 77, 1989, No. 2, pp. 257–286.

[36] McCallum, A.—Freitag, D.—Pereira, F.: Maximum Entropy Markov Models for Information Extraction and Segmentation. Proceedings of the 17th International Conference on Machine Learning (ICML), 2000, pp. 591–598.

[37] Taskar, B.—Guestrin, C.—Koller, D.: Max-Margin Markov Networks. Advances in Neural Information Processing Systems 16 (NIPS 2003), 2003.

[38] Gimpel, K.—Smith, N.: Softmax-Margin CRFs: Training Log-Linear Models with Cost Functions. HLT '10 Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, 2010, pp. 733–736.

[39] Berg-Kirkpatrick, T.—Bouchard-Côté, A.—DeNero, J.—Klein, D.: Painless Unsupervised Learning with Features. HLT '10 Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, 2010, pp. 582–590.

[40] Gimpel, K.—Das, D.—Smith, N.: Distributed Asynchronous Online Learning for Natural Language Processing. Proceedings of the Fourteenth Conference on Computational Natural Language Learning (CoNLL '10), pp. 213–222.

[41] Luo, Z.—Osborne, M.—Petrovic, S.—Wang, T.: Improving Twitter Retrieval by Exploiting Structural Information. Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012, pp. 648–654.

[42] Gimpel, K.—Schneider, N.—O'Connor, B.—Das, D.—Mills, D.—Eisenstein, J.—Heilman, M.—Yogatama, D.—Flanigan, J.—Smith, N. A.: Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, 2011, pp. 42–47.

[43] Tang, J.—Yao, L.—Zhang, D.—Zhang, J.: A Combination Approach to Web User Profiling. ACM Transactions on Knowledge Discovery from Data (TKDD), Vol. 5, 2010, No. 1, Art. No. 2.

[44] Tang, J.—Fong, A. C. M.—Wang, B.—Zhang, J.: A Unified Probabilistic Framework for Name Disambiguation in Digital Library. IEEE Transactions on Knowledge and Data Engineering, Vol. 24, 2012, No. 6, pp. 975–987.

[45] Smith, N.: Linguistic Structure Prediction. Morgan & Claypool Publishers, 2011.

[46] Martins, A. F. T.: Advances in Structured Prediction for Natural Language Processing. Ph.D. Thesis of Carnegie Mellon University, 2012.

[47] Jelinek, F.: Statistical Methods for Speech Recognition. MIT Press, 1997.

[48] Huang, X.—Acero, A.—Hon, H.-W.: Spoken Language Processing: A Guide to Theory, Algorithm, and System Development. Prentice Hall PTR, 2001.

[49] Katz, S.: Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. IEEE Transactions on Acoustics, Speech, and Signal Processing, 1987, pp. 400–401.

[50] Stevenson, M.—Wilks, Y.: The Interaction of Knowledge Sources in Word Sense Disambiguation. Computational Linguistics, Vol. 27, 2001, No. 3, pp. 321–349.

[51] Yarowsky, D.—Florian, R.: Evaluating Sense Disambiguation across Diverse Parameter Spaces. Natural Language Engineering, Vol. 8, 2002, No. 4, pp. 293–310.

[52] Lee, Y. K.—Ng, H. T.: An Empirical Evaluation of Knowledge Sources and Learning Algorithms for Word Sense Disambiguation. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Philadelphia, July 2002, pp. 41–48.

[53] Berger, A. L.—Della Pietra, V. J.—Della Pietra, S. A.: A Maximum Entropy Approach to Natural Language Processing. Computational Linguistics, Vol. 22, 1996, No. 1, pp. 39–71.

[54] Darroch, J.—Ratcliff, D.: Generalized Iterative Scaling for Log-Linear Models. The Annals of Mathematical Statistics, Vol. 43, 1972, No. 5, pp. 1470–1480.

[55] Tratz, S.—Sanfilippo, A.—Gregory, M.—Chappell, A.—Posse, Ch.—Whitney, P.: PNNL: A Supervised Maximum Entropy Approach to Word Sense Disambiguation. Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval '07), 2007, pp. 264–267.

[56] Bottou, L.: A Theorectical Approach to Connectionist Learning, with Applications for Speech Recognition (in French). Ph.D. thesis, Université de Paris XI, Orsay, France, 1991.

[57] Paz, A.: Introduction to Probabilistic Automata. Academic Press, 1971.

[58] Liu, D.—Nocedal, J.: On the Limited Memory BFGS Method for Large Scale Optimization. Mathematical Programming, Vol. 45, 1989, No. 3, pp. 503–528.

[59] Phan, X.-H.—Nguyen, L.-M.—Nguyen, C.-T.: FlexCRFs: Flexible Conditional Random Fields. `http://flexcrfs.sourceforge.net`, 2012/10/03.

[60] Kübler, S.—McDonald, R.—Nivre, J.: Dependency Parsing. Morgan & Claypool Publishers, 2009.

[61] Palmer, M.—Fellbaum, Ch.—Cotton, S.—Delfs, L.—Dang, H. T.: English Tasks: All-Words and Verb Lexical Sample. The Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems (SENSEVAL '01), 2001, pp. 21–24.

[62] Pradhan, S.—Loper, E.—Dligach, D.—Palmer, M.: SemEval-2007 Task 17: English Lexical Sample, SRL and All Words. SEMEVAL, 2007.

[63] Navigli, R.—Litkowski, K.—Hargraves, O.: SemEval-2007 Task 07: Coarse-Grained English All-Words Task. Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval '07), 2007, pp. 87–92.

[64] Agirre, E.—Lopez de Lacalle, O.—Fellbaum, Ch.—Hsieh, S.-K.—Tesconi, M.—Monachini, M.—Vossen, P.—Segers, R.: SemEval-2010 Task 17: All-Words Word Sense Disambiguation on a Specific Domain. Proceed-

ings of the 5th International Workshop on Semantic Evaluation (SemEval'10), 2010, pp. 75–80.

[65] Mihalcea, R. F.—Moldovan, D. I.: Pattern Learning and Active Feature Selection for Word Sense Disambiguation. The Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems (SENSEVAL'01), 2001, pp. 127–130.

[66] Gale, W. A.—Church, K. W.: Poor Estimates of Context are Worse than None. Proceedings of DARPA Speech and Natural Language Workshop, 1990, pp. 283–287.

[67] Radovanović, M.—Ivanović, M.—Budimac, Z.: Text Categorization and Sorting of Web Search Results. Computing and Informatics, Vol. 28, 2009, No. 6, pp. 861–893.

[68] Kääriäinen, M.: Lower Bounds for Reductions. Atomic Learning Workshop, 2006.

Yun Zhou received his B.Sc. and M.Sc. degrees from PLA University of Foreign Languages in 1999 and 2006, respectively. He is now a Ph.D. student of School of Computer, National University of Defense Technology. His research interests focus on natural language processing, machine learning and social network analysis.

Ting Wang received his B.Sc. and Ph.D. degrees from National University of Defense Technology in 1992 and 1997, respectively. He is now Full Professor and Doctoral Advisor of School of Computer, National University of Defense Technology. His research interests focus on natural language processing and social computing.

Zhiyuan Wang received her B.Sc., M.Sc. and Ph.D. degrees from National University of Defense Technology in 2003, 2006 and 2011, respectively. She is now Assistant Professor of State Key Laboratory of High Performance Computing, National University of Defense Technology & School of Computer, National University of Defense Technology. Her research interests focus on parallel and distributed systems, fault tolerance and scalability.