Applying Co-Training methods to Statistical Parsing*

Anoop Sarkar
Dept. of Computer and Information Science
University of Pennsylvania
200 South 33rd Street,
Philadelphia, PA 19104-6389 USA
anoop@linc.cis.upenn.edu

Abstract

We propose a novel Co-Training method for statistical parsing. The algorithm takes as input a small corpus (9695 sentences) annotated with parse trees, a dictionary of possible lexicalized structures for each word in the training set and a large pool of unlabeled text. The algorithm iteratively labels the entire data set with parse trees. Using empirical results based on parsing the Wall Street Journal corpus we show that training a statistical parser on the combined labeled and unlabeled data strongly outperforms training only on the labeled data.

1 Introduction

The current crop of statistical parsers share a similar training methodology. They train from the Penn Treebank (Marcus et al., 1993); a collection of 40,000 sentences that are labeled with corrected parse trees (approximately a million word tokens). In this paper, we explore methods for statistical parsing that can be used to combine small amounts of labeled data with unlimited amounts of unlabeled data. In the experiment reported here, we use 9695 sentences of bracketed data (234467 word tokens). Such methods are attractive for the following reasons:

- Bracketing sentences is an expensive process. A parser that can be trained on a small amount of labeled data will reduce this annotation cost.
- Creating statistical parsers for novel domains and new languages will become easier.
- Combining labeled data with unlabeled data allows exploration of unsupervised methods which can now be tested using evaluations compatible with supervised statistical parsing.

In this paper we introduce a new approach that combines unlabeled data with a small amount of labeled (bracketed) data to train a statistical parser. We use a Co-Training method (Yarowsky, 1995; Blum and Mitchell, 1998; Goldman and Zhou, 2000) that has been used previously to train classifiers in applications like word-sense disambiguation (Yarowsky, 1995), document classification (Blum and Mitchell, 1998) and named-entity recognition (Collins and Singer, 1999) and apply this method to the more complex domain of statistical parsing.

* I would like to thank Aravind Joshi, Mitch Marcus, Mark Liberman, B. Srinivas, David Chiang and the anonymous reviewers for helpful comments on this work. This work was partially supported by NSF Grant SBR8920230, ARO Grant DAAH0404-94-G-0426, and DARPA Grant N66001-00-1-8915.

2 Unsupervised techniques in language processing

While machine learning techniques that exploit annotated data have been very successful in attacking problems in NLP, there are still some aspects which are considered to be open issues:

- Adapting to new domains: training on one domain, testing (using) on another.
- Higher performance when using limited amounts of annotated data.
- Separating structural (robust) aspects of the problem from lexical (sparse) ones to improve performance on unseen data.

In the particular domain of statistical parsing there has been limited success in moving towards unsupervised machine learning techniques (see Section 7 for more discussion). A more promising approach is that of combining small amounts of seed labeled data with unlimited amounts of unlabeled data to bootstrap statistical parsers. In this paper, we use one such machine learning technique: Co-Training, which has been used successfully in several classification tasks like web page classification, word sense disambiguation and named-entity recognition.

Early work in combining labeled and unlabeled data for NLP tasks was done in the area of unsupervised part of speech (POS) tagging. (Cutting et al., 1992) reported very high results (96% on the Brown corpus) for unsupervised POS tagging using Hidden Markov Models (HMMs) by exploiting hand-built tag dictionaries and equivalence classes. Tag dictionaries are predefined assignments of all possible POS tags to words in the test data. This impressive result triggered several follow-up studies in which the effect of hand tuning the tag dictionary was quantified as a combination of labeled and unla-

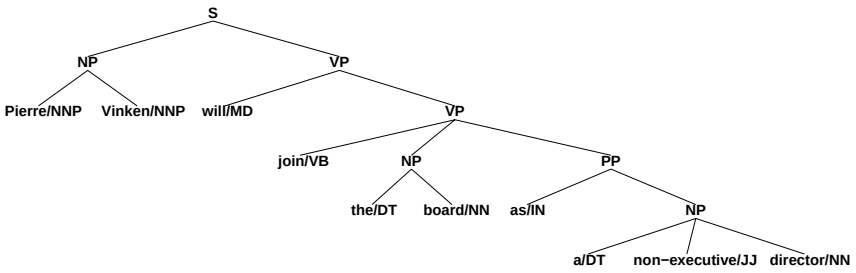

Figure 1: An example of the kind of output expected from a statistical parser.

beled data. The experiments in (Merialdo, 1994; Elworthy, 1994) showed that only in very specific cases HMMs were effective in combining labeled and unlabeled data.

