

# Extracting Deeper Information from Richer Resource: EM Models for LTAG Treebank Induction

Libin Shen and Aravind K. Joshi

Department of Computer and Information Science

University of Pennsylvania

Philadelphia, PA 19104

{libin, joshi}@linc.cis.upenn.edu

## Abstract

In this paper, we propose novel EM algorithms for LTAG treebank induction, and present inside-outside algorithms on LTAG derivation shared forest. We illustrate our approach by showing how to use richer resources for this induction, in particular, the Penn Treebank, Propbank, and XTAG English Grammar.

## 1 Introduction

The framework of Lexicalized Tree Adjoining Grammar (LTAG) (Joshi and Schabes, 1997) provides representations for deeper linguistic information such as predicate-argument information, selectional information, and scope information (multi-component LTAG). In LTAG, each word in a sentence is associated with an *elementary tree*. Elementary trees are put together by *substitution* or *adjunction*. The tree corresponding to the composition of the elementary trees is called the *derivation* tree of the sentence, and the tree resulting from carrying out the composition is called the *derived* tree of the sentence.

The XTAG project (XTAG-Group, 2001) has implemented a handcrafted English grammar. The XTAG English Grammar contains about 1094 elementary trees, divided into 52 tree families and about 218 individual trees. Tree families represent subcategorization frames. The lexicon contains 30,000 uninflected forms of word, each of which is associated with a set of elementary trees or tree-families as well as the feature equations. Its morph database consists of approximately 317,000 inflected items. Sarkar (1996)

reported a rule based parser using the XTAG English Grammar. Unfortunately, we cannot use the XTAG grammar in a statistical parser due to the lack of a large LTAG treebank for the grammar.

In order to overcome this problem, Joshi and Srinivas (1994) first implemented a *supertag* (elementary tree or trees associated with a lexical item) corpus by extracting it from the Penn Treebank (PTB) (Marcus et al., 1994), using heuristic rules. Due to various limitations of this system, extracted supertags of the words in a sentence cannot always be successfully put together. Xia (2001) and Chen (2001) described deterministic systems that extract LTAG-style grammars from PTB. In their systems, the *head table* (Magerman, 1995) and the PTB functional tags were used to solve disambiguation in extraction. Chiang (2000) reported a similar method to extract an LTAG like treebank from PTB, and used it in a statistical parser. Shen et al. (2003) employed a similar technique to induce an LTAG treebank, to be used in a parse reranking system.

In these LTAG grammar and treebank induction systems, deterministic rules were used to solve the ambiguity in the elementary tree extraction process. However, it is clear that deterministic rules are not enough to solve ambiguity in extraction, especially in the case of the argument-adjunct distinction. In this paper, we will propose a statistical approach and treebank induction systems based on the Expectation-Maximization (EM) algorithm (Dempster et al., 1977). Since the EM algorithm is used to provide the maximum-likelihood estimate of the hidden data, the LTAG derivation trees in this application, it is natural for us to use this algorithm to induce the LTAG derivation trees from the PTB trees which is the

observed data in our EM algorithm. In (Chiang and Bikel, 2002), the EM algorithm was employed to obtain the preprocessing structures to be used in Treebank based parsing.

In the previous extraction work, only the Penn Treebank was used as input. There exist other linguistic resources which we believe are useful in LTAG grammar and treebank induction. In particular, we will focus on the Penn Proposition Bank (Propbank) (Kingsbury and Palmer, 2002), which provides a rich set of predicate-argument annotation on the PTB. In this paper, we will show how to use all these linguistic resources in our statistical LTAG extraction system.

## 2 A Statistical Approach

In this section, we will propose an Expectation-Maximization framework for LTAG treebank induction. The EM Algorithm is a general method of searching the maximum-likelihood estimate of the parameters of the hidden data from the observed data.

As far as LTAG grammar and treebank extraction is concerned, the observed data is a set of sentences with structure constraints associated to each sentence. For example, partially bracketed sentences as in (Pereira and Schabes, 1992) can be used as the observed data. In this paper we will mainly use fully bracketed PTB sentences as the observed data in EM. Furthermore, PTB trees with Propbank annotation will be employed as the observed data.

