Tree-Structured Conditional Random Fields for Semantic Annotation*

Jie Tang[1], Mingcai Hong[1], Juanzi Li[1], and Bangyong Liang[2]

[1] Department of Computer Science, Tsinghua University
12#109, Tsinghua University, Beijing, 100084. China
j-tang02@mails.tsinghua.edu.cn,
{hmc, ljz}@keg.cs.tsinghua.edu.cn
[2] NEC Labs China
11th Floor, Innovation Plaza, Tsinghua Science Park, Beijing, 100084, China
liangbangyong@research.nec.com.cn

Abstract. The large volume of web content needs to be annotated by ontologies (called Semantic Annotation), and our empirical study shows that strong dependencies exist across different types of information (it means that identification of one kind of information can be used for identifying the other kind of information). Conditional Random Fields (CRFs) are the state-of-the-art approaches for modeling the dependencies to do better annotation. However, as information on a Web page is not necessarily linearly laid-out, the previous linear-chain CRFs have their limitations in semantic annotation. This paper is concerned with semantic annotation on hierarchically dependent data (*hierarchical semantic annotation*). We propose a Tree-structured Conditional Random Field (TCRF) model to better incorporate dependencies across the hierarchically laid-out information. Methods for performing the tasks of model-parameter estimation and annotation in TCRFs have been proposed. Experimental results indicate that the proposed TCRFs for hierarchical semantic annotation can significantly outperform the existing linear-chain CRF model.

1 Introduction

Semantic web requires annotating existing web content according to particular ontologies, which define the meaning of the words or concepts in the content [1].

In recent years, automatic semantic annotation has received much attention in the research community. Many prototype systems have been developed using information extraction methods. The methods usually convert a document into an 'object' sequence and then identify a sub-sequence of the objects that we want to annotate (i.e. targeted instance). (Here, the object can be either natural language units like token and text line, or structured units indicated by HTML tags like "<table>" and "<image>"). The methods make use of the contexts information that is previous to and next to the target instances for the identification task.

Empirical study shows that strong dependencies exist across different types of targeted instances. The type of dependencies varies in different kinds of documents

* Supported by the National Natural Science Foundation of China under Grant No. 90604025.

I. Cruz et al. (Eds.): ISWC 2006, LNCS 4273, pp. 640 – 653, 2006.

and different applications, for instance, in Part-Of-Speech (POS) tagging from NLP, the dependencies between POS labels can be linear-chain [20]; while in object extraction from web pages, the dependencies can be two-dimensional [26].

Conditional Random Fields (CRFs) are the state-of-the-art approaches in information extraction taking advantage of the dependencies to do better annotation, compared with Hidden Markov Model (HMMs) [8] and Maximum Entropy Markov Model (MEMMs) [17]. However, the previous linear-chain CRFs only model the linear-dependencies in a sequence of information, and is not able to model hierarchical dependencies [14] [26].

In this paper, we study the problem of *hierarchical semantic annotation*. In hierarchical semantic annotation, targeted instances on a web page can have hierarchical dependencies with each other, for example, an instance may have a dependency with another instance in the upper level (i.e. child-parent dependency), have a dependency with one in the lower level (i.e. parent-child dependency), or have a dependency with one in the same level (i.e. sibling dependency).

To better incorporate dependencies across hierarchically laid-out information, a Tree-structured Conditional Random Field (TCRF) model has been proposed in this paper. We present the graphical structure of the TCRF model as a tree (see Figure 3) and reformulate the conditional distribution by defining three kinds of edge features. As the tree structure can be cyclable, exact inference in TCRFs is expensive. We propose to use the Tree Reparameterization algorithm to compute the approximate marginal probabilities for edges and vertices. Experimental results indicate that the proposed TCRF models perform significantly better than the baseline methods for hierarchical semantic annotation.

The rest of the paper is organized as follows. In Section 2, we introduce related work. In Section 3, we formalize the problem of hierarchical semantic annotation. In Section 4, we describe our approach to the problem. Section 5 gives our experimental results. We make some concluding remarks in Section 6.

2 Related Work

Semantic annotation is an important area in semantic web. Many research efforts have been made so far. However, much of the previous work views web page as an 'object' sequence and focuses on annotating web page by using existing information extraction techniques. To the best of our knowledge, no previous work has been done on semantic annotation of hierarchically laid-out information.

1. Semantic Annotation Using Rule Induction

Many semantic annotation systems employ rule induction to automate the annotation process (also called as 'wrapper' induction, see [13]).

For example, Ciravegna et al propose a rule learning algorithm, called LP^2, and have implemented an automatic annotation module: Amilcare [4]. The module can learn annotation rules from the training data. Amilcare has been used in several annotation systems, for instance, S-CREAM [12]. See also [18] [22].

The rule induction based method can achieve good results on the template based web pages. However, it cannot utilize dependencies across targeted instances.

2. Semantic Annotation as Classification
The method views semantic annotation as a problem of classification, and automates the process by employing statistical learning approaches. It defines features for candidate instances and learns a classifier that can detect the targeted instance from the candidate ones.

For example, SCORE Enhancement Engine (SEE) supports web page annotation by using classification model [11]. It first classifies the web page into a predefined taxonomy; then identifies name entities in the classified web pages; finally recognizes the relationships between the entities via analysis of the web content.

The classification based method can obtain good results on many annotation tasks. However, it cannot also use the dependencies across different targeted instances.

3. Semantic Annotation as Sequential Labeling
Different from the rule induction and the classification methods, sequential labeling enables describing dependencies between targeted instances. The dependencies can be utilized to improve the accuracy of the annotation.

For instance, Reeve et al propose to utilize Hidden Markov Model (HMM) in semantic annotation [19]. As a generative model, HMM needs enumerate all possible observation sequences, and thus requires the independence assumption to ease the computation. Despite of its usefulness, limited research has been done using the sequential labeling method in semantic annotation.

4. Information Extraction Methods
Many information extraction methods have been proposed. Hidden Markov Model (HMM) [8], Maximum Entropy Markov Model (MEMM) [17], Conditional Random Field (CRF) [14], Support Vector Machines (SVM) [6], and Voted Perceptron [5] are widely used information extraction models.

Some of the methods only model the distribution of contexts of target instances and do not model dependencies between the instances, for example, SVM and Voted Perceptron. Some other methods can model the linear-chain dependencies, for example, HMM, MEMM, and CRF.

Recently, several research efforts have been also made for modeling the non-linear dependencies. For instance, Sutton et al propose Dynamic Conditional Random Fields (DCRFs) [21]. As a particular case, a factorial CRF (FCRF) was used to jointly solve two NLP tasks (noun phrase chunking and Part-Of-Speech tagging) on the same observation sequence. Zhu et al propose 2D Conditional Random Fields (2D CRFs) [26]. 2D CRFs is also a particular case of CRFs. It is aimed at extracting object information from two-dimensionally laid-out web pages. See also [3].

3 Hierarchical Semantic Annotation

For semantic annotation, we target at detecting targeted instances from a document and annotating each of the instances by concepts/attributes of a particular ontology.

Information on a web page can be laid-out differently, for example, product information on a web page is typically two-dimensionally laid-out [26]; and in Natural Language Processing, word's POS (Part-Of-Speech) can be organized as a sequence, and thus viewed as linearly laid-out [20]. In this paper, we concentrate on

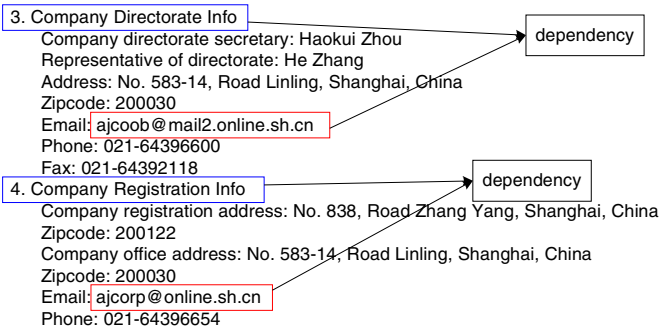

Fig. 1. Example of Hierarchical laid-out information

semantic annotation on hierarchically laid-out information that we name as *hierarchical semantic annotation*. In hierarchical semantic annotation, information is laid-out hierarchically. An example is shown in Figure 1.

In Figure 1, there are two emails. One is the email of the company directorate secretary and the other is the email of the company registration office. Previous linear-chain models such as linear-chain CRFs view the text as a token-sequence (or text-line sequence) and assign a label to each token in the sequence by using neighborhood contexts (i.e. information previous to and next to the targeted instance).

However, the neighborhood contexts of the two emails are the same with each other in the linear-chain token-sequence. The neighborhood contexts include tokens previous to and next to the emails. Tokens previous to the two emails are both "Email: " and tokens next to them are also identical "<return>Phone:". It is inevitable that the linear-chain CRF models will fail to distinguish them from each other.

By further investigation, we found that the information is hierarchically laid-out: the two emails are respectively located in two sections and each section has a heading, i.e. "3. Company directorate Info" and "4. Company Registration Info". The two headings can be used to distinguish the two emails from each other. We call it as hierarchically laid-out information when existing hierarchical dependencies across information and call the task of semantic annotation on hierarchically laid-out information as *hierarchical semantic annotation*. In hierarchical semantic annotation, we target at improving the accuracy of semantic annotation by incorporating hierarchical dependencies. For instance, in Figure 1, we can use the upper level information "3. Company directorate Info" to help identify the email "ajcoob@ mail2.online.sh.cn".

4 Tree-Structured Conditional Random Fields

In this section, we first introduce the basic concepts of Conditional Random Fields (CRFs) and introduce the linear-chain CRFs, and then we explain a Tree-structured CRF model for hierarchically laid-out information. Finally we discuss how to perform parameter estimation and annotation in TCRFs.

4.1 Linear-Chain CRFs

Conditional Random Fields are undirected graphical models [14]. As defined before, X is a random variable over data sequences to be labeled, and Y is a random variable over corresponding label sequences. All components Y_i of Y are assumed to range over a finite label alphabet Y. CRFs construct a conditional model $p(Y|X)$ with a given set of features from paired observation and label sequences.

CRF Definition. *Let $G = (V, E)$ be a graph such that $Y=(Y_v)_{v\in V}$, so that Y is indexed by the vertices of G. Then (X, Y) is a conditional random field in case, when conditioned on X, the random variable Y_v obey the Markov property with respect to the graph: $p(Y_v|X, Y_w, w\neq v) = p(Y_v|X, Y_w, w\backsim v)$, where $w\backsim v$ means that w and v are neighbors in G.*

Thus, a CRF is a random field globally conditioned on the observation X. Linear-chain CRFs were first introduced by Lafferty et al [14]. The graphical structure of linear-chain CRFs is shown in Figure 2.

By the fundamental theorem of random fields [10], the conditional distribution of the labels y given the observations data x has the form

$$p(y \mid x) = \frac{1}{Z(x)} \exp\left(\sum_{e\in E, j} \lambda_j t_j(e, y|_e, x) + \sum_{v\in V, k} \mu_k s_k(v, y|_v, x) \right) \quad (1)$$

where x is a data sequence, y is a label sequence, and $y|_e$ and $y|_v$ are the set of components of y associated with edge e and vertex v in the linear chain respectively; t_j and s_k are feature functions; parameters λ_j and μ_k correspond to the feature functions t_j and s_k respectively, and are to be estimated from the training data; $Z(x)$ is the normalization factor, also known as partition function.

4.2 Tree-Structured Conditional Random Fields (TCRFs)

Linear-chain CRFs cannot model dependencies across hierarchically laid-out information. This paper proposes a Tree-structured Conditional Random Field (TCRF) model which is also a particular case of CRFs. The graphical structure of TCRFs is a tree (see Figure 3).

From Figure 3, we see that y_4 is the parent vertex of y_2 and y_{n-1} (for simplifying description, hereafter we use parent-vertex to represent the upper-level vertex and use child-vertex to represent the lower-level vertex of the current vertex). TCRFs can model the parent-child dependencies, e.g. y_4-y_2 and y_4-y_{n-1}. Furthermore, y_2 and y_{n-1} are in the same level, which are represented as a sibling dependency in TCRFs.

Here we also use X to denote the random variable over observations, and Y to denote the random variable over the corresponding labels. Y_i is a component of Y at the vertex i. Same as the linear-chain CRFs, we consider one vertex or two vertices as a clique in TCRFs. TCRFs can also be viewed as a finite-state model. Each variable Y_i has a finite set of state values and we assume the one-to-one mapping between states and labels. And thus dependencies across components Y_i can be viewed as transitions between states.

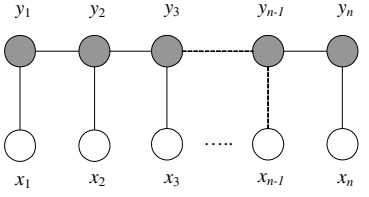

Fig. 2. The Graphical structure of Linear-chain CRFs

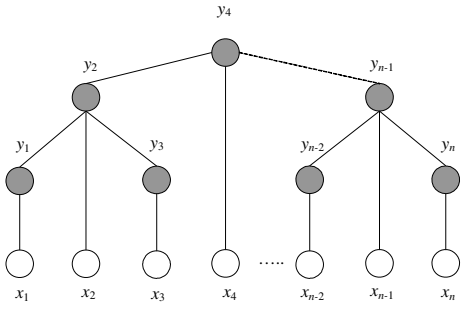

Fig. 3. The Graphical structure of TCRFs

Let (y_p, y_c) be the dependency between a parent- and a child-vertices, (y_c, y_p) be the dependency between a child- and a parent-vertices, and (y_s, y_s) be the dependency between sibling vertices. A TCRF model, as a particular case of CRFs, has the form

$$p(y \mid x) = \frac{1}{Z(x)} \exp\left(\sum_{e \in \{E^{pc}, E^{cp}, E^{ss}\}, j} \lambda_j t_j(e, y|_e, x) + \sum_{v \in V, k} \mu_k s_k(v, y|_v, x) \right) \tag{2}$$

where E^{pc} denotes the set of (y_p, y_c), E^{cp} denotes the set of (y_c, y_p), and E^{ss} denotes the set of (y_s, y_s). t_j and s_k are feature functions.

TCRFs have the same form as that of linear-chain CRFs except that in TCRFs the edges include parent-child edges, child-parent edges, and sibling-vertices edges while in CRFs the edges mean the transitions from the previous-state to the current-state.

