A Probabilistic Graphical Model for Identifying Text Patterns

Minhua Huang and **Robert M. Haralick**
Department of Computer Science
Graduate Center
City University of New York

Abstract

We describe a generic probabilistic graphical model that works for recognizing three types of text patterns in a sentence: noun phrases; the meaning of an ambiguous word; and semantic arguments of a verb. We model dependency in the input symbol sequence rather than the output class identification sequence as in CRF's, HMM's, and MEMMs. This enables the determination of the optimal class for each input symbol to be obtained independent of the optimal class for every other input symbol. The algorithm needs no dynamic programming.

Experiments conducted on standard data sets show results just as good or better that the competing techniques. For instance, our method achieves an average precision of 97.7% and an average recall of 98.8% for recognizing noun phrases on WSJ data from Penn Treebank; an average accuracy of 81.12% for recognizing the six senses of the word $'line'$; an average precision of 92.96% and an average of recall of 94.94% for classifying semantic argument boundaries of a verb of a sentence on WSJ data from Penn Treebank and PropBank.

1 Introduction

Researchers have focused on using probabilistic graphical models, such as HMMs (Molina et al. 2002), MEMMs (MaCallum, Freitag, and Pereira 2000), or CRFs (Lafferty, MaCallum, and Pereira 2001) to recognize patterns in texts [1]. These models are derived from either a joint probability function or a conditional probability function for a sequence of categories given a sequence of symbols (associated with a sentence) under some conditional independence assumptions. One of their conditional independence assumptions is that for every i, given $i-1$ input classes, the true class identification of input symbol i depends only on the true class identification of previous class $i-1$. These assumptions might not be the best assumptions for capturing text patterns in a sentence. Moreover, for these graphical models, the maximum global 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 probability assumptions we have chosen to use leads to an algorithm whose memory requirements and computational complexity is less than dynamic programming and whose performance is just as good or better.

The competing methods use a graphical model that leads 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. Understanding the optimization as an optimization of expected gain, we can see that they use an implicit gain function which has a value of one if all the class assignments are correct and value of 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 class identification sequence. In effect what happens here is that a noisy or perturbed symbol at any position in the input sequence can produce a wrong category path for the whole sequence.

Our probabilistic graphical model improves this situation. In effect the model gives partial credit and yet takes dependencies into account. The model is derived from the probability function of a sequence of classes given a sequence of symbols by using the information carried by each current symbol, the association between the current symbol and the preceding symbol, and the association between the current symbol and the succeeding symbol. Understanding our optimization as an optimization of expected gain, our implicit gain function gives credit for each correct class assignment. It has a value of K if K of the class assignments are correct. The mathematical representation of the model can be found in Section 2 and the graphical representation of the model can be found in Figure 1.

Relative to competing methods, for recognizing a new symbol sequence N long, the time complexity is reduced from $O(M^2N)$ to $O(MN)$ while the memory complexity is reduced from $O(MN)$ to $O(M)$, where M is the number of possible classes. Numerical comparisons are shown on Section 2. Furthermore, when we make a mistake on one

Copyright © 2011, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

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

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.

Three text pattern recognition tasks are designed for testing our model. They are noun phrases, the meaning of a polysemous word , and semantic arguments of a verb in a sentence. In Section 3, we will formerly discuss these tasks. Moreover, we have conducted our tests on standard data. The results demonstrate that our method is effective. Some results exceed or approach the current state of the art. 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 related researches and discussions. The sixth section is the conclusion.

2 The Method

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 classes, $C = \{C_1, \ldots, C_M\}$ [2]. The task is to find a sequence of classes $< c_1^*, \ldots, c_N^* >$, $c_n^* \in C$ that best describes $S =< s_1, \ldots, s_N >$ in the sense that

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

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.

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 our model works because the dependency from neighboring classes to measurement 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

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

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

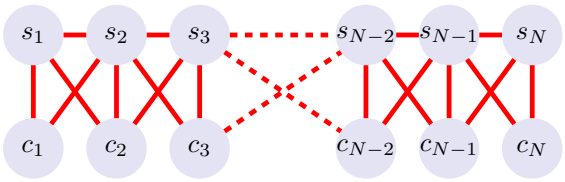

Figure 1: The conditional independence graph defining our graphical model.

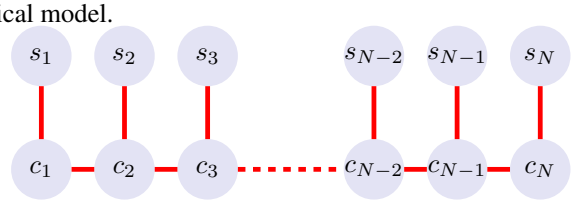

Figure 2: The usual conditional independence graph for Markov dependencies among the classes.

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]

[3] A clique is a maximal complete set of nodes. $\Lambda \subseteq V$ 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 $\Gamma_k = \Lambda_k \cap (\Lambda_1 \cup ..., \cup \Lambda_{k-1})$.

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. The pair of nodes in each edge 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. The product of the probabilities for the cliques divided by the product of the probabilities for the separators is the joint probability $p(c_1, \ldots, c_N, s_1, \ldots, s_N)$

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

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

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

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

Property 9. By property 7 and property 8, we find a sequence of category $< c_1^*, \ldots, c_N^* >$ for a sequence of symbols $< s_1, \ldots, s_N >$, we only need to find c_n^* for s_n individually. Note, the denominator in (1) is a constant. Therefore, it does not effect a decision for assigning c_i to s_i.

$$
\begin{aligned}
& < c_1^*, c_2^*, ..., c_N^* > = \\
& \underset{c_1 \in C}{argmax} \{p(s_2|s_1, c_1) p(s_1|c_1) p(c_1)\} \\
& \underset{c_2 \in C}{argmax} \{p(s_1|s_2, c_2) p(s_3|s_2, c_2) p(s_2|c_2) p(c_2)\} \\
& \ldots \\
& \underset{c_{N-1} \in C}{argmax} \{p(s_{N-2}|s_{N-1}, c_{N-1}) p(s_N|s_{N-1}, c_{N-1}) \\
& p(s_{N-1}|c_{N-1}) p(c_{N-1})\} \\
& \underset{c_N \in C}{argmax} \{p(s_{N-1}|s_N, c_N) p(s_N|c_N) p(c_N)\} \qquad (2)
\end{aligned}
$$

Complexity

Time Complexity For each symbol $s_n \in S$, we need to assign a c_n, s.t. $\mathcal{P}_{s_n|c_n} = max\{p(s_{n-1}|s_n, c_n)\ p(s_{n+1}|s_n, c_n)$

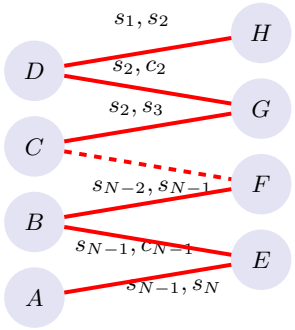

Figure 3: The junction tree shows the cliques in running order and the separators between them. The product of the probabilities for the cliques divided by the product of the probabilities for the separators is the joint probability $p(c_1, \ldots, c_N, s_1, \ldots, s_N)$. $A = \{s_{N-1}, s_N, c_N\}$, $B = \{s_{N-2}, s_{N-1}, c_{N-1}\}$, $C = \{s_2, s_3, c_3\}$, $D = \{s_1, s_2, c_2\}$, $E = \{s_{N-1}, s_N, c_{N-1}\}$, $F = \{s_{N-2}, s_{N-1}, c_{N-2}\}$, $G = \{s_2, s_3, c_2\}$, $H = \{s_1, s_2, c_1\}$.

$p(s_n|c_n)\ p(c_n)|c_n \in C\} = max\{P(s_n|c_n)|c_n \in C\}$. To compute a $P(s_n|c_n)$, we need to have four multiplications. To obtain the maximum probability value $\mathcal{P}_{s_n|c_n}$, we need to have $M-1$ comparisons. In the case of a sequence of N symbols, we have

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

where M is the cardinality of C and N is the length of symbol sequence S.

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 symbol. 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 classes 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 $p(c_1 \ldots c_N | s_1 \ldots s_n)$. In dynamic programming, the optimal class for the current symbol is obtained based on an optimal class of the previous symbol. Therefore, the optimal class for the last symbol is determined after the last symbol has been reached. The optimal class sequence needs to be determined by tracing back from the last optimal class assignment to the first optimal class assignment. For each symbol, the information of M classes needs to be stored. Hence, for a sequence of N symbols, the time complexity is $O(M^2 N)$ and the memory complexity is $O(M * N)$.

Ratio of Time Complexity

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

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. It is clear 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 compared to 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

Descriptions

In the previous section, we have conducted a numerical comparison on complexities between our model and other probabilistic graphic models such as HMMs and CRFs. Starting from this section, we will discuss three tasks and apply our method on these tasks. The tasks are: identifying noun phrases (NP chunking), identifying the meaning of a polysemous word (word sense disambiguation WSD), and identifying semantic arguments (ISA) 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 represents a polysemous word (called the context of the word); in ISA, it is a path related to a verb in a parse tree. Moreover, a set of categories in each task also has different representations. In NP chunking, it is a set of locations of a word related a noun phrase; in WSD, it is a set of predefined sense of the ambiguous word; in SAI, it is a set of directions from the current node to its neighbours. Then our method is applied to find an optimal category sequence associating with a symbol sequence with the maximum conditional probability. Then 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 appeared category in the optimal category sequence; and semantic arguments before and after a verb is obtained by finding siblings of each node in the optimal path.

Definition of Task 1

Let L be English language, V be a vocabulary associates with a parse tree defined by (), and T be a set of part-of-speech tags of V defined by Penn Treebank project (Marcus, Santorini, and Marcinkiewicz 1994). Let S be a sequence of symbols associated with a sentence, s.t. $S =< s_1, ..., s_i, ..., s_N >$, $s_i = (w_i, t_i)$, $w_i \in V,\ t_i \in T$. Let C be a set of categories, $C = \{C_1, C_2, C_3\}$, where C_1 represents a symbol is inside an noun phrase, C_2 represents a symbol is not in an noun phrase, C_3 represents a symbol starts at 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 $\rightarrow$ $\mathcal{B}' \Longleftrightarrow \mathcal{B}$

Formulating Task

- 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 $\mathcal{B}$.

Definition of Task 2

Let L, V, and T be defined in Task 1. Let S be a sequence of symbols associated with the context of a polysemous word, s.t. $S =< s_{t-i}...s_t...s_{t+j} >$, $s_t = (w_t, t_t)$, w_t is a polysemous word $w_t \in V$, $t_t \in T$. Let C be a set of categories, $C = \{C_1, C_2, \ldots, C_M\}$, where each C_m is a predefined sense of w_t.

Formulating Task

- finding a sequence of categories $< c_{t-i}^* \ldots c_t^* \ldots c_{t+j}^* >$ s.t.

$$< c_{t-i}^* \ldots c_t^* \ldots c_{t+j}^* >= \underset{c_{t-i}\ldots c_{t+j}}{argmax}\, p(c_{t-i} \ldots c_{t+j} | s_{t-i} \ldots s_{t+j})$$

- assigning w_t to c_t^{**} if and only if

$$c_t^{**} = max\{\#\{c_l^* | c_l^* = C_m, C_m \in C, l = t-i \ldots t+j\}\}$$

Definition of Task 3

Let $\mathcal{T} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{L})$ be a labeled rooted tree associated with a sentence, where $\mathcal{A}$ is defined by (Weischedel et al. 2007). Let $\pi = \{VB,\ VBN,\ VBG,\ VBZ,\ VBP,\ VBD\} \subseteq \mathcal{A}$. Let $C = \{C_1, C_2, C_3, C_4, C_5\}$ be a set of class labels, where C_1 represents from the current node to its parent, C_2 represents from the current node to its first child, C_3 represents from the current node to its right sibling, C_4 represents from the current node to its left sibling, and C_5 represents from the current node to itself.

Formulating Task

- For each $x \in \mathcal{V}$ and $\mathcal{L}(x) \in \pi$, find a sequence nodes $< b_1^*, b_2^*, \ldots, b_L^* >$, s.t.

$$< b_1^*, \ldots, b_L^* > \ = \ \underset{b_1,\ldots,b_L}{argmax}\, p(c_1, \ldots, c_L, b_1, \ldots, b_L)$$

- Find a path $\mathcal{P}(x) = \tau_1 \rightarrow \ldots, \rightarrow \tau_K$ from $< b_1^*, \ldots, b_L^* >$, s.t. for each $\tau_k \in \mathcal{P}(x)$, there is a least one child y, $\mathcal{L}(y) \in \pi$
- Find a child r, $\mathcal{L}(r) \notin \pi$ for each τ_k to form $R(x) = \{r_i | i = 1 \ldots M\}$

A labeled rooted forest $F(x) = \{T_1, \ldots T_M\}$ is formed. Each T_i is a labeled rooted tree, rooted as r_i, and induced by the descendants of r_i. Each labeled rooted tree T_i associates with a semantic arguments of x. $|F|$ is # of semantic arguments of x.

4 Empirical Results

Experiments Set Up

We test these three tasks on data sets, such as WSJ data from the Peen TreeBank and the PropBank!(Weischedel et al. 2007), CoNLL-2000 Shared Task Data (Tjong and Sang 2000), and data developed by (Leacock, Towell, and Voorhees 1993) and (Bruce and Wiebe 1994) for WSD. The evaluation metrics we have used for task 1 and task 3 are *precision , recall, and F-measure* (F_1). Moreover, the evaluation metric we have used for task 2 is *accuracy* . We have used 10-fold cross validation technique to obtain an average output.

Results on the First Task

We have conducted experiments for identifying NP chunks on two kinds of data sets. One is CoNLL-2000 Shared Task data set and the other is WSJ data set from Penn Treebank. On the first data set, three types of symbols are designed to test our model. They are the lexicon of a word, the POS tag of a word, and the lexicon and the POS tag of a word. The results are shown in the table 1. By comparing the results, we have noticed that if the model is built only on the lexical information, it has the lowest performance of F-measure 89.75%. The model's performance improved 3% on F-measure if it is constructed by POS tags. The model achieves the best performance of 95.59% on F-measure if both lexicon and POS tags are included.

The second data set we have used to verify our model is the WSJ data from Penn Treebank: *WSJ 0200 - WSJ 2999*. The main reason for using this data set is that we want to see whether the performance of our model can be improved when it is built on more data. In this experiment, based on the result we obtained from the CoNLL-2000 data, each symbol is the lexicon word+POS tag. In this case, the training set is seven times larger than the CoNLL-2000 shared task training data set. The testing set result is shown in Table 1. The standard deviations are the numbers inside the parentheses.

Compared with the results on these two data sets, we have noticed that the average precision is improved about 2.7% from 95.15% to 97.73% . The average recall is improved about 2.8% from 96.05% to 98.65%. The average F-measure is improved about 2.7% from 95.59% to 98.2% as the training sets expended into the seven times larger. This suggests a tradeoff between sizes of training sets and the performances of our model need to be considered.

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

Data	Symbol type	Precision %	Recall %	F-measure %
CoNLL	Lexicon+POS	95.15	96.05	95.59
	POS	92.27	93.76	92.76
	Lexicon	86.27	93.35	89.75
WSJ	Lexicon+POS	97.73 (0.19)	98.65 (0.14)	98.18 (0.08)

Results on the Second Task

We test our model 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 (Leacock, Towell, and Voorhees 1993) and (Bruce and Wiebe 1994). In these data sets, *line* and *interest* are polysemous nouns, *hard* is a polysemous adjective, and *serve* is a polysmous verb. In our experiment, *line* has 6 senses, *serve* has 4 senses, *hard* has 3 senses, *interest* has 3 senses (other 3 senses are omitted due to lack of instances). The test metric that we use is $accuracy$.

Table 2 shows the test results. In the table, $Mean$ represents the average accuracy, Std represents the stand deviation, $MaxA$ represents the maximum accuracy obtained from tests, and $MinA$ represents the minimum average accuracy obtained obtained from tests.

Table 2: The results on *line, serve, hard, interest* data

Ambiguous word	Senses	Mean %	Std %	MaxA %	MinA %
Line (n)	6	81.16	1.92	84.50	78.0
	3	85.25	2.13	91.70	81.05
Serve (v)	4	79.80	1.90	82.92	76.88
Hard (adj)	3	82.88	3.10	87.03	78.11
Interest (n)	3	92.10	2.21	95.50	86.00

By observing the results in Table2, we notice that, whether a ambiguous word is a noun, an adjective, or a verb, whether it has three senses, four senses, even six senses, our model achieves an average of accuracy 80%. This result is very encouraging and surpasses the results published by other researchers (Leacock, Towell, and Voorhees 1993) and (Levin, Sharifi, and Ball 2006). Moreover, by observing the outputs of two polysemous nouns *line* and *interest*, we found that as number of senses of a polysemous noun increases, the accuracy decreases. This suggests that nouns with larger number of senses are more difficult to recognize than nouns with small number of senses by our model. Furthermore, by observing the Means in column three, we notice that nouns are relatively easier to identify than adjectives or verbs. By observing the standard deviations in column four, the accuracies produced by our model on adjective data is more divergent than that of the nouns or verbs.

Results on the Third Task

In this experiment, we use WSJ data, section 00 from Pen Treebank and PropBank to test our model for identifying semantic arguments of a verb. For each sentence, Treebank provides a corresponding parse tree while PropBank provides corresponding semantic arguments of predicates in the sentence. The total number of trees in our data set is 233. These trees are generated by a statistic parser from corresponding sentences with an average accuracy 95% (Weischedel et al. 2007). In our data set, the number of predicates is 621. For each predicate, an average of three semantic arguments and a total of 1959 semantic argu-

ments are obtained. These semantic arguments of predicates in PropBank are generated by human labels.

Among the 233 trees, 208 trees are in the training set while 25 trees are are in the testing set. [**What happened to cross validation?**] By our model, total 196 labeled rooted subtrees associated with 63 verbs are constructed. Among these trees, 186 trees are associating correct semantic arguments of verbs while 10 trees have been misclassified. Among 10 trees, 4 trees are not semantic arguments while other 6 trees are some leaves associating with semantic arguments while other parts are not.

We also used our model to determine the semantic arguments of verbs. The results are shown in Table 3. Among 196 semantic arguments to be classified, 151 semantic arguments are correctly identified and 10 semantic arguments are classified incorrectly. Two kinds of errors can be noticed from this experiment. One of them is incorrect boundary. That is, some parts of a sequence does not belong to the semantic argument. Another error is a sequence is misclassified as a semantic argument.

Table 3: The test results on WSJ data

Data	Precision %	Recall %	F-measure %
WSJ	92.335 (0.6195)	94.1675 (0.5174)	93.2512 (0.4605)

5 Related Researches and More Comparisons

The graphical models used by most researchers consist of HMMs (MaCallum, Freitag, and Pereira 2000) (Molina et al. 2002), MEMMs(MaCallum, Freitag, and Pereira 2000), and CRFs(Lafferty, MaCallum, and Pereira 2001) (Sha and Fereira 2003). These models are built for obtaining an optimal corresponding category sequence $c =< c_1, \ldots, c_N >$ for a symbol sequence $s =< s_1, \ldots, s_N >$ by finding the maximum value for the joint probability $p(c, s)$ or the conditional probability $p(c|s)$. Each c_i or s_i is a node, ideally, for each c_i, $degree(c_i)$ should be $2N-1$. That is for any pair of nodes c_i, c_j and c_i, s_j, there is a link. However, assumptions are made in these methods. Figure 4 show these graphical models for symbol s_i. By examining these graphical models, we find that for each c_i, $degree(c_i)$ or ($in\ degree(c_i)$ are different. While $in\ degree(c_i)$ equals 1 for a HMM, $in\ degree(c_i)$ equals 2 for a $MEMM$, $degree(c_i)$ equals 2 for a CRF, $degree(c_i)$ equals 3 for the model presented by the paper. Our model has more links suggesting it has fewer assumptions. Moreover, there is a link from c_{i-1} to c_i in any model except our model. In our model, instead of using c_{i-1} to predict c_i, we use s_{i-1} and s_{i+1} to predict c_i. We believe that c_i can be better predicted from s_{i-1} and s_{i+1} rather than c_{i-1} when these symbols consist of several kinds of information. For example, in the case of NP chunking, POS tag information carried on a symbol is much more useful than the class information assigned to the previous symbol.

By observing the graphical representation of each model, we notice that a HMM makes two conditional independence assumptions. First, given its previous class identification, the current class identification is independent of other classes. Moreover, given its current class, the symbol is independent of other classes and symbols. $MEMM$ makes one conditional independence assumption. Given its previous class identification and the current symbol, the current class identification is independent of other classes and symbols. A CRF makes the same two conditional assumptions as a HMM. The model presented in this paper makes one conditional independence assumption. Given the current, the preceding, and the succeeding symbol, the current class identification is independent of other class identifications and symbols.

Moreover, mathematical notations of directed graphic models can be directly built from it graphical representation for a joint probability. Then a conditional probability can be found by applying Bayes' theorem. The models representing a HMM and a $MEMM$ are directed models. The mathematical notation of the HMM is $p(c, s) = \prod_{i=1}^{N} p(s_i|c_i)p(c_i|c_{i-1})$ while the mathematical notation of the $MEMM$ is $p(c|s) = \frac{\prod_{i=1}^{N} p(c_i|c_{i-1}s_i)}{\sum_{c_n \in C} \prod_{n=1}^{N} p(c_n|c_{n-1}s_n)}$. Furthermore, mathematical notations of undirected models can be derived from the notion of the product of cliques divided by the product of separators in a graph for a joint probability. The same method is applied as in directed graphic model for a conditional probability. For example, the CRF is represented as $p(c|s) = \frac{\prod_{i=1}^{N} p(s_i|c_i)p(c_i|c_{i-1})}{\sum_{c_n \in C} \prod_{n=1}^{N} p(s_n|c_n)p(c_n|c_{n-1})}$. The model discussed in this paper is represented by $p(c|s) = \frac{\prod_{i=1}^{N} p(s_{i-1}|s_i,c_i)p(s_{i+1}|s_i,c_i)p(s_i|c_i)p(c_i)}{\sum_{c_n \in C} \prod_{n=1}^{N} p(s_{n-1}|s_n,c_n)p(s_{n+1}|s_n,c_n)p(s_n|c_n)p(c_n)}$. Compared with these mathematical expressions for $p(c|s)$, CRF has two terms, $MEMM$ has one term, and our model has four terms in the numerator. Moreover, different from the other two models, none of the terms in our model is related to c_{i-1}.

A number of NP chunking, WSD, semantic role labelling methods has been developed over the years. The methods for NP chunking are (Church 1988) (Ramshaw and Marcus 1995) (Molina et al. 2002) (Abney and Abney 1991) (Wu-Chieh et al. 2008), the methods for WSD are (Hearst 1991) (Gale, Church, and Yarowsky 1992) (Leacock, Towell, and Voorhees 1993) (Leacock, Miller, and Chodorow 1998) (Yarowsky 1994), and the methods for semantic role labelling are (Gildea and Jurafsky 2002) (Baldewein et al. 2004) (Cohn and Blunsom 2005). In contrast to these methods, we apply our new model to identify a noun phrase, the meaning of a polysemous word, and semantic arguments of a verb in a sentence. In NP chunking task, we adopt Ramshaw's idea (Ramshaw and Marcus 1995) of designing three possible class identifications for a word in a sentence to determine whether the word is inside a NP chunk, outside a NP chunk, or start a new NP chunk. In WSD task, in contrast with other WSD methods, the polysemous word

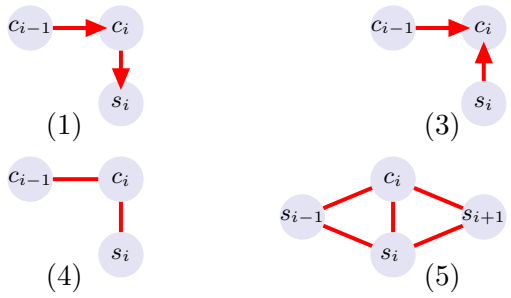

Figure 4: (1): a HMM model $p(s,c) = \prod_{i=1}^{N} p(s_i|c_i)p(c_i|c_{i-1})$, (3): a MEMM model $p(c|s) = \frac{\prod_{i=1}^{N} p(c_i|c_{i-1},s_i)}{\sum_{c_n \in C} \prod_{n=1}^{N} p(c_n|c_{n-1}s_n)}$, (4): a CRF model $p(c|s) = \frac{\prod_{i=1}^{N} p(s_i|c_i)p(c_{i-1}|c_i)}{\sum_{c_n \in C} \prod_{n=1}^{N} p(s_n|c_n)p(c_n|c_{n-1})}$, and (5): the model presented by this paper $p(c|s) = \frac{\prod_{i=1}^{N} p(s_{i-1}|s_i,c_i)p(s_{i+1}|s_i,c_i)p(s_i|c_i)p(c_i)}{\sum_{c_n \in C} \prod_{n=1}^{N} p(s_{n-1}|s_n,c_n)p(s_{n+1}|s_n,c_n)p(s_n|c_n)p(c_n)}$, where $s =< s_1, \ldots, s_N >, < c_1, \ldots, c_N >$

is represented by a sequences of symbols, each symbol is a ordered pair (the lexicon and the POS tag of a word). Each symbol is represented by it's left symbol and right symbol. Moreover, in the semantic argument identification task, we created our own algorithm. The experiments in the section 4 show our model achieves better performance than HMMs and CRFs (Sha and Fereira 2003).

6 Conclusions

A generic probabilistic graphical model has been discussed throughout this paper. It has an unique graphic representation and mathematical notation which are different from other existed graphic models such as HMMs, CRFs, and $MEMM$s. It does not need to employ dynamic programming for obtaining a sequence of optimal class assignments for a sequence of symbols. As a consequence, it requires less operating time and less memory spaces than competing techniques. Moreover, because a sequence of optimal class assignments for a sequence of symbols is determined by finding the optimal class assignment for each symbol independently, the misclassification for the sequence of class assignments can be reduced. Performance of three different types of tasks: noun phase identification, word sense disambiguation, and semantic arguments of a verb classification, have demonstrated the effectiveness of our graphical model.

References

Abney, S., and Abney, S. P. 1991. Parsing by chunks. In *Principle-Based Parsing*, 257–278. Kluwer Academic Publishers.

Baldewein, U.; Erk, K.; Pad, S.; and Prescher, D. 2004. Semantic role labeling with chunk sequences. In *Proceedings of CoNLL-2004 Shared Task*.

Bruce, R., and Wiebe, J. 1994. Word-sense disambiguation using decomposable models. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, 139–146.

Church, K. W. 1988. A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the second conference on Applied natural language processing*, 136 – 143.

Cohn, T., and Blunsom, P. 2005. Semantic role labelling with tree conditional random fields. In *Proceedings of CoNLL-2005 Shared Task*.

Gale, W.; Church, K.; and Yarowsky, D. 1992. A method for disambiguating word senses in a large corpus. In *Computers and the Humanities*, 415–439.

Gildea, D., and Jurafsky, D. 2002. Automatic labelling of semantic roles. *Computational Linguistics* 245–288.

Hearst, M. A. 1991. 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*, 1–22.

Lafferty, J.; MaCallum, A.; and Pereira, F. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of 18th International Conf. on Machine Learning*, 282–289.

Leacock, C.; Miller, G. A.; and Chodorow, M. 1998. Using corpus statistics and wordnet relations for sense identification. *Computational Linguist.* 24:147–165.

Leacock, C.; Towell, G.; and Voorhees, E. 1993. Corpus based statistical sense resolution. In *Proceedings of the workshop on Human Language Technology*, 260 – 265.

Levin, E.; Sharifi, M.; and Ball, J. 2006. Evaluation of utility of lsa for word sense discrimination. In *Preceedings of HLT-NAACL*, 77 – 80.

MaCallum, A.; Freitag, D.; and Pereira, F. 2000. Maximum entropy markov models for information extraction and segmentation. In *Proceedings of 17th International Conf. on Machine Learning*, 591–598.

Marcus, M. P.; Santorini, B.; and Marcinkiewicz, M. A. 1994. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics* 19(2):313–330.

Molina, A.; Pla, F.; tics, D. D. S. I.; Hammerton, J.; Osborne, M.; Armstrong, S.; and Daelemans, W. 2002. Shallow parsing using specialized hmms. *Journal of Machine Learning Research* 2:595–613.

Ramshaw, L. A., and Marcus, M. P. 1995. Text chunking using transformation-based learning. In *Proceedings of the Third Workshop on Very Large Corpora*, 82–94.

Sha, F., and Fereira, F. 2003. Shallow parsing with conditional random fields. In *Proceedings of HLT-NAACL*, 213–220.

Tjong, E. F., and Sang, K. 2000. Introduction to the conll-2000 shared task: Chunking. In *Proceedings of CoNLL-2000*, 127–132.

Weischedel, R.; Palmer, M.; Marcus, M.; and Hovy, E. 2007. 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*.

Wu-Chieh; Wu; Lee, Y.-S.; and Yang, J.-C. 2008. Robust and efficient multiclass svm models for phrase pattern recognition. *Pattern Recognition* 41:2874–2889.

Yarowsky, D. 1994. Decision lists for lexical ambiguity resolution: Application to accent restoration in spanish and frech. In *Preceedings of the 32nd Annual Meeting*.