However, (Brill, 1997) showed that aggressively using tag dictionaries extracted from labeled data could be used to bootstrap an unsupervised POS tagger with high accuracy (approx 95% on WSJ data). We exploit this approach of using tag dictionaries in our method as well (see Section 3.2 for more details). It is important to point out that, before attacking the problem of parsing using similar machine learning techniques, we face a representational problem which makes it difficult to define the notion of tag dictionary for a statistical parser.

The problem we face in parsing is more complex than assigning a small fixed set of labels to examples. If the parser is to be generally applicable, it has to produce a fairly complex "label" given an input sentence. For example, given the sentence *Pierre Vinken will join the board as a non-executive director*, the parser is expected to produce an output as shown in Figure 1.

Since the entire parse cannot be reasonably considered as a monolithic label, the usual method in parsing is to decompose the structure assigned in the following way:

S(join) → NP(Vinken) VP(join)

NP(Vinken) → Pierre Vinken

VP(join) → will VP(join)

VP(join) → join NP(board) PP(as)

. . .

However, such a recursive decomposition of structure does not allow a simple notion of a tag dictionary. We solve this problem by decomposing the structure in an approach that is different from that shown above which uses context-free rules.

The approach uses the notion of tree rewriting as defined in the Lexicalized Tree Adjoining Grammar (LTAG) formalism (Joshi and Schabes, 1992)[1] which retains the notion of lexicalization that is crucial in the success of a statistical parser while permitting a simple definition of tag dictionary. For example, the parse in Figure 1 can be generated by assigning the structured labels shown in Figure 2 to each word in the sentence (for simplicity, we assume that the noun phrases are generated here as a single word). We use a tool described in (Xia et al., 2000) to convert the Penn Treebank into this representation.

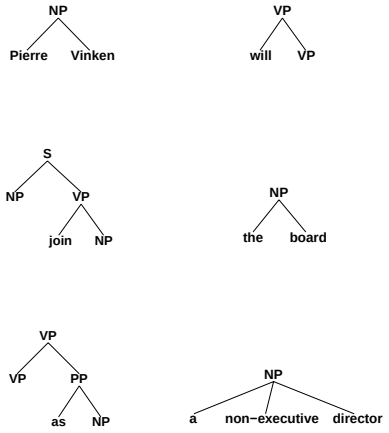

Figure 2: Parsing as tree classification and attachment.

Combining the trees together by rewriting nodes as trees (explained in Section 2.1) gives us the parse tree in Figure 1. A history of the bi-lexical dependencies that define the probability model used to construct the parse is shown in Figure 3. This history is called the *derivation tree*.

In addition, as a byproduct of this kind of representation we obtain more than the phrase structure of each sentence. We also produce a more embellished parse in which phenomena such as predicate-argument structure, subcategorization and movement are given a probabilis-

[1]This is a lexicalized version of Tree Adjoining Grammar (Joshi et al., 1975; Joshi, 1985).

tic treatment.

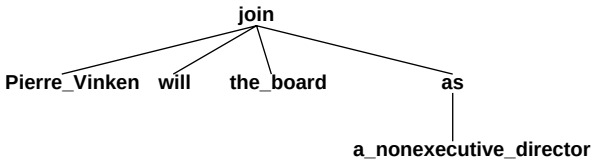

Figure 3: A derivation indicating all the attachments between trees that have occurred during the parse of the sentence.

2.1 The Generative Model

A stochastic LTAG derivation proceeