Transactions on Machine Learning
and Data Mining
Vol. 4, No. 2 (2011) 55-74

ISSN:1865-6781,
ISBN: 978-3-942952-07-1

ibai Publishing
www.ibai-publishing.org

Discovering Text Patterns by a New Graphical Model

Minhua Huang and Robert M. Haralick

Computer Science Department
The Graduate School and University Center
The City University of New York
New York, NY 10016

Abstract. We discuss a probabilistic graphical model that works for recognizing three types of text patterns in a sentence: noun phrases; the meaning of an ambiguous word; and the semantic arguments of a verb. The model has an unique mathematical expression and graphical representation compared to existing graphical models such as CRFs, HMMs, and MEMMs. In our model, a sequence of optimal categories for a sequence of symbols is determined by finding the optimal category for each symbol independently. Two consequences follow. First, it does not need to employ dynamic programming. The on-line time complexity and memory complexity are reduced. Moreover, the misclassification rate is smaller than that obtained by CRFs, HMMs, or MEMMs. Experiments conducted on standard data sets show good results. The performance of each task surpasses or approaches the state-of-art level.

Keywords: probabilistic graphic models, cliques, separators, NP chunks, word senses, semantic arguments

1 Introduction

Researchers have focused on using probabilistic graphical models, such as HMMs [1], MEMMs [2], or CRFs [3] to recognize patterns in texts [1]. These models are derived from either a joint or conditional probability function for a sequence of categories and a sequence of symbols (associated with a sentence) under some conditional independence assumptions. These assumptions might not be the best assumptions for capturing text patterns in a sentence. Moreover, for these graphical models, the maximum global

[1] Text patterns related to this research are NP chunks (noun phrases), the meaning of a polysemous word, and semantic arguments of a verb.

probability value cannot be determined until the last symbol of the sequence has been reached and the computation must be done by a dynamic programming algorithm. Although dynamic programming is an efficient optimization technique, the conditional independence assumptions we have chosen to use lead to an algorithm whose memory requirements and computational complexity are less than dynamic programming and whose performance is just as good or better.

The graphical models that lead to an optimization that must dependently thread through the sequence of class assignments to optimize the joint probability of the class assignment given the measurements, essentially use an implicit gain function that specifies a gain of one if all the class assignments are correct and zero if one or more of the class assignments are wrong. No partial credit is given for some correct assignments. This criterion leads to difficulties where noise in the text data extraction can cause the resulting optimal class assignments to hallucinate an incorrect yet seemingly coherent result. In effect what happens here is that a noisy or perturbed symbol at any position in the input sequence can produce a wrong local category then a wrong category path for part of the sequence.

In this paper, we discuss a probabilistic graphical model that improves this situation. In effect the model gives partial credit and yet takes class dependencies into account. The model is derived from the probability function of a sequence of categories given a sequence of symbols by using the information carried on each current symbol, the association between the current symbol and the preceding measurement, and the association between the current symbol and the succeeding measurement. The mathematical representation of the model can be found on Section 2.1 and the graphical representation of the model can be found in Figure 1.

Compared to existing graphical models, such as CRFs [3], HMMs [1], MEMMs [2], our model has an unique character. That is, even though sequential dependencies from class to measurement are modeled, the optimal class assignments can be determined independently. Specifically, a sequence of optimal categories for a sequence symbols is determined by finding the optimal category for each symbol independently in a way which takes into account the neighborly dependencies. Several benefits result. First, we do not need to compute a set of category paths and store them in order to determine the optimal category path once the last symbol of a sequence has been reached. As a consequence, for recognizing a new symbol sequence, the time complexity is reduced from $O(M^2N)$ to $O(MN)$ while the memory complexity is reduced from $O(MN)$ to $O(M)$. Numerical comparisons are shown on Section 2.5. Furthermore, when we make a mistake on one symbol in a sequence, it will not effect other correct decisions that have been made or will be made for other symbols. Therefore, the misclassification rate for the whole sequence of categories can be reduced. Indeed, this is the behavior we have observed using this kind of model for three different types of text pattern recognition.

We developed three algorithms for recognizing three kinds of text patterns based on this model. These patterns are noun phrases, the meaning of a polysemous word, and the semantic arguments of a verb in a sentence. For the first task, the model is used to assign each of word in the sentence with one of the predefined class labels [4]. Then a block is defined to find noun phrases. For the second task, a polysemous word is represented by a sequence of context words. The model is used to assign one of the classes (senses

of the polysemous word) to each context symbol. The sense of the ambiguous word is determined by selecting that sense assigned to more context symbols than any other sense. For the third task, the model is used to extract a path for each verb node in a labeled rooted tree (parser tree) associating with a sentence. From this path, we can find a set of roots, each of them associated with a semantic argument of the verb.

The rest of the paper is structured in the following way. The second section presents the proposed method. The third section describes the three tasks. The fourth section demonstrates the empirical results. The fifth section reviews the related researches. The sixth section gives a conclusion.

2 The Method

In the rest of the paper, depending on the task, a symbol can be a word, a word plus its speech tag, or a node in a parse tree.

2.1 Defining the Task

Let $S =< s_1, \ldots, s_N >$ be a sequence of N symbols associated with a sentence. Let C be a set of M categories, $C = \{C_1, \ldots, C_M\}$ [2]. The task is to find a sequence of categories $< c_1^*, \ldots, c_N^* >$, $c_n^* \in C$, that best describes $S =< s_1, \ldots, s_N >$ in the sense that

$$< c_1^*, c_2^*, ..., c_N^* >= \underset{c_1, c_2, \ldots, c_N}{argmax}\ p(c_1, c_2, \ldots, c_N | s_1, s_2, \ldots, s_N)$$

In order to find the sequence $< c_1^*, c_2^*, \ldots, c_N^* >$, we need to compute $p(c_1, c_2, \ldots, c_N|s_1, s_2, \ldots, s_N)$. For this we build a decomposable graphical model.

2.2 The Model

In the usual kind of hidden Markov models, there is a dependency in the class sequence. In our model, there is also sequence dependency. That dependency is between neighboring classes to measurements instead of between class and neighboring class. In effect the model works because the dependency from neighboring classes to measurements is greater than the dependency between class and neighboring class.

The conditional independence graph that defines our graphical model is shown in Figure 1. For comparison, the conditional independence graph that defines the typical Markov dependency in the class sequence is shown in Figure 2.

Our graphical model leads to the following representation for the probability

$$p(c_1, \ldots, c_N | s_1, \ldots, s_N) = \frac{\prod_{n=1}^{N} p(s_{n-1}|s_n, c_n) p(s_{n+1}|s_n, c_n) p(s_n|c_n) p(c_n)}{\sum_{(c_1, \ldots, c_N) \in C} \prod_{k=1}^{N} p(s_{k-1}|s_k, c_k) p(s_{k+1}|s_k, c_k) p(s_k|c_k) p(c_k)} \quad (1)$$

[2] $C_m \in C$ will have a different meaning for each of the different tasks

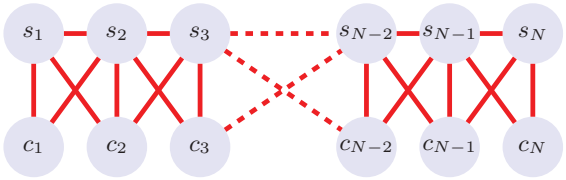

Fig. 1. The conditional independence graph defining our graphical model.

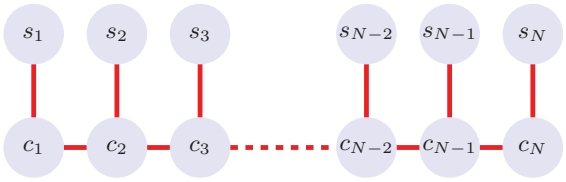

Fig. 2. The usual conditional independence graph for Markov dependencies among the classes.

2.3 Properties of The Model

Property 1. The Markov blanket for node c_n is s_{n-1}, s_n, s_{n+1}. Therefore, the Markov blanket property of the conditional independence graph tells us that class c_n is conditionally independent of $s_1, \ldots, s_{n-2}, s_{n+2}, \ldots s_N$ given s_{n-1}, s_n, s_{n+1}. Therefore, $P(c_n|s_1, \ldots, s_N) = p(c_n|s_{n-1}, s_n, s_{n+1})$

Property 2. Notice that in our conditional independence graph, all paths between nodes s_{n-1} and s_{n+1} must go through the one of the nodes in s_n and c_n. This means that s_{n-1} is conditionally independent of s_{n+1} given s_n and c_n. Hence $p(s_{n-1}, s_{n+1}|s_n, c_n) = p(s_{n-1}|s_n, c_n)p(s_{n+1}|s_n, c_n)$. Therefore,

$$\begin{aligned} p(c_n|s_{n-1}, s_n, s_{n+1}) &= \frac{p(s_{n-1}, s_n, s_{n+1}, c_n)}{p(s_{n-1}, s_n, s_{n+1})} \\ &= \frac{p(s_{n-1}, s_{n+1}|s_n, c_n)p(s_n, c_n)}{p(s_{n-1}, s_n, s_{n+1})} \\ &= \frac{p(s_{n-1}|s_n, c_n)p(s_{n+1}|s_n, c_n)p(s_n|c_n)p(c_n)}{p(s_{n-1}, s_n, s_{n+1})} \\ n &= 2, \ldots, N-1 \end{aligned}$$

Property 3. Properties 1 and 2 imply that

$$\begin{aligned} p(c_n|s_1, \ldots, s_N) &= p(c_n|s_{n-1}, s_n, s_{n+1}) \\ &= \frac{p(s_{n-1}|s_n, c_n)p(s_{n+1}|s_n, c_n)p(s_n|c_n)p(c_n)}{p(s_{n-1}, s_n, s_{n+1})} \\ n &= 2, \ldots, N-1 \end{aligned}$$

Property 4. A node s_i together with its left neighbour s_{i-1} and the node c_i forms a clique. A node s_i together with its right neighbour s_{i+1} and the node c_i forms of a

clique.[3] Moreover, nodes s_i and c_i form a separator and nodes s_i and s_{i+1} form a separator.[4]

Property 5. For a sequence of N symbols, our model has a set of $2N-2$ cliques and a set of $2N-3$ separators.

Property 6. Our model has an unique junction tree $G_1 = (V_1, E_1)$. Each $v \in V_1$ is a clique. Each $e \in E_1$ is a separator. The junction tree is illustrated in Figure 3, where nodes $A = \{s_{N-1}, s_N, c_N\}$, $B = \{s_{N-2}, s_{N-1}, c_{N-1}\}$, $C = \{s_2, s_3, c_3\}$, $D = \{s_1, s_2, c_2\}$, $E = \{s_{N-1}, s_N, c_{N-1}\}$, $F = \{s_{N-2}, s_{N-1}, c_{N-2}\}$, $G = \{s_2, s_3, c_2\}$, $H = \{s_1, s_2, c_1\}$.

Property 7. By the theorem for decomposable graphical model, the product of the probabilities for the cliques divided by the product of the probabilities for the separators is the joint probability $p(c_1, \ldots, c_N, s_1, \ldots, s_N)$

$$\begin{aligned} p(c_1, \ldots, c_N, s_1, \ldots, s_N) &= \frac{\prod_{n=2}^{N-1} p(s_{n+1}|s_n, c_n)p(s_{n-1}|s_n, c_n)p(s_n|c_n)p(c_n)}{\prod_{m=1}^{N-1} p(s_m, s_{m+1})} \\ &\times p(s_2|s_1, c_1)p(s_0|s_1, c_1)p(s_1|c_1)p(c_1) \\ &\times p(s_{N+1}|s_N, c_N)p(s_{N-1}|s_N, c_N)p(s_N|c_N)p(c_N) \\ &= \frac{\prod_{n=1}^{N} p(s_{n-1}|s_n, c_n)p(s_{n+1}|s_n, c_n)p(s_n|c_n)p(c_n)}{\prod_{m=1}^{N-1} p(s_m, s_{m+1})} \end{aligned}$$

Where $p(s_0|s_1, c_1) = 1$ and $p(s_{N+1}|s_N, c_N) = 1$.

Property 8. By property 7, the conditional probability can be obtained by:

$$\begin{aligned} &p(c_1, \ldots, c_N|s_1, \ldots, s_N) \\ &= \frac{p(c_1, \ldots, c_N, s_1, \ldots, s_N)}{\sum_{c_1, \ldots, c_N} p(c_1, \ldots, c_N, s_1, \ldots, s_N)} \\ &= \frac{\prod_{n=1}^{N} p(s_{n-1}|s_n, c_n)p(s_{n+1}|s_n, c_n)p(s_n|c_n)p(c_n)}{\sum_{c'_n \in C} \prod_{n=1}^{N} p(s_{n-1}|s_n, c'_n)p(s_{n+1}|s_n, c'_n)p(s_n|c'_n)p(c'_n)} \end{aligned}$$

2.4 Finding $< c_1^*, \ldots, c_N^* >$

Property 9. By property 7 and property 8, to find a sequence of categories $< c_1^*, \ldots, c_N^* >$ for a sequence of symbols $< s_1, \ldots, s_N >$, we only need to find

[3] A clique is a maximal complete set of nodes. Let $G = (V, E)$ be a graph, s.t $E \subseteq V \times V$. $\Lambda \subseteq V$ is a clique if and only if $\lambda_1, \lambda_2 \in \Lambda, \lambda_1 \neq \lambda_2$, imply $\{\lambda_1, \lambda_2\} \in E$ and there is no set that properly contains Λ with this property.

[4] $\Gamma = \{\Gamma_1, ..., \Gamma_M\}$ is a set of separators, where $\Gamma_k = \Lambda_k \cap (\Lambda_1 \cup ... , \cup \Lambda_{k-1})$ and $\Lambda_1, ..., \Lambda_K$ is a running set of cliques.

c_n^* for s_n individually. Note, the denominator in (1) is a constant. Therefore, it does not effect a decision making for assigning c_i to s_i.

$$< c_1^*, c_2^*, ..., c_N^* > = \prod_{n=1}^{N} \underset{c_n \in C}{argmax}\, p(s_{n-1}|s_n, c_n)p(s_{n+1}|s_n, c_n)p(s_n|c_n)p(c_n)$$
$$= \prod_{n=1}^{N} \underset{c_n \in C}{argmax}\, P_{s_{n-1},s_n,s_{n+1},c_n} \tag{2}$$

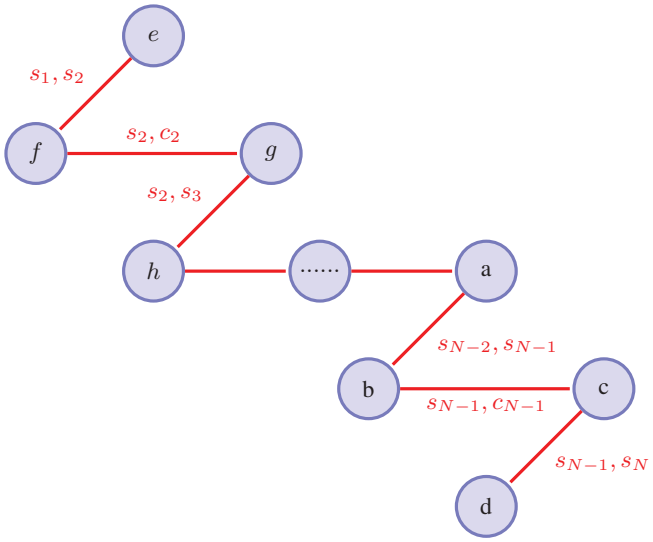

Fig. 3. The junction tree shows the cliques in running order and the separators between them. The product of the probabilities for the cliques divided by the product of the probabilities for the separators is the joint probability $p(c_1, \ldots, c_N, s_1, \ldots, s_N)$. $a = \{s_{N-2}, s_{N-1}, c_{N-2}\}$, $b = \{s_{N-2}, s_{N-1}, c_{N-1}\}$, $c = \{s_{N-1}, s_N, c_{N-1}\}$, $d = \{s_{N-1}, s_N, c_N\}$, $e = \{s_1, s_2, c_1\}$, $f = \{s_1, s_2, c_2\}$, $g = \{s_2, s_3, c_2\}$, $h = \{s_2, s_3, c_3\}$.

2.5 Complexities

Time Complexity. For each symbol s_n in equation (2), we need to assign a c_n, s.t.

$$max\{P_{s_{n-1},s_n,s_{n+1},c_n} | c_n \in C\} =$$
$$max\{p(s_{n-1}|s_n, c_n)\, p(s_{n+1}|s_n, c_n)p(s_n|c_n)\, p(c_n) \mid c_n \in C\}$$

The computation for $P_{s_{n-1},s_n,s_{n+1},c_n}$, requires four multiplications. To obtain the maximum probability value $max\ \{P_{s_{n-1},s_n,s_{n+1},c_n}|c_n \in C\}$, we must make $M-1$ comparisons. In the case of a sequence of N symbols, we need

$$T_c = 4 * N * (M-1) = O(N * M)$$

Memory Complexity. Because the global maximum probability is determined by each local maximal probability, for a sequence of N symbols, we only need to store the information of the current node. That is, we need only store M probability values in order to find the maximal probability value. Therefore,

$$M_c = M = O(M)$$

Comparisons. HMMs or CRFs employ dynamic programming to obtain a sequence optimal of categories for a sequence of symbols by computing a joint probability $p(s_1 \ldots s_n\, c_1 \ldots c_N)$ or or a conditional probability $p(c_1 \ldots c_N|\, s_1 \ldots s_n)$. By dynamic programming, an optimal category for the current symbol is obtained based on an optimal category of the previous symbol. Therefore, the optimal category for the last symbol is determined after the last symbol has been reached. The optimal category sequence needs to be determined by tracing back from the last optimal category to the first optimal category. For each sequence index, information for M categories needs to be stored. Hence, for a sequence of N symbols, the time complexity is $O(M^2N)$ and the memory complexity is $O(M * N)$.

Ratio of Time Complexity.

$$\frac{NM}{M^2N} = \frac{1}{M}$$

Ratio of Memory Complexity.

$$\frac{M}{M * N} = \frac{1}{N}$$

We compute ratios of time complexity and memory complexity of our model to HMMs and CRFs to see the differences. By observing these two ratios, we have noticed that if we need to recognize a sequence of N symbols with M categories, our model only takes $\frac{1}{M}$ time and $\frac{1}{N}$ memory space of HMMs or CRFs. For example, if the cardinality of C is ($M = 8$), for a sequence of thirty symbols ($N = 30$), our method only needs to have $\frac{1}{8}$ time and $\frac{1}{30}$ memory space of a HMM or a CRF to recognize this sequence.

3 Three Tasks

In the previous section, we compared the complexity of our model and other probabilistic graphic models such as HMMs and CRFs. Starting from this section, we

will discuss three tasks. They are to identify noun phrases (NP chunking), to identify the meaning of a polysemous word (word sense disambiguation WSD), and to identify semantic arguments (semantic role labeling SRL) of a verb in a sentence. A symbol sequence in each task associates with different objects. In NP chunking, it associates with a sentence; in WSD, it associates with the context of a polysemous word; in SRL, it associates with a path in a parse tree. Moreover, a set of class categories in each task also has different representations. In NP chunking, it is a set of locations of a word relating a noun phrase; in WSD, it is a set of predefined senses of the ambiguous word; in SRL, it is a set of choices from the current node to its neighbours. The model is applied to find an optimal category sequence for each symbol sequence. NP chunks are formed by finding blocks in the optimal category sequence; the meaning of a polysemous word is determined by selecting the most frequently appearing category in the optimal category sequence; and semantic arguments of a verb are determined by finding a set of subtrees from the optimal path.

3.1 Task 1: Identification of Noun Phrases

Let S be a sequence of symbols associated with a sentence, s.t. $S =< s_1, ..., s_i, ..., s_N >$, s.t. s_i is a pair consisting of a word and its speech tag. Let C be a set of categories, $C = \{C_1, C_2, C_3\}$, where C_1 represents a symbol inside a noun phrase, C_2 represents a symbol not in an noun phrase, and C_3 represents a symbol that starts a new noun phrase.

Building Blocks. $\mathcal{B}$ is a block if and only if:

1. For some $i \leq j$, $\mathcal{B} = < (s_i, c_i), (s_{i+1}, c_{i+1}), \ldots, (s_j, c_j) >$
2. $c_i \in \{C_1, C_3\}$
3. $c_n = C_1, n = i+1, \ldots, j$
4. For some $\mathcal{B}'$, if $\mathcal{B}' \supseteq \mathcal{B}$ and $\mathcal{B}'$ satisfying 1, 2, 3 $\rightarrow \mathcal{B}' \subseteq \mathcal{B}$

Formulating the Task: Identifying Noun Phrases

- Finding a sequence of categories $< c_1^*, \ldots, c_N^* >$, s.t.

$$< c_1^*, \ldots, c_N^* >= \underset{c_1,\ldots,c_N}{argmax}\, p(c_1, \ldots, c_N | s_1, \ldots, c_N)$$

- Finding $\{B_1, \ldots, B_M\}$, each B_m is a block satisfying the definition of $\mathcal{B}$.

An Example

- The sentence: *Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29.*
- The input sequence: *Pierre/NP Vinken/NP ,/, 61/CD years/NNS old/JJ ,/, will/MD join/VB the/DT board/NN as/IN a/DT nonexecutive/JJ director/NN Nov./NN 29/CD ./.*

- The category sequence: $C_1C_1C_2C_1C_1C_1C_2C_2C_2C_1C_1C_2C_1C_1C_1C_3C_1C_2$
- The blocks: $\overbrace{C_1C_1}^{\mathcal{B}_1}C_2\overbrace{C_1C_1C_1}^{\mathcal{B}_2}C_2C_2C_2\overbrace{C_1C_1}^{\mathcal{B}_3}C_2\overbrace{C_1C_1C_1}^{\mathcal{B}_4}\overbrace{C_3C_1}^{\mathcal{B}_5}C_2$
- The NPs:
  - *Pierre Vinken*
  - *61 years old*
  - *the board*
  - *a nonexecutive director*
  - *Nov. 29*

3.2 Task 2: Word Sense Disambuation

Let $S =< s_1, .., s_t, .., s_N >$ be a sequence of symbols associated with a sentence, $s_t \in S$ be a word that needs to be disambiguated . Let $C = \{C_m | M = 1, ..., M\}$, $C_m \in C$, where C is a set of predefined senses of s_t.

Defining Contexts. The Context of an ambiguous symbol s_t is a $k-tuple$, represented by T_t. Each element in T_t is a symbol in S, $T_t = (t_1, ..., t_K)$, $t_k \in S$, and $K \leq N$.

Formulating the Task: Identifying Word Senses

- Find the context T_t for s_t.
- Find a sequence of categories $< c_1^*, \ldots, c_K^* >$ for $T_t = (t_1, ..., t_K)$, s.t.

$$< c_1^*, \ldots, c_K^* >= \underset{c_1,\ldots,c_K}{argmax}\, p(c_1, \ldots, c_K | t_1, \ldots, t_K)$$

- Assign s_t to C_j if and only if

$$c_t = max\{\#\{c_k = C_j | C_j \in C, k = 1, ..., K\}\}$$

An Example

- The sentence: *Yields on money-market mutual funds continued to slid, amid signs that portfolio managers expect further declines in interest rates.*
- s_t: *interest*
- $C = \{C_1, C_2, C_3\} = \{$*money paid for the use of money*, *a share in a company*, *readiness to give attention* $\}$
- The context of s_t is a $6-tuple$: *(yields money-market funds portfolio manager rates)*
- The category sequence: $C_1C_1C_1C_3C_2C_1$
- Assign C_1 to $interest$

3.3 Task 3: Semantic Role Labeling

Let $T = (V, E, r, A, L)$ be a labeled rooted tree associated with a sentence, where V is a set of vertices, E is a set of edges, $E \subseteq V \times V$, r is the root, A is an alphabet defined by [5], and L is a labeling function $L : V \rightarrow A$ that assigns labels to vertices. T is a form for a parse tree of the sentence. Let π be a set of labels, s.t. $\pi \subseteq A$. Let $C = \{C_1, C_2\}$ be a set of class categories, where C_1 represents that a path will be extended from the current node to an adjacent node; C_2 represents that a path will not be extended from the current node to an adjacent node.

Defining Labeled Rooted Forests. A labeled rooted forest is a set of labeled rooted trees, s.t.

$$F = \{T_i | i = 1, ..., N\}, \quad T_i \textit{ is a labeled rooted tree.}$$

Formulating the Task: Identifying Semantic Arguments

- Form a path $\mathcal{P}(x) = \tau_1, \rightarrow \ldots, \rightarrow \tau_K$, $x \in V$, $L(x) \in \pi$, and x is not a node in $\mathcal{P}'(y)$, $\mathcal{P}'(y)$ is a path that has been already formed previously.

$$< \tau_1, \ldots, \tau_K > = \underset{b_1,\ldots,b_K}{argmax}\, p(c_1, \ldots, c_K, b_1, \ldots, b_K)$$

  Note, $c_k \in C$, $b_k \in V$, $b_{k-1}b_k \in E$.
- Form a set of roots $R(x) = \{r_i | i = 1 \ldots M\}$ from $\mathcal{P}(x)$, where $r_i \leq \tau_k$.
  - For all siblings of τ_k, find z, s.t. $L(z) \notin \pi$ and $z \notin \{\tau_k | k = 1, \ldots, K\}$, $R(x) \leftarrow R(x) \cup \{z\}$
  - For all children of τ_k, find y, s.t. $L(y) \notin \pi \; and \; y \notin \{\tau_k | k = 1, \ldots, K\}$, $R(x) \leftarrow R(x) \cup \{y\}$
- Find a rooted forest $F(x) = \{T_i | i \in \{1, .., I\}\}$,
  - Each T_i is induced from the root r_i by all its codependents.
  - For each $T_i \in F(x)$, the leaves $\{l_i^1, ..., l_i^K\}$ are corresponding to one of the semantic arguments of x.

An Example

- The sentence: *Mrs. Hills said that the U.S. is still concerned about "disturbing developments in Turkey and continuing slow process in Malaysia".*
- The labeled rooted tree T for this sentence is in Figure 4.
- A path for verb *x* = *concern* is in Figure 5.
- A labeled rooted forest $F(x) = \{T_1, T_2, T_3\}$ is in Figure 6, Figure 7, and Figure 8.
- The semantic arguments of verb *concern* are:
  - *still*;
  - *the U.S.*;
  - *about "disturbing developments in Turkey and continuing slow process in Malaysia"*

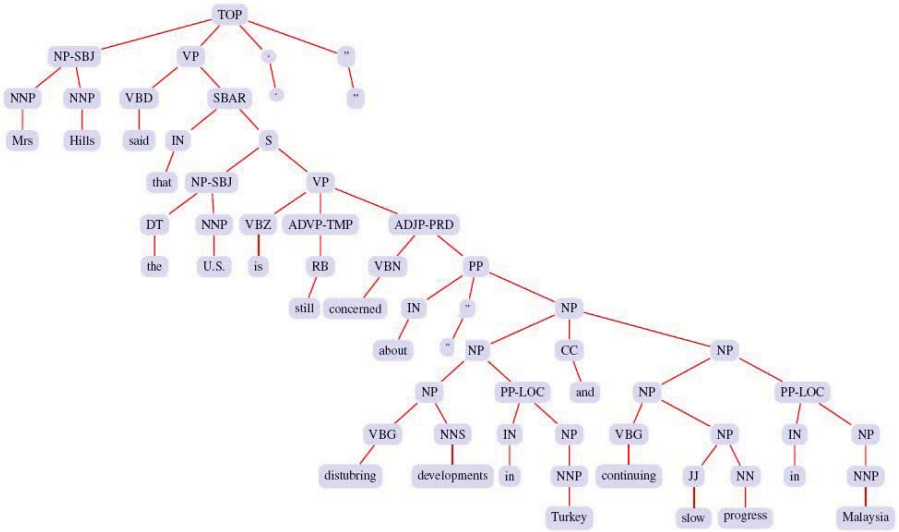

Fig. 4. The parse tree of the sentence: Mrs. Hills said that the U.S. is still concerned about "disturbing developments in Turkey and continuing slow process in Malaysia".

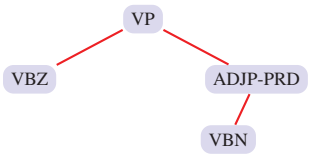

Fig. 5. All the semantic arguments of the verb *concern* can be extracted from this path.

4 Empirical Results

4.1 Experimental Set Up

In order to verify our methods, we have tested the first task on WSJ data from the Penn TreeBank [6] and CoNLL-2000 Shared Task [7], the second task on data developed by [8] [9], and the third task on WSJ data from the Peen TreeBank and the PropBank [5]. Our reasons for using these data sets are that they have been studied by multiple researchers and many results have been published over the years. The evaluation metrics designed for testing the first and the third tasks were *Precision , recall, f-measure* (F_1) and for testing the second task were *accuracy*. The reason of selecting different evaluation methods was based on the design of class categories described in sections 3.1, 3.2, and 3.3 [5]. One of categories was not needed to be evaluated in task

[5] There was a 'NOT' category (representing none of these categories) in the first task and third task while every category was a distinct sense in the second task.

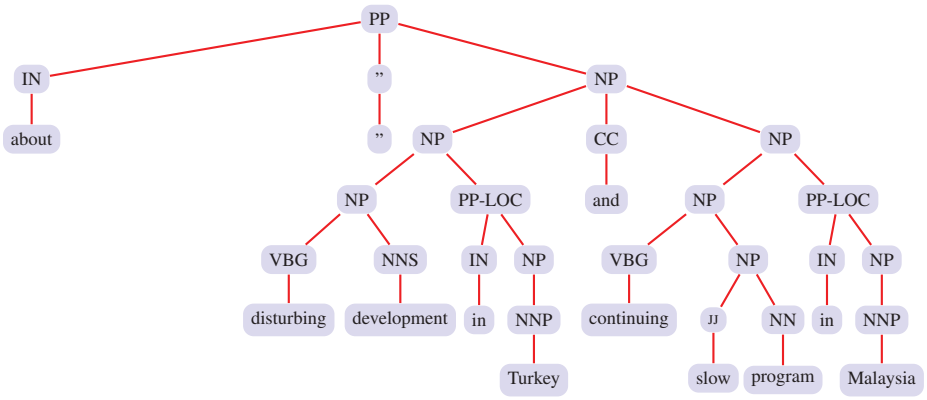

Fig. 6. Labeled rooted tree T_3 associating with one of the semantic arguments of verb *concern*

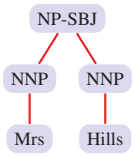

Fig. 7. Labeled rooted tree T_1 associating with one of the semantic arguments of verb *concern*

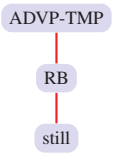

Fig. 8. Labeled rooted tree T_2 associating with one of the semantic arguments of verb *concern*

one and three while all categories were needed to be evaluated in task two. All our experimental results use a 10-fold cross validation technique for obtaining an result in all experiments. The reason of not using 10-fold cross validation was that we had a larger number of instances in the data set [6].

4.2 Results on the First Task

Three types of symbols were designed for identifying NP chunks on CoNLL-2000 Shared Task data set. They were the lexicon of a word, the POS tag of a word, and the lexicon and the part of speech (POS) tag of a word. The results are shown in the second row of Table 1. By comparing the results, we notice from column four (F-measure) that if the model was built only on the lexical information, it had the lowest performance

89.75%. The model's performance improved 3% if it was constructed by POS tags. The model achieved the best performance of 95.59% if both lexicon and POS tags were included.

Different from the first experiment, the second experiment on the WSJ data from Penn Treebank used only one type of symbol: the lexicon and the POS tag of a word. The main reason for using this data set was that we wanted to see whether the performance of our model could be improved when it was built on more data. In this case, the training set was seven times larger than the CoNLL-2000 shared task training data set. The test results was shown in the third row of Table 1. Data inside parentheses in the table represented standard deviation.

Compared with the results on these two data sets, we noticed that the average precision was improved about 2.74% from 95.15% to 97.89% . The average recall was improved about 2.24% from 96.05% to 98.29%. The average F-measure was improved about 2.50% from 95.59% to 98.09% as the training sets expanded into seven times larger. This suggested that a tradeoff needs to be considered between the sizes of training sets and the performances of the method.

Table 1. The test results on the CoNLL-2000 and WSJ data

Data	Symbol type	Precision %	Recall %	F-measure %
CoNLL-2000 Shared Task Data	Lexicon + POS	95.15	96.05	95.59
	POS	92.27	93.76	92.76
	Lexicon	86.27	93.35	89.75
WSJ from Penn Treebank	Lexicon + POS	97.89 (0.62)	98.29 (0.57)	98.09 (0.58)

Comparisons Table 2 shows the best performances of related methods on the CoNLL-2000 shared task data. The F-measures are ordered in descending order. Among of them, the role based learning achieves the worst F-measure performance and our method achieves the best F-measure performance.

4.3 Results on the Second Task

We tested our method for identifying the sense of a word on the data sets *line, hard, serve*, and *interest*. The senses' descriptions and instances' distributions can be found in [8] and [9]. In these data sets, *line* and *interest* are polysemous nouns, *hard* is a polysemous adjective, and *serve* is a polysemous verb. In our experiment, *line* had 6 senses, *serve* 4 senses, *hard* 3 senses, and *interest* 3 senses (3 other senses were omitted due to insufficient number of instances). The test metric that we use is $accuracy$.

We formed the context of each given target word by including the left four open class words and the right four open class words combining with the left word and the right word for each of these words. The test results were indicated in Table 3.

Table 2. Comparisons for different methods on the CoNLL-2000 data set

Method	Recall %	Precision %	F-measure %
Role Based Learning [10]	92.03	91.05	91.54
HMM [1]	93.52	93.43	93.48
MEMM [11]	–	–	93.70
Voted perceptrons [12]	93.29	94.19	93.74
CRF [11]	–	–	94.38
SVM [13]	94.38	94.52	94.45
our method [14]	95.31	96.36	95.74

Table 3. The second algorithm on *line, serve, hard, interest* data

Ambiguous word	Senses	Accuracy %	Standard deviation	Maximum %	Minimum %
$line$ (noun)	6	81.16	1.92	84.50	78.0
	3	85.25	2.13	91.70	81.05
$serve$ (verb)	4	79.80	1.90	82.92	76.88
$hard$ (adjective)	3	82.88	3.10	87.03	78.11
$interest$ (noun)	3	92.10	2.21	95.50	86.00

By analyzing our errors, we found that the misclassified instances are primarily generated by the ambiguity of context words. For example in Table 3, comparing with three sense noun $interest$ and three sense noun $line$ obtained by selecting three senses at each time from six senses and examining all twenty combinations, we found that the accuracy of the word $interest$ was almost 9% higher than the accuracy for the word $line$. Moreover, by examining accuracies generated from each combination for the word $line$, we found that some combination ($S_1S_2S_4$) has the highest average accuracy: 91.7% while some combination ($S_1S_3S_5$) has lowest average accuracy: 77.1%. The difference is almost 20%. By carefully checking these misclassified instances, we learned that if two senses are similar to each other, there were more chances that their contexts consisted of the same words. As a consequence, the misclassification rate had to increase.

From the results in Table3, if we average the results from the ambiguous nouns, the ambiguous adjective, and the ambiguous verb, our model achieves an average of accuracy 83.5184%. This result is encouraging and surpasses the results published by other researchers [8] and [15]. Moreover, by observing the outputs of two polysemous nouns *line* and *interest*, we find that as number of senses of a polysemous noun increases, the accuracy decreases. This suggests that nouns with a larger number of senses are more difficult to recognize than nouns with small number of senses by our model. Furthermore, by observing the mean in column three, we notice that nouns were relatively easier to identify than adjectives or verbs. From the standard deviation in column four, we see that the accuracy produced by our model on adjectives has a larger variance than that on nouns or verbs.

Comparisons Our results are better than the results reported by other WSD researchers [15] and [8]. Our method achieves the average accuracy 81.12% for identifying the six sense noun $line$ using 2450 training context words while the method proposed by [8] achieves the average accuracy 73% using 8900 training context words. Moreover, the experiment of Latent Semantic Analysis method conducted by [15] achieves an average accuracy of 75% for identifying only three senses of $line$. The comparisons are shown in Table 4

Table 4. $line$ Results Comparisons

	Accuracy 3 senses %	Accuracy 6 senses %	# of Context words in Training Set 6 senses k	Base Line 6 senses %
LSA [15]	75			
Bayesian [8]	76	71	8.9	16.67
Context Vector [8]	73	72	8.9	16.67
Neural Network [8]	79	76	8.9	16.67
This Method	85.25	81.12	2.45	19.09

4.4 Results on the Third Task

The data set, the section 00 of WSJ from Penn Treebank and PropBank [5], was used for testing the third task. A total of 223 sentences was placed in files *20, 37, 49, and 89*. Associated with each of these sentences, is a parse tree provided by Penn Treebank. The Penn Treebank parse tree has been evaluated and found to have an average accuracy of 95.0%. Among these sentences, there were 621 verbs. Each verb had an average of three semantic arguments. Hence about 2000 semantic arguments were used. These semantic arguments were provided by PropBank tagged by human processed labels.

Among 621 verbs, about 560 verbs were used for obtaining probability values while about 60 verbs were used to form paths based on these probability values. Some of the paths were listed on Table 5. They were obtained based on the procedure described in Section 3.3. We observe that 86% of the paths fall into the first three patterns.

After forming a path for a verb in test instances, a set of roots were found. From these roots, a set of labeled rooted subtrees, whose leaves were associating with semantic arguments of the verb, was formed. The test results were shown in Table 6. On the average, each time among $\frac{1}{10}$ of the semantic arguments were classified, about 93% semantic arguments were correctly identified and 7% semantic arguments were classified wrong. By checking these classified instances, we found that our method was very effective in the case of a semantic argument being a sequence of consecutive words. However, if a semantic argument consisted of two or more word fragments, separated by some phrases, our algorithm was less effective. The reason was that these phrases were parts of leaves of a tree induced from a root determined by our algorithm. This

Table 5. Six Types of Paths

NO	%	Path
1	62.1	$VBZ(VBD, VBG, VBP, VBN, VB) \rightarrow VP$
2	14.2	$MD(TO) \rightarrow VP \rightarrow VP \rightarrow VB$
3	10.1	$VBP(VBZ, VBD) \rightarrow VP \rightarrow VP \rightarrow VBN$
4	4.2	$VBD(VBZ, VBN) \rightarrow VP \rightarrow RB \rightarrow VP \rightarrow VB$
5	2.4	$TO \rightarrow VP \rightarrow VP \rightarrow VB \rightarrow VP \rightarrow VBN$
6	2.2	$MD \rightarrow VP \rightarrow RB \rightarrow VP \rightarrow VBP(VB) \rightarrow VP \rightarrow VBN$

suggests us that in order to exclude phrases from a semantic argument, we need to develop a method so that a set of subroots needs to be found. Each of them corresponds to a fragment of a semantic argument. Moreover, other misclassified instances are generated by errors carried in the Penn Treebank parse trees.

Table 6. The Third Task on WSJ data

Files 20, 37, 49, 89	Precision %	Recall %	F-Measure %
Average	92.335	94.1675	93.2512
Standard Deviation	0.6195	0.5174	0.4605

5 Related Research and More Comparisons

Different Graphical Representations. The graphical models used by most researchers are HMMs [2] [1], MEMMs[2], and CRFs[3] [11]. These models are built to obtain an optimal sequence of N categories $c =< c_1, \ldots, c_N >$ from a sequence of N symbols $s =< s_1, \ldots, s_N >$ by finding the maximum value of the joint probability $p(c, s)$ or the conditional probability $p(c|s)$. The conditional independence graphs for these graphical models are shown in Figure 9. While HMMs and MEMMs are directed graphical models, CRFs and our model are undirected graphical models. While others have a link from c_{i-1} to c_i, our model links between c_i and s_{i+1} and between c_i and s_{i-1}. We believe that c_i can be better predicated from s_{i-1} and s_{i+1} rather than c_{i-1} when the symbols contain several types of information. For example, in the case of NP chunking, the POS tag information carried on a symbol is more useful than the category of the previous symbol.

Different Assumptions. In the graphical model representation, $s_i,\ i = 1, ..., N$ and $c_i,\ i = 1, ..., N$ are nodes. We have $2N$ nodes in total. If no assumption is made, there is a link between every pair of nodes. The degree of each node should be $2N - 1$. Some assumptions are made for the graphical models represented in Figure 9. Compared to

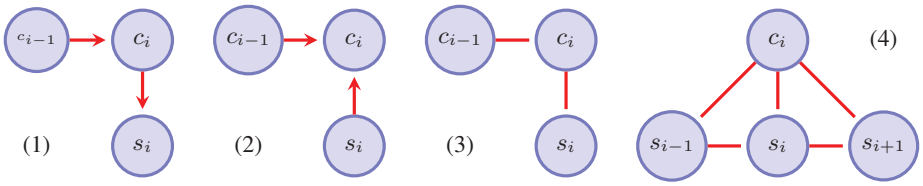

Fig. 9. (1): a HMM model, (2): a MEMM model, (3): a CRF model, and (4): the model presented by this paper

these graphical models, we find that for each s_i, the degree of our model is three while the others have degree two. This indicates that our model has less assumptions. By analysis, a HMM is built under two conditional independence assumptions. First, given its previous category, the current category is independent of other categories. Moreover, given its current category, the symbol is independent of other categories and symbols. A $MEMM$ is built under one conditional independence assumption. Given its previous category and the current symbol, the current category is independent to other categories and symbols. A CRF is built under the same two conditional assumptions as a HMM. The model presented on this paper makes one conditional independence assumption. Given the current, the preceding, and the succeeding symbol, the current category is independent of other categories and symbols.

Joint Probability Decompositionl Expressions. The joint probabiity decomposition associated with directed graphic models can be read directly from its graphical representation. The joint probability decomposition for undirected graphical models can be derived by taking the product of the cliques divided by the product of separators. In Figure 9, the HMM and the $MEMM$ are directed models while the CRF and our model are undirected models. The joint or conditional probability decompositions of these models are as follows.

- HMMs:

$$p(c, s) = \prod_{i=1}^{N} p(s_i|c_i)p(c_i|c_{i-1})$$

- $MEMM$s:

$$p(c|s) = \frac{\prod_{i=1}^{N} p(c_i|c_{i-1}s_i)}{\sum_{c_n \in C} \prod_{n=1}^{N} p(c_n|c_{n-1}s_n)}$$

- CRFs:

$$p(c|s) = \frac{\prod_{i=1}^{N} p(s_i|c_i)p(c_i|c_{i-1})}{\sum_{c_n \in C} \prod_{n=1}^{N} p(s_n|c_n)p(c_n|c_{n-1})}$$

- Our model:

$$p(c|s) = \frac{\prod_{i=1}^{N} p(s_{i-1}|s_i, c_i)p(s_{i+1}|s_i, c_i)p(s_i|c_i)p(c_i)}{\sum_{c_n \in C} \prod_{n=1}^{N} p(s_{n-1}|s_n, c_n)p(s_{n+1}|s_n, c_n)p(s_n|c_n)p(c_n)}$$

Comparing the numerators for $p(c|s)$, $MEMM$ has one term, CRF has two terms, and our model has four terms in a numerator. In all the other models, c_i and c_{i-1} appear together in some term. In our model c_i and c_{i-1} never appear together in the same term.

Comparisons Related To The Three Tasks. A numbers of methods for NP chunking [16] [17] [1] [18] [13], word sense disambiguation [19] [20] [8] [21] [22], and semantic role labelling [23] [24] [25] [26] [27] have been developed over the years. We adopted some ideas from these methods. For instance, in NP chunking, we follow Ramshaw's idea [17] of designing three categories for a word in a sentence to determine whether the word is inside a NP chunk, outside a NP chunk, or the start a new NP chunk. However, most methods for this task use HMMs [2] [1], MEMMs[2], and CRFs[3] [11]. In contrast to these methods, we have created a new method whose graphical model uses a different set of conditional independence assumptions.. Our model is fast, uses less memory, and works well for text data.

In the WSD task, in contrast with other WSD methods, the polysemous word is represented by a sequences of context symbols, each symbol is an ordered pair of the lexicon and the POS tag of a word. Each symbol is represented by it's left symbol and right symbol. Moreover, in semantic argument identification task, most existing methods transform a syntactic tree into a sequence of constituents. Each argument of a verb is represented by a set of constituents. Each constituent is represented by a set of features. These features are extracted based on linguistic knowledge and local knowledge of the tree structure. Finally, sophisticated classifiers such as support vector machines or maximum entropy classifiers have been employed to identify the semantic arguments of each verb. In contrast to these methods, our method is based on the idea that if a sentence has a correspondent labeled rooted tree, a semantic argument of a verb in the sentence will be associated with a labeled rooted subtree. Hence, all semantic arguments of a verb in the sentence will be represented by a set of labeled rooted subtrees. For each verb node v, there exists a path, from which, all roots of the subtrees will be extracted. Obviously, the unique feature, which is a path, represents all semantic arguments of a verb. We find such a path for each verb in a labeled rooted tree associated with a sentence by the probabilistic graphical model.

6 Conclusions

We have discussed a new probabilistic graphical model. The joint probability for obtaining a sequence of categories from a sequence of symbols has a decomposition as a product of marginal and conditional probability functions. It does not need to employ dynamic programming for obtaining a sequence of optimal categories. As a consequence, it requires less computing time and less memory space then existing graphical models. Moreover, because a sequence of optimal categories for a sequence of symbols is determined by finding the optimal category for each symbol independently, the misclassification rate is reduced. The performance of our model on three different types of text pattern recognition: noun phase identification, word sense disambiguation, and semantic arguments of a verb classification, is better than or equal to that compared to other state of the art algorithms, thereby demonstrating its superiority and effectiveness.

References

1. Molina, A., Pla, F., tics, D.D.S.I., Hammerton, J., Osborne, M., Armstrong, S., Daelemans, W.: Shallow parsing using specialized hmms. Journal of Machine Learning Research 2, 595–613 (2002)
2. MaCallum, A., Freitag, D., Pereira, F.: Maximum entropy markov models for information extraction and segmentation. In: Proceedings of 17th International Conf. on Machine Learning. pp. 591–598 (2000)
3. Lafferty, J., MaCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of 18th International Conf. on Machine Learning. pp. 282–289 (2001)
4. Huang, M., Haralick, R.M.: A probabilistic graphic model for recognizing np chunks from texts. In: The 22nd International Conference on the Computer Processing of Oriental languages. pp. 23–33 (2009)
5. Weischedel, R., Palmer, M., Marcus, M., Hovy, E.: Ontonotes release 2.0 with ontonotes db tool v. 0.92 beta and ontoviewer v.0.9 beta. In: http://www.bbn.com/NLP/OntoNotes (2007)
6. Marcus, M.P., Santorini, B., Marcinkiewicz, M.A.: Building a large annotated corpus of english: The penn treebank. Computational Linguistics 19(2), 313–330 (1994)
7. Tjong, E.F., Sang, K.: Introduction to the conll-2000 shared task: Chunking. In: Proceedings of CoNLL-2000. pp. 127–132 (2000)
8. Leacock, C., Towell, G., Voorhees, E.: Corpus based statistical sense resolution. In: Proceedings of the workshop on Human Language Technology. pp. 260 – 265 (1993)
9. Bruce, R., Wiebe, J.: Word-sense disambiguation using decomposable models. In: Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics. pp. 139–146 (1994)
10. Veenstra, J., den BoschJ, A.V.: Single-classifier memory-based phrase chunking. In: Preceedings of CoNLL-2000 and LLL-2000. pp. 157–159 (2000)
11. Sha, F., Fereira, F.: Shallow parsing with conditional random fields. In: Proceedings of HLT-NAACL. pp. 213–220 (2003)
12. Carreras, X., Mrquez, L.: Phrase recognition by filtering and ranking with perceptrons. In: the International Conference on Recent Advances on Natural Language Processing (2003)
13. Wu-Chieh, Wu, Lee, Y.S., Yang, J.C.: Robust and efficient multiclass svm models for phrase pattern recognition. Pattern Recognition 41, 2874–2889 (2008)
14. Huang, M., Haralick, R.M.: Recognizing patterns in texts. In: Pattern Recognition and Machine Vision. pp. 19–35. River (2010)
15. Levin, E., Sharifi, M., Ball, J.: Evaluation of utility of lsa for word sense discrimination. In: Preceedings of HLT-NAACL. pp. 77 – 80 (2006)
16. Church, K.W.: A stochastic parts program and noun phrase parser for unrestricted text. In: Proceedings of the second conference on Applied natural language processing. pp. 136 – 143 (1988)
17. Ramshaw, L.A., Marcus, M.P.: Text chunking using transformation-based learning. In: Proceedings of the Third Workshop on Very Large Corpora. pp. 82–94 (1995)
18. Abney, S., Abney, S.P.: Parsing by chunks. In: Principle-Based Parsing. pp. 257–278. Kluwer Academic Publishers (1991)
19. Hearst, M.A.: Noun homograph disambiguation using local context in large text corpora. In: Proceedings of the Seventh Annual Conference of the UW centre for the New OED and Text Research. pp. 1–22 (1991)
20. Gale, W., Church, K., Yarowsky, D.: A method for disambiguating word senses in a large corpus. In: Computers and the Humanities. pp. 415–439 (1992)

21. Leacock, C., Miller, G.A., Chodorow, M.: Using corpus statistics and wordnet relations for sense identification. Computational Linguist. 24, 147–165 (1998)
22. Yarowsky, D.: Decision lists for lexical ambiguity resolution: Application to accent restoration in spanish and frech. In: Preceedings of the 32nd Annual Meeting (1994)
23. Gildea, D., Jurafsky, D.: Automatic labelling of semantic roles. Computational Linguistics pp. 245–288 (2002)
24. Baldewein, U., Erk, K., Pad, S., Prescher, D.: Semantic role labeling with chunk sequences. In: Proceedings of CoNLL-2004 Shared Task (2004)
25. Cohn, T., Blunsom, P.: Semantic role labelling with tree conditional random fields. In: Proceedings of CoNLL-2005 Shared Task (2005)
26. Hacioglu, K.: A semantic chunking model based on tagging. In: Proceedings of HLT/NACCL-2004 (2004)
27. Hacioglu, K.: Semantic role labeling using dependency trees. In: Proceedings of Coling 2004. pp. 1273–1276. COLING, Geneva, Switzerland (Aug 23–Aug 27 2004)