The LTAG *derivation tree* for each observed sentence is treated as the hidden data in the EM model. Each PTB tree is decomposed into a set lexicalized elementary trees. These elementary trees are combined together by substitution and adjunction, constituting the PTB tree or the LTAG derived tree. The LTAG derivation tree represents what the lexicalized elementary trees are and how they are put together.

### 2.1 Basic EM

We use  $\mathbf{x}, \mathbf{y}$  to represent a single sample, use  $\mathbf{X}, \mathbf{Y}$  to represent a set of samples, and use  $\mathcal{X}, \mathcal{Y}$  to represent a random variable for a single sample. Let  $\theta$  be the parameter variable, and  $\theta'$  the given parameter for the last step, which is constant.

In the **E** step, the EM algorithm first computes the expected value of complete data log-likelihood  $P(\mathcal{X}, \mathcal{Y} | \theta)$  with respect to the observed data and current parameter estimates  $\theta'$ , which is

$$Q(\theta, \theta') = E[\log P(\mathcal{X}, \mathcal{Y} | \theta) | \mathcal{X}, \theta']$$

In the **M** step, the EM algorithm searches for the next parameter estimates by maximizing the expectation.

$$\theta^* = \operatorname{argmax}_{\theta} Q(\theta, \theta')$$

### 2.2 EM for Derivation Tree Extraction

Throughout this paper,  $x$  stands for the observed data which is the PTB tree, and  $y$  stands for the hidden data which is the LTAG derivation tree.

Let the observed data be  $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ , and the hidden data be  $\mathbf{Y} = \{\mathbf{y}_1, \dots, \mathbf{y}_N\}$  respectively, where  $N$  is the size of the data set. Thus the  $Q$  function in EM is defined as

$$Q(\theta, \theta') \equiv \sum_{\mathbf{y} \in \mathbf{Y}} (\log P(\mathcal{X}, \mathbf{y} | \theta)) P(\mathbf{y} | \mathcal{X}, \theta') \quad (1)$$

Since

$$P(\mathcal{X}, \mathbf{y} | \theta') = P(\mathbf{y} | \mathcal{X}, \theta') P(\mathcal{X} | \theta')$$

and  $P(\mathcal{X} | \theta')$  is irrelevant to  $\theta$  in maximization, we can redefine  $Q(\theta, \theta')$  as follows for the sake of simplicity.

$$Q(\theta, \theta') = \sum_{\mathbf{y} \in \mathbf{Y}} (\log P(\mathcal{X}, \mathbf{y} | \theta)) P(\mathcal{X}, \mathbf{y} | \theta') \quad (2)$$

$$= \sum_{i=1}^N \sum_{j=1}^{M_i} (\log P(\mathbf{x}_i, \mathbf{y}_{ij} | \theta)) P(\mathbf{x}_i, \mathbf{y}_{ij} | \theta'), \quad (3)$$

where  $M_i$  is the number of possible distinct hidden structures for the observed sample  $\mathbf{x}_i$ .

### 2.3 Using Linguistic Resources

A fundamental problem of using EM in natural language is that the EM algorithm does not guarantee to converge to a global maximum of the likelihood, and the local maximum that EM finds depends on the initial parameter setting. For example, Pereira and Schabes (1992) showed that

the context free grammar learned from an EM algorithm is usually not linguistically motivated.

One of the solutions to this problem is to apply linguistic knowledge to treebank induction. By using linguistic knowledge derived from richer resources such as the XTAG English Grammar and the Propbank, in our case, we will be able to select the initial parameters close to a convergence point at which the resulting LTAG grammar and treebank are linguistically sound, and to limit the search space.

As far as our EM algorithm is concerned, linguistic knowledge is employed to decrease the space of hidden structures. In this way, we can greatly decrease the number of the possible hidden structures so as to limit the search space as well as the initial setting.

Since we use Penn Treebank as the input of the induction, we only need to consider the derivation trees whose corresponding derived tree is compatible with the PTB tree. Our method of extracting derivation trees from PTB trees is similar to the algorithms used in (Xia, 2001) and (Chen, 2001). The difference is that here we maintain the ambiguity in extraction, i.e. argument-adjunction distinction, instead of use deterministic rules based on PTB annotation, which is not enough to solve the ambiguity.

We use the algorithm reported in (Shen, 2004) to generate a *shared forest* to represent all the possible derivation trees for a given PTB tree. In that paper, PTB tags, Propbanks tags and the XTAG formalism were used to constrain the space of all possible derivation trees. For example, the predicate-argument tags in Propbank are used to recognize *predicate auxiliary* structures, such as “John *is supposed* to take 5 courses this semester”, where the predicate structure *is supposed* modifies *take*, and *John* is an argument of *take*. However, the use of Propbank tags cannot solve all the ambiguities. In the Propbank, arguments and adjuncts are not distinguished. For example, an PropBank argument with thematic role *location* can be either an argument or an adjunct in LTAG, depending on the individual lexical item. In this case, we maintain both choices in our shared forest. Anyway, the Propbank tags help us to find all the arguments and adjuncts for each predicate.

### 3 Model 1

Based on how we compute  $P(\mathbf{x}_i, \mathbf{y}_{ij} | \theta)$ , we will propose two models for LTAG treebank induction. Model 1 is an unlexicalized model and model 2, which we will describe in the next section, is a lexicalized model.

In model 1, each hidden structure  $\mathbf{y}_{ij}$  can be decomposed as a set of 3-tuples.

$$\begin{aligned} \mathbf{y}_{ij} &= \{n_{ijk} | k = 1, \dots, S_{ij}\}, \text{ and} \\ n_{ijk} &= (t_{ijk}, p_{ijk}, c_{ijk}), \end{aligned}$$

where  $S_{ij}$  is the number of nodes in the derivation tree represented by  $\mathbf{y}_{ij}$ .

Each node  $n$  in the derivation tree is represented with a 3-tuple  $(t, p, c)$ , where

- $t$  is the elementary tree of  $n$  extracted from PTB,
- $p$  is the parent elementary tree of  $n$ ,
- $c$  is the location on  $p$  and the type that  $t$  is attached,

Then we compute  $P(\mathbf{x}_i, \mathbf{y}_{ij} | \theta)$  as

$$P(\mathbf{x}_i, \mathbf{y}_{ij} | \theta) = \prod_{k=1}^{S_{ij}} P_{\theta}(t_{ijk} | p_{ijk}, c_{ijk}) \quad (4)$$

For the sake of simplicity, we define

$$\alpha_{\langle t, p, c \rangle}^{\theta} = P_{\theta}(t | p, c)$$

where

$$\sum_t \alpha_{\langle t, p, c \rangle}^{\theta} = 1 \quad \forall p, c \quad (5)$$

We have

$$P(\mathbf{x}_i, \mathbf{y}_{ij} | \theta) = \prod_{k=1}^{S_{ij}} \alpha_{\langle t_{ijk}, p_{ijk}, c_{ijk} \rangle}^{\theta} \quad (6)$$

Now we combine (6) with (3). Since  $P(\mathbf{x}_i, \mathbf{y}_{ij} | \theta')$  is constant which can be computed with (6), we use  $f_{ij}^{\theta'}$  to represent it. Thus

$$\begin{aligned} &Q(\theta, \theta') \\ &= \sum_{i=1}^N \sum_{j=1}^{M_i} f_{ij}^{\theta'} \log P(\mathbf{x}_i, \mathbf{y}_{ij} | \theta) \end{aligned} \quad (7)$$

$$= \sum_{i=1}^N \sum_{j=1}^{M_i} f_{ij}^{\theta'} \sum_{k=1}^{S_{ij}} \log \alpha_{\langle t_{ijk}, p_{ijk}, c_{ijk} \rangle}^{\theta} \quad (8)$$

Then we search for new  $\theta$  by solving

$$\operatorname{argmax}_{\alpha^\theta} Q(\theta, \theta'),$$

with constraints (5). We solve  $\alpha^\theta$  as follows. For each pair of  $(p, c)$  in (5), we add a Lagrange multiplier  $\lambda_{p,c}$ . According to (8), for any fixed  $t, p, c$

$$\frac{\partial}{\partial \alpha_{<t,p,c>}^\theta} \left[ \sum_{(i,j,k) \in A_{<t,p,c>}} f_{ij}^{\theta'} \log \alpha_{<t,p,c>}^\theta + \sum_{p,c} \lambda_{p,c} \left( \sum_{t_x} \alpha_{<t_x,p,c>}^\theta - 1 \right) \right] = 0, \quad (9)$$

where

$$A_{<t,p,c>} = \{(i, j, k) \mid t_{ijk} = t, p_{ijk} = p, c_{ijk} = c\}.$$

Having  $p, c$  fixed, summing over  $t$ , solving for  $\lambda_{p,c}$ , we get

$$\lambda_{p,c} = - \sum_{t_x} \sum_{(i,j,k) \in A_{<t_x,p,c>}} f_{ij}^{\theta'} \quad (10)$$

Solving for  $\alpha_{<t,p,c>}^\theta$  with (9) and (10), we get

$$\alpha_{<t,p,c>}^\theta = \frac{\sum_{(i,j,k) \in A_{<t,p,c>}} f_{ij}^{\theta'}}{\sum_{t_x} \sum_{(i,j,k) \in A_{<t_x,p,c>}} f_{ij}^{\theta'}} \quad (11)$$

(11) is the update equation of the EM algorithm for LTAG derivation tree extraction. Although the inference is complicated, the rule of parameter updating is very simple. The new parameters are achieved by maximum-likelihood estimation with respect to the distribution given in the last step.

### 3.1 Inside-Outside Algorithm

In (11), we sum up over all the possible deep structures in set  $A$ . A naive implementation is to enumerate all the elements in  $A$ , which is of exponential complexity. The method to avoid this is to use a dynamic programming algorithm similar to the Inside-Outside algorithm in (Pereira and Schabes, 1992). However, the situation here is more complicated.

In (Pereira and Schabes, 1992), only CFGs of Chomsky normal-form (CNF) is under consideration. The bottom-up inside computation and the top-down outside computation can be defined recursively with respect to the word indices neatly. For us, the search space is the LTAG derivation

trees that generate the LTAG derived tree in PTB style, as described in the section 2.3. Here we will introduce inside-outside algorithms over the shared forest of derivation trees.

The shared forest for a given PTB tree is a directed acyclic graph (DAG). Each node  $n_i$  represent an elementary tree anchored on word  $w_i$ . Each node  $n_i$  has  $p_i$  slots, numbered from 1 to  $p_i$ . Each slot  $c_{ij}$  is associated with  $q_{ij}$  children nodes, which are  $h_{ij1}, h_{ij2}, \dots, h_{ijk}, \dots$ , where  $1 \leq j \leq p_i$  and  $1 \leq k \leq q_{ij}$ . Each slot  $c_{ij}$  represents a substitution or adjunction at some position of  $n_i$ . Furthermore, each node  $n_i$  is also associated with a set of parent nodes  $g_{i1}, g_{i2}, \dots, g_{im}, \dots$ . Obviously  $n_i$  is one of the parent nodes for any child node  $h_{ijk}$ .

The the inside-outside probabilities are defined as follows. First, the inside probability

$$I(n_i) = \prod_j \sum_k I(h_{ijk}) P(h_{ijk} | n_i, c_{ij}) \quad (12)$$

if  $n_i$  is not a leaf node. Otherwise  $I(n_i) = 1$ .

Let  $Id : bsa.tex, v1.72004/06/2317 : 31 : 44libinExp$  be an index function, such that  $Id(n_i) = i$ . Then, the outside probability

$$O(n_i) = \sum_{l: l=Id(g_{im})} O(n_l) P(n_i | n_l, c_{l,x}) \times \prod_{x \neq j} \sum_k I(h_{ljk}) P(h_{ljk} | n_l, c_{lj}),$$

where  $x$  means that  $n_i$  is in the  $x^{th}$  slot of  $n_l$ 's, if  $n_i$  is not the *root* node<sup>1</sup>. Otherwise  $O(n_i) = 1$ . Then, for any elementary tree  $p$  and  $q$

$$\hat{P}(p|q, c) = \frac{\sum_S M(p|q, c)}{\sum_S \sum_x M(x|q, c)} \quad (13)$$

where

$$M(u|v, c) = \sum_{n_i=v, c_{ij}=c, h_{ijk}=u} \frac{I(h_{ijk}) O(n_i) P(h_{ijk} | n_i, c_{ij})}{I(root)} \times \prod_{y \neq j} \sum_z I(h_{iyz}) P(h_{iyz} | n_i, c_{iy}), \quad (14)$$

and  $S$  is the set of all derivation forests for the trees in the PTB.

<sup>1</sup>We put a unique artificial root node over all the roots in the shared forest that represent all the derivation trees for a given PTB sentence.

In order to compute inside-outside probability and reestimate distribution recursive, we first run a topological sorting over the DAG. The result of the topological sorting is an array of nodes that respect dependence relation between the nodes. Thus we can read nodes from left to right or from right to left to implement bottom-up or top-down visiting respectively.

#### 4 Model 2

In model 2, each hidden structure  $\mathbf{y}_{ij}$  can be decomposed as a set of 4-tuples.

$$\begin{aligned} \mathbf{y}_{ij} &= \{n_{ijk} | k = 1, \dots, S_{ij}\}, \text{ and} \\ n_{ijk} &= (t_{ijk}, p_{ijk}, c_{ijk}, w_{ijk}), \end{aligned}$$

where  $S_{ij}$  is the number of nodes in the derivation tree represented by  $\mathbf{y}_{ij}$ . Each node  $n$  in the derivation tree is represented with a 4-tuple  $(t, p, c, w)$ , where

- $t$  is the elementary tree of  $n$  extracted from the PTB,
- $p$  is the parent elementary tree of  $n$ ,
- $c$  is the location on  $p$  and the type that  $t$  is attached,
- $w$  is the lexical item or surface word of  $t$

Then we compute  $P(\mathbf{x}_{ij}, \mathbf{y}_{ij} | \theta)$  as

$$P(\mathbf{x}_{ij}, \mathbf{y}_{ij} | \theta) = \prod_{k=1}^{S_{ij}} P_{\theta}(t_{ijk} | p_{ijk}, c_{ijk}) P_{\theta}(w_{ijk} | t_{ijk})$$

For the sake of simplicity, we define

$$\begin{aligned} \alpha_{\langle t, p, c \rangle}^{\theta} &= P_{\theta}(t | p, c) \\ \beta_{\langle t, w \rangle}^{\theta} &= P_{\theta}(w | t) \end{aligned}$$

where

$$\sum_t \alpha_{\langle t, p, c \rangle}^{\theta} = 1 \quad \forall p, c \quad (15)$$

$$\sum_w \beta_{\langle t, w \rangle}^{\theta} = 1 \quad \forall t \quad (16)$$

Thus we decompose  $\theta$  into  $(\alpha^{\theta}, \beta^{\theta})$ .

We have

$$\begin{aligned} P(\mathbf{x}_{ij}, \mathbf{y}_{ij} | \theta) &= \prod_{k=1}^{S_{ij}} \alpha_{\langle t_{ijk}, p_{ijk}, c_{ijk} \rangle}^{\theta} \\ &\quad \times \beta_{\langle t_{ijk}, w_{ijk} \rangle}^{\theta} \end{aligned} \quad (17)$$

Similarly, solving  $\beta^{\theta}$  and  $\gamma^{\theta}$ , we get

$$\alpha_{\langle t, p, c \rangle}^{\theta} = \frac{\sum_{(i, j, k) \in A_{\langle t, p, c \rangle}} f_{ij}^{\theta'}}{\sum_{t_x} \sum_{(i, j, k) \in A_{\langle t_x, p, c \rangle}} f_{ij}^{\theta'}} \quad (18)$$

$$\beta_{\langle t, w \rangle}^{\theta} = \frac{\sum_{(i, j, k) \in B_{\langle t, w \rangle}} f_{ij}^{\theta'}}{\sum_{w_x} \sum_{(i, j, k) \in B_{\langle t, w_x \rangle}} f_{ij}^{\theta'}} \quad (19)$$

where

$$A_{\langle t, p, c \rangle} = \{(i, j, k) | t_{ijk} = t, p_{ijk} = p, c_{ijk} = c\},$$

$$B_{\langle t, w \rangle} = \{(i, j, k) | t_{ijk} = t, w_{ijk} = w\}.$$

#### 4.1 Inside-Outside Algorithm

The inside-outside algorithm for model 2 is similar to that for model 1. There is a slight difference. Now we need to take  $P(w|t)$  into account. For example, now the inside probability

$$I(n_i) = P(w_i | n_i) \prod_j \sum_k I(h_{ijk}) P(h_{ijk} | n_i, c_{ij})$$

if  $n_i$  is not a leaf node. Otherwise  $I(n_i) = P(w_i | n_i)$ .

### 5 Experiments and Analysis

In this section, we report the experiments on the Penn Treebank.

Figure 1 and 2 show statistics of the grammars extracted from all the WSJ sections of PTB with model 1 and model 2. The X axis represents the rounds of EM algorithm. The Y axis in Figure 1 represents the entropy estimate

$$H(G) = - \sum_{root \in S} \log I(root) \quad (20)$$

that sums over all the sentences in  $S$ . The Y axis in Figure 2 stands for the number of elementary trees obtained by selecting the derivation tree with the highest probability for each PTB tree.

The uniform distribution is used in the initial setting. For both models, the entropy drops significantly in the first round of EM, but does not change much afterwards.

The number of all the possible elementary trees is 11918. With model 1, the number is decreased to 10062, and with model 2, the number is decreased to 10156. Model 2 obtains more elementary trees than model 1. The reason is that the lexicalized model prefers elementary trees of

low frequency due to the introduction of the emit probability on lexical items.

Figure 3 show the convergence curves. The X axis represents the number of sections used in grammar extraction. The Y axis stands for the number of elementary trees obtained by selecting the derivation tree with the highest probability for each PTB tree. The performance of model 1 and model 2 roughly the same. The number of elementary tree does not converge with respect to the size of the data set, which was the same as the previous works in (Xia, 2001; Chen, 2001).

## 6 Future Work

As for the future work, we will use more linguistic resource to refine the result, especially the handcrafted XTAG English Grammar and Penn Propbank. In our current work, they are only used in generating derivation shared forest to limit the search space. We plan to incorporate them into the EM models.

### 6.1 Model 3

We plan to use the handcrafted XTAG English Grammar in the EM model as follows. Each PTB extracted elementary tree will be mapped to an elementary tree defined in the XTAG English Grammar. For each word, only elementary trees in the lexicon entry of this word will be considered as the possible hidden structures. On the other hand, PTB extracted elementary trees will become linguistically meaningful because of the mapping.

We have shown that model 2, the lexicalized model, prefers elementary trees with low frequency, thus having introduced many incorrect elementary trees. It is clear that this problem will be solved with model 3.

We represent the deep structure  $\mathbf{y}_{ij}$  with a 5-tuple  $(t, p, c, w, v)$  as follows.

- $t$  is the elementary tree of  $n$ ,
- $p$  is the parent elementary tree of  $n$ ,
- $c$  is the location on  $p$  and the type of attachment that  $t$  is attached,
- $w$  is the lexical item or surface word of  $t$

- $v$  is a variation of  $t$ , which is identical to a segment of the PTB tree  $\mathbf{x}_{ij}$

Then we compute  $P(\mathbf{x}_{ij}, \mathbf{y}_{ij}|\theta)$  as

$$P(\mathbf{x}_{ij}, \mathbf{y}_{ij}|\theta) = \prod_{k=1}^{S_{ij}} P_{\theta}(t_{ijk}|p_{ijk}, c_{ijk}) P_{\theta}(w_{ijk}|t_{ijk}) P_{\theta}(v_{ijk}|t_{ijk})$$

Thus, we can not only solve the problem in model 2, but also build the connection between the extracted grammar and the handcrafted grammar.

### 6.2 Model 4

In model 1 and 2, we actually cannot distinguish some elementary trees. For example, the tree for *broke* in *John broke the window* and tree for *found* in *Mary found a book* are the same. However, if we combine the Propbank tags with LTAG elementary trees, these problems can be solved easily. Up to now, 90% of predicates in the PTB has been tagged. We hope to be able to use the these Propbank tags for all the predicates in the near future.

According to the mapping between the PTB extracted elementary trees and elementary trees in XTAG English Grammar, we can transport the Propbank annotation from PTB to XTAG. In this way, we automatically assign thematic role and sense tagging on the handcrafted XTAG grammar.

## 7 Conclusions

In this paper, we have proposed a method for LTAG treebank induction for extracting deeper information from richer resources and we have given inside-outside algorithms over LTAG derivation shared forests. In our approach, we use various linguistic resources in LTAG treebank induction, which includes the Penn Treebank, Propbank and XTAG English Grammar. The experimental results with Model 1 and 2 are promising. We believe the expected LTAG treebank will be useful in various NLP tasks, such as semantic parsing, information extraction and machine translation.

## References

- J. Chen. 2001. *Towards Efficient Statistical Parsing using Lexicalized Grammatical Information*. Ph.D. thesis, University of Delaware.
- D. Chiang and D. Bikel. 2002. Recovering latent information in treebanks. In *Proceedings of COLING 2002*.
- D. Chiang. 2000. Statistical Parsing with an Automatically-Extracted Tree Adjoining Grammar. In *Proceedings of ACL-2000*.
- A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society Series B*, 39(1):1–38.
- A. Joshi and Y. Schabes. 1997. Tree-adjoining grammars. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 69 – 124. Springer.
- A. Joshi and B. Srinivas. 1994. Disambiguation of super parts of speech (or supertags): Almost parsing. In *COLING'94*.
- P. Kingsbury and M. Palmer. 2002. From treebank to propbank. In *Proceedings of the 3rd LREC*.
- D. Magerman. 1995. Statistical decision-tree models for parsing. In *Proceedings of the 33rd ACL*.
- M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1994. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2):313–330.
- F. Pereira and Y. Schabes. 1992. Inside-outside reestimation from partially bracketed corpora. In *Proceedings of the 30th ACL*.
- A. Sarkar. 1996. Incremental parser generation for tree adjoining grammars. In *Proceedings of the 34th Meeting of the ACL, Student Session*.
- L. Shen, A. Sarkar, and A. K. Joshi. 2003. Using Itag based features in parse reranking. In *Proceedings of EMNLP 2003*.
- L. Shen. 2004. Nondeterministic Itag derivation tree extraction. to appear.
- F. Xia. 2001. *Automatic Grammar Generation From Two Different Perspectives*. Ph.D. thesis, University of Pennsylvania.
- XTAG-Group. 2001. A lexicalized tree adjoining grammar for english. Technical Report 01-03, IRCS, Univ. of Pennsylvania.

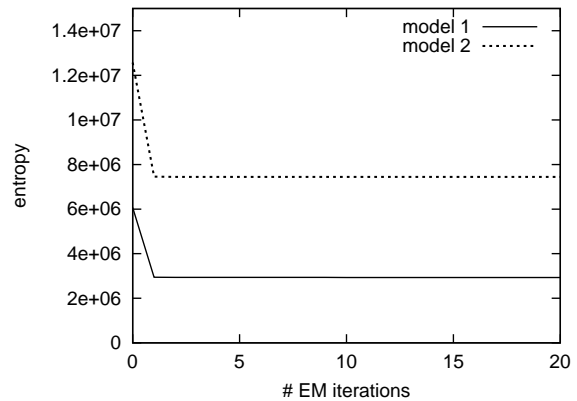


Figure 1: Entropy on all PTB WSJ sections

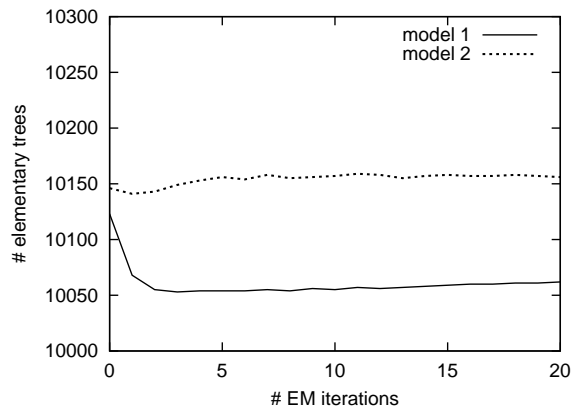


Figure 2: Number of elementary trees on all PTB WSJ sections.

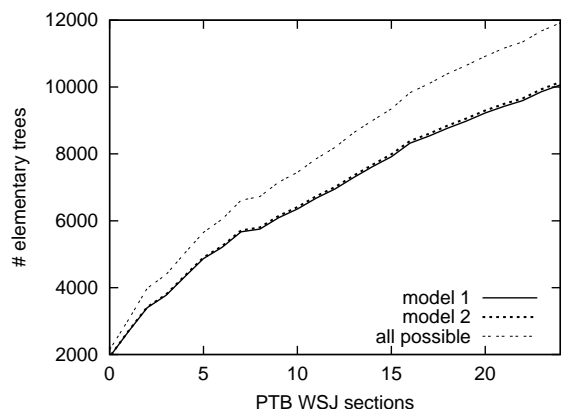


Figure 3: Convergence on the number of elementary trees.