In semantic annotation, the observation x in TCRFs can correspond to a document (as the example shown in Figure 1). The label y thus corresponds to the annotation result for the document. Specifically, x_i is a token in the document, and label y_i is the annotation result (called label) to the token, where the label corresponds to either one of the concept/attribute from a particular ontology or none.

4.3 Parameter Estimation

The parameter estimation problem is to determine the parameters $\Theta=\{\lambda_1, \lambda_2, \ldots; \mu_k, \mu_{k+1}, \ldots\}$ from training data $D=\{(x^{(i)}, y^{(i)})\}$ with empirical distribution $\tilde{p}(x, y)$. More specifically, we optimize the log-likelihood objective function with respect to a conditional model $p(y|x, \Theta)$:

$$L_\Theta = \sum_i \tilde{p}(x^{(i)}, y^{(i)}) \log p_\Theta(y^{(i)} \mid x^{(i)}) \tag{3}$$

In the following, to facilitate the description, we use f to denote both the edge feature function t and the vertex feature function s; use c to denote both edge e and vertex v; and use λ to denote the two kinds of parameters λ and μ. Thus, the derivative of the object function with respect to a parameter λ_j associated with clique index c is:

$$\frac{\delta L_{\Theta}}{\delta \lambda_j} = \sum_i \left[\sum_c f_j(c, y_{(c)}^{(i)}, x^{(i)}) - \sum_y \sum_c p(y_{(c)} \mid x^{(i)}) f_j(c, y_{(c)}, x^{(i)}) \right] \tag{4}$$

where $y^i_{(c)}$ is the label assignment to clique c in $x^{(i)}$, and $y_{(c)}$ ranges over label assignments to the clique c. We see that it is the factors $p(y_{(c)}|x^{(i)})$ that require us to compute the marginal probabilities. The factors $p(y_{(c)}|x^{(i)})$ can be again decomposed into four types of factors: $p(y_p, y_c|x^{(i)})$, $p(y_c, y_p|x^{(i)})$, $p(y_s, y_s|x^{(i)})$, and $p(y_t|x^{(i)})$, as we have three types of dependencies (described as edges here) and one type of vertex. Moreover, we also need to compute the global conditional probability $p(y^{(i)}|x^{(i)})$.

The marginal probabilities can be done using many inference algorithms for undirected model (for example, Belief Propagation [25]). However, as the graphical structure in TCRFs can be a tree with cycles, exact inference can be expensive in TCRFs. We propose utilizing the Tree Reparameterization (TRP) algorithm [24] to compute the approximate probabilities of the factors. TRP is based on the fact that any exact algorithm for optimal inference on trees actually computes marginal distributions for pairs of neighboring vertices. For an undirected graphical model over variables x, this results in an alternative parameterization of the distribution as:

$$p(x) = \frac{1}{Z} \prod_{s \in V} \varphi_s(x_s) \prod_{(s,t) \in V} \varphi_{st}(x_s, x_t) \Rightarrow p(x) = \prod_{s \in V} p_s(x_s) \prod_{(s,t) \in V} \frac{p_{st}(x_s, x_t)}{p_s(x_s) p_t(x_t)} \tag{5}$$

where $\varphi_s(x_s)$ is the potential function on single-vertex x_s and $\varphi_{st}(x_s, x_t)$ is the potential function on edge (x_s, x_t); and Z is the normalization factor.

TRP consists of two main steps: Initialization and Updates. The updates are a sequence of $T^n \rightarrow T^{n+1}$ on the undirected graph with edge set E, where T represents the set of marginal probabilities maintained by TRP including single-vertex marginals $T_u^{n+1}(x_u)$ and pairwise joint distribution $T_{uv}^{n+1}(x_u, x_v)$; and n denotes the iteration number. The TRP algorithm is summarized in Figure 4. (The algorithm is adopted from [21]).

1. Initialization: for every node u and every pair of nodes (u, v), initialize T^0 by $T_u^0 = \kappa\varphi_u$ and $T_{uv}^0 = \kappa\varphi_{uv}$, with κ being a normalization factor.

1. TRP Updates: for i=1, 2, …, do:

- Select some spanning tree $\Gamma^i \in R$ with edge set E^i, where $R=\{\Gamma^i\}$ is a set of spanning trees.
- Use any exact algorithm, such as belief propagation, to compute exact marginals $p^i(x)$ on Γ^i. For all $(u, v) \in E^i$, set

$$T_u^{i+1}(x_u) = p^i(x_u), \; T_{uv}^{i+1}(x_u, x_v) = \frac{p^i(x_u, x_v)}{p^i(x_u) p^i(x_v)}$$

- Set $T_{uv}^{i+1} = T_{uv}^{i}$ for all $(u, v) \in E/E^i$ (i.e. all the edges not included in the spanning treeΓ^i).
- Stop if termination conditions are met.

Fig. 4. The TRP Algorithm

So far, the termination conditions are defined as: if the maximal change of the marginals is below a predefined threshold or the update times exceed a predefined number (defined as 1000 in our experiments), then stop the updates. When selecting spanning trees $R=\{\Gamma^i\}$, the only constraint is that the trees in R cover the edge set of the original undirected graph U. In practice, we select trees randomly, but we select first edges that have never been used in any previous iteration.

Finally, to reduce overfitting, we define a spherical Gaussian weight prior $p(\Theta)$ over parameters, and penalize the log-likelihood object function as:

$$L_\Theta = \sum_i p(x^{(i)}, y^{(i)}) \log p_\Theta(y^{(i)} \mid x^{(i)}) - \frac{\|\lambda\|^2}{2\sigma^2} + const \tag{6}$$

with gradient

$$\frac{\delta L_\Theta}{\delta \lambda_j} = \sum_i \left[\sum_c f_j(c, y^{(i)}_{(c)}, x^{(i)}) - \log Z(x^{(i)}) \right] - \frac{\lambda_j}{\sigma^2} \tag{7}$$

where *const* is a constant.

The function L_Θ is convex, and can be optimized by any number of techniques, as in other maximum-entropy models [14] [2]. In the result below, we used gradient-based L-BFGS [15], which has previously outperformed other optimization algorithms for linear-chain CRFs [20].

4.4 Annotation

Annotation (also called as 'labeling') is the task to find labels y^* that best describe the observations x, that is, $y^*=\max_y p(y|x)$. Dynamic programming algorithms are the most popular methods for this problem. However, it is difficult to directly adopt it for annotation in TCRFs. Two modes exist in the annotation of TCRFs: known structure and unknown structure. In the first mode, the hierarchical structure is known, for example, one can use the document logic structure to infer the hierarchical structure. Hence, we can use the TRP algorithm to compute the maximal value of $p(y|x)$. In the second mode, the hierarchical structure is unknown. We used a heuristics based method and performed the annotation for a given observation x_i as follows: (1) use vertex features to preliminary identify the possible labels for each vertex; (2) incorporate the edge features to compute all possible label results (that is, to enumerate all possible hierarchical structures) for an observations x_i; (3) use equation (6) to compute the log-likelihood for each structure and choose one as the annotation result y^* that has the largest log-likelihood. (The annotation in the second mode can be expensive, the issue and some of the related problems are currently researching, and will be reported elsewhere.)

4.5 Using TCRFs for Semantic Annotation

Currently, there is still not sufficient Semantic Web content available. Existing web content should be upgraded to Semantic Web content. Our proposed TCRFs can be used to create an annotation service, especially for the hierarchically laid-out data.

The output of the service will be generated according to the language pyramid of Semantic Web, so that agents can automatically handle the semantic information.

TCRFs can be used in two ways. One is to extract the web content (that we are interested) from its source, annotate it by an ontology, and store it in knowledge base. The other is to add the annotation results into the web page.

5 Experimental Results

5.1 Data Sets and Evaluation Measures

1. Data Sets

We carried out the experiments on two data sets, one synthetic and one real. For the real data set, we collected company annual reports from Shanghai Stock Exchange (http://www.sse.com.cn). We randomly chose in total 3,726 annual reports (in Chinese) from 1999 to 2004. To evaluate the effectiveness of our approach, we extracted the Section "Introduction to Company" from each annual report for experiments. For Chinese tokenization, we used a toolkit proposed in [16].

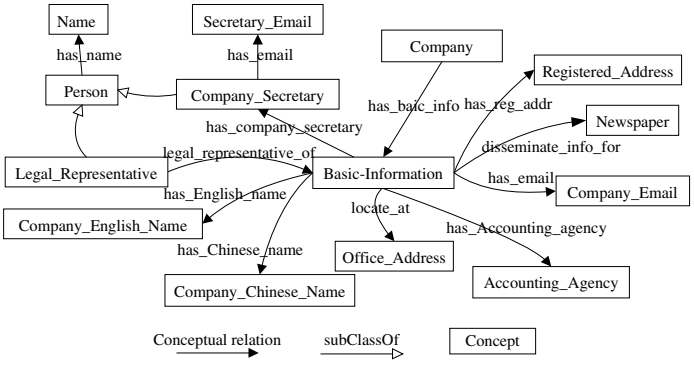

Fig. 5. Ontology of company annual report

Figure 5 shows the ontological information (that is we need to annotate) defined for the annual report. In total, fourteen concepts were defined in the ontology and the annotation task is to find instances for the fourteen concepts. Most of the concepts have hierarchical dependencies. Human annotators conducted annotation on all annual reports.

We also constructed a synthetic data set. The data set contains 62 company annual reports chosen from the real data set. In this data set, four concepts are defined only: "Company_Secretary", "Secretary_Email", "Registered_Address", and "Company_ Email". Where the first two concepts and the last two concepts have the parent-child dependencies respectively and the concepts "Company_Secretary", "Registered-_Address" have the sibling dependency. Every report in the data set exclusively has the four types of instances and the four instances are organized hierarchically.

2. Features in Annotation Models

Table 1 indicates the features used in the annotation models.

Table 1. Features used in the annotation models

Category	Feature
Edge Feature	$f(y_p, y_c), f(y_c, y_p), f(y_s, y_s)$
Vertex Feature	$\{w_i\}, \{w_p\}, \{w_c\}, \{w_s\}$
	$\{w_p, w_i\}, \{w_c, w_i\}, \{w_s, w_i\}$

Given the j-th vertex in the observation x_i, $f(y_p, y_c)$, $f(y_c, y_p)$, and $f(y_s, y_s)$ represent whether the current vertex has a parent-child dependency with a parent vertex, whether it has a child-parent dependency with a child vertex, and whether it has a sibling dependency with a sibling vertex, respectively. For vertex features, each element in $\{w_i\}$ represents whether the current vertex contains the word w_i. Similarly, each element in $\{w_p\}$, $\{w_c\}$, and $\{w_s\}$ represents whether the parent vertex of the current vertex contains word w_p whether its child vertices contain w_c, and whether its sibling vertices contain w_s, respectively. $\{w_p, w_i\}$ represents whether the current vertex contains word w_i and its parent vertex contains word w_p. To save time in some of our experiments, we omitted the vertex features that appear only once.

3. Evaluation Measures

In all the experiments of annotation, we conducted evaluations in terms of precision, recall, and F1-measure. By comparison of the previous work, we also give statistical significance estimates using Sign Test [9].

4. Baselines

To evaluate our model's effectiveness of incorporating hierarchical dependencies for semantic annotation, we choose linear-chain CRFs as the baseline models for their outstanding performance over other sequential models. The linear-chain CRF models are trained using the same features as those in table 1 (the only difference is that the linear-chain CRFs uses the linear edge features and the TCRFs uses the hierarchical edge features).

We also compared the proposed method with the classification based annotation method, which is another popular annotation method. The classification based method treats a company annual report as a sequence of text lines, employs two classification models to respectively identify the start line and the end line of a target instance, and then view lines that between the start line and the end line as a target (see [7] and [23] for details). In the experiments, we use Support Vector Machines (SVMs) as the classification models. In the SVMs models, we use the same features as those in table 1 (excluding the edge features).

5.2 Experiments

We evaluated the proposed method on the two data sets. We conducted the experiments in the following way. First, we converted each company annual report

into a sequence of text lines; for the SVMs base method, we use the vertex features and train two SVM models for each concept; for the linear-chain CRFs, we use the vertex features and the linear edge features to train the models; for TCRFs, we use the vertex features and the hierarchical edge features to train the models. For training SVM models we use SVM-light, which is available at http://svmlight.joachims.org/. For training linear-chain CRF models, we use KEG_CRFs, which is available at http://keg.cs.tsinghua.edu.cn/persons/tj/.

5.2.1 Experimental Results on the Synthetic Data Set

Table 2 shows the five-fold cross-validation results on the synthetic data set. SVM, CRF, and TCRF respectively represent the SVMs based method, the linear-chain CRFs method, and the proposed TCRFs method. Prec., Rec., and F1 respectively represent the scores of precision, recall, and F1-measure.

Table 2. Performance of semantic annotation on the synthetic data set (%)

Annotation Task	SVM			CRF			TCRF		
	Prec.	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**
Company_Secretary	99.26	88.74	93.71	100.0	100.0	100.0	100.0	100.0	100.0
Secretary_Email	50.00	7.52	13.07	50.00	42.86	46.15	100.0	100.0	100.0
Registered_Address	97.46	89.84	93.50	100.0	100.0	100.0	100.0	100.0	100.0
Company_Email	0.00	0.00	0.00	46.15	50.00	48.00	100.0	100.0	100.0
Average	61.68	46.53	50.07	89.15	89.15	89.15	100.0	100.0	100.0

We see that for both “Company_Secretary” and “Registered_Address”, all of the three methods can achieve high accuracy of annotation. Compared with the SVMs based method, CRF and TCRF can obtain better results. We can also see that for “Secretary_Email” and “Company_Email”, the proposed method TCRF significantly outperforms the SVMs based method and the linear-chain CRFs based method. We conducted sign tests on the results. The p values are much smaller than 0.01, indicating that the improvements are statistically significant.

5.2.2 Experimental Results on the Real Data Set

Table 3 shows the five-fold cross-validation results on the real data set. In the table, we also use SVM, CRF, and TCRF to respectively represent the SVMs based method, the linear-chain CRFs method, and the proposed TCRFs method; and use Prec., Rec., and F1 to respectively represent the scores of precision, recall, and F1-measure.

From the results we see that TCRF can achieve the best performance 89.87% in terms of F1-measure (outperforming CRF+7.67% and SVM+14.10% on average). In terms of both precision and recall, CRF can outperform SVM. TCRF again outperform CRF +3.14% in terms of precision and +12.08% in terms of recall. We conducted sign tests on the results. The p values are much smaller than 0.01, indicating that the improvements are statistically significant.

5.2.3 Discussions

(1) Effectiveness of TCRF. In the synthetic data set, the data are hierarchically organized. TCRF can indeed improve the annotation performance. On annotation of "Secretary_Email" and "Company_Email", the SVMs based method only uses the neighborhood contexts and thus cannot disambiguate them from each other (only 13.07% and 0.00% in terms of F1-measure). The linear-chain CRFs based method can improve the annotation result by making use of linear dependencies (46.15% and 48.00% respectively). However, as the linear-chain CRFs cannot model hierarchical dependencies, the improvements are limited. The proposed TCRFs based method can model the hierarchical dependencies, and obtain the best performance (100.00% and 100.00% respectively). This indicates that the proposed Tree-structured Conditional Random Fields are effective for the problem of hierarchical semantic annotation.

Table 3. Performance of semantic annotation on the real data set (%)

Annotation Task	SVM			CRF			TCRF		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
Company_Chinese_Name	88.82	89.40	89.11	82.10	80.69	81.37	84.34	92.72	88.33
Company_English_Name	90.51	95.33	92.86	71.68	80.14	75.66	89.26	88.67	88.96
Legal_Representative	94.84	97.35	96.08	92.86	96.60	94.66	94.84	97.35	96.08
Company_Secretary	99.29	93.33	96.22	91.65	96.99	94.23	77.96	96.67	86.31
Secretary_Email	57.14	8.89	15.39	69.94	56.53	62.34	73.86	97.01	83.87
Registered_Address	98.66	96.71	97.68	94.75	87.20	90.80	84.05	90.13	86.98
Office_Address	70.41	97.54	81.78	77.41	87.06	81.94	86.93	89.86	88.37
Company_Email	0.00	0.00	0.00	84.57	85.64	85.09	95.20	90.84	92.97
Newspaper	100.0	99.34	99.67	94.51	91.97	93.21	98.69	100.0	99.34
Accounting_Agency	83.15	95.63	88.95	73.81	56.77	62.73	79.57	97.19	87.50
Average	78.28	77.35	75.77	83.33	81.96	82.20	86.47	94.04	89.87

(2) Improvements over CRF. In the real data set, TCRF significantly outperforms the linear-chain CRF for the annotation of most concepts. For the concepts that have strong hierarchical dependencies, TCRF can achieve much better results than CRF, for example, on "Secretary_Email" and "Company_Email" TCRF outperforms CRF by +21.53% and +7.88%, respectively.

(3) Improvements over SVM. In the real data set, TCRF outperforms SVM +8.19% in terms of precision and +16.69% in terms of recall. The SVMs based method suffers from the extremely bad results on the annotation of "Secretary_Email" and "Company_Email". This is due to that the SVMs based method considers only neighborhood contexts. Besides the two concepts, TCRF also outperforms SVM on annotation of some other concepts, for example "Office_Address". We need notice that in some cases, TCRF underperforms SVM. For example on "Company_Chinese_Name" and "Company_English_Name", TCRF underperforms SVM by -0.78% and -2.9%,

respectively. This is because instances of such concepts seem to be independent and do not have dependencies with instances of the other concepts.

(4) Time complexity. We conducted analysis of time complexity of our approach. We tested the three methods on a computer with two 2.8G Dual-Core CPUs and three Gigabyte memory. In total, for training and annotating the fourteen concepts, the SVMs based method takes about 96 seconds and 30 seconds respectively, while the CRF method takes about 5 minutes 25 seconds and 5 seconds respectively. Our current implementation of the TCRF method used more time for training and annotation (about 50 minutes 40 seconds and 50 seconds respectively.) This indicates that the efficiency of TCRF still needs improvements.

(5) Error analysis. We conducted error analysis on the results of our approach.

There are mainly three types of errors. The first type of errors (about 34.85% of the errors) is that in some concepts, there are no hierarchical dependencies, for example "Company_Chinese_Name" and "Company_English_Name". In such cases, the proposed TCRFs contrarily result in worse performance than the SVMs based method that does not consider dependencies. About 28.05% of the errors occur when there are extra email addresses in the text. The third type of errors was due to extra line breaks in the text, which mistakenly breaks the targeted instance into multiple lines.

6 Conclusions

In this paper, we investigated the problem of hierarchical semantic annotation. We proposed a Tree-structured Conditional Random Field (TCRF) model. This model provides a novel way of incorporating the dependencies across the hierarchical structure to improve the performance of hierarchical semantic annotation. Using an approximate algorithm, i.e. Tree Reparameterization (TRP), efficient parameter estimation and annotation can be performed. Experimental results on two data sets show that the proposed model significantly outperforms the linear-chain CRF models and the SVMs based models for annotating hierarchically laid-out data. We also found that the efficiency of the proposed TCRF model still needs improvements.

References

[1] R. Benjamins and J. Contreras. Six challenges for the semantic web. Intelligent Software Components. Intelligent Software for the Networked Economy (isoco). 2002.

[2] A. L. Berger, S. A. Della Pietra, and V. J. Della Pietra. A maximum entropy approach to natural language processing. Computational Linguistics, Vol,22,1996. pp. 39-71

[3] R. C. Bunescu, R. J. Mooney. Collective information extraction with relational Markov networks. In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL'04), 2004. pp. 439-446

[4] F. Ciravegna. (LP)2, an adaptive algorithm for information extraction from web-related texts. In Proceedings of the IJCAI'2001 Workshop on Adaptive Text Extraction and Mining held in conjunction with 17th IJCAI'2001, Seattle, USA. 2001. pp. 1251-1256

[5] M. Collins. Discriminative training methods for hidden Markov models: Theory and Experiments with Perceptron Algorithms. In Proceedings of EMNLP'02. 2002.

[6] C. Cortes and V. Vapnik. Support-Vector Networks. Machine Learning, Vol. 20, 1995, pp. 273-297

[7] A. Finn and N. Kushmerick. Multi-level boundary classification for information extraction. In Proceedings of the ECML'2004, Pisa, 2004. pp.156-167
[8] Z. Ghahramani and M. I. Jordan. Factorial hidden Markov models. Machine Learning, Vol.29, 1997, pp. 245-273
[9] L. Gillick and S. Cox. Some statistical issues in the compairson of speech recognition algorithms. In International Conference on Acoustics Speech and Signal Processing, 1989, Vol. 1: 532-535
[10] J. Hammersley and P. Clifford. Markov fields on finite graphs and lattices. Unpublished manuscript. 1971.
[11] B. Hammond, A. Sheth, and K. Kochut. Semantic enhancement engine: a modular document enhancement platform for semantic applications over heterogeneous content, in real world semantic web applications. IOS Press, 2002. pp. 29-49
[12] S. Handschuh, S. Staab, and F. Ciravegna. S-CREAM - semi-automatic creation of metadata. In Proceedings of the 13th International Conference on Knowledge Engineering and Management (EKAW'2002), Siguenza, Spain, 2002. pp. 358-372
[13] N. Kushmerick, D. S. Weld, and R. B. Doorenbos. Wrapper induction for information extraction. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI). Nagoya, Japan, 1997. pp. 729-737
[14] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In Proceedings of the 18th International Conference on Machine Learning (ICML'01), 2001. pp. 282-289
[15] D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. Mathematical Programming, 1989. pp. 503-528
[16] T. Lou, R. Song, W.L. Li, and Z.Y. Luo. The design and implementation of a modern general purpose segmentation system, Journal of Chinese Information Processing, (5), 2001.
[17] A. McCallum, D. Freitag, and F. Pereira. Maximum entropy Markov models for information extraction and segmentation. In Proceedings of the 17th International Conference on Machine Learning (ICML'00), 2000. pp. 591-598
[18] B. Popov, A. Kiryakov, A. Kirilov, D. Manov, D. Ognyanoff, and M. Goranov. KIM - semantic annotation platform. In Proceedings of 2nd International Semantic Web Conference (ISWC'2003), Florida, USA, 2003. pp. 834-849
[19] L. Reeve. Integrating hidden Markov models into semantic web annotation platforms. Technique Report. 2004.
[20] F. Sha and F. Pereira. Shallow parsing with conditional random fields. In Proceedings of Human Language Technology, NAACL. 2003.
[21] C. Sutton, K. Rohanimanesh, and A. McCallum. Dynamic conditional random fields: factorized probabilistic models for labeling and segmenting sequence data. In Proceedings of ICML'2004. 2004.
[22] J. Tang, J. Li, H. Lu, B. Liang, and K. Wang. 2005a. iASA: learning to annotate the semantic web. Journal on Data Semantic, IV. Springer Press. pp. 110-145
[23] J. Tang, H. Li, Y. Cao, and Z. Tang. Email data cleaning. In Proceedings of SIGKDD'2005. August 21-24, 2005, Chicago, Illinois, USA. Full paper. pp. 489-499
[24] M. Wainwright, T. Jaakkola, and A. Willsky. Tree-based reparameterization for approximate estimation on graphs with cycles. In Proceedings of Advances in Neural Information Processing Systems (NIPS'2001), 2001. pp. 1001-1008
[25] J. Yedidia, W. Freeman, and Y. Weiss. Generalized belief propagation. Advances in Neural Information Processing Systems (NIPS). 2000.
[26] J. Zhu, Z. Nie, J. Wen, B. Zhang, and W. Ma. 2D conditional random fields for web information extraction. In Proceedings of ICML'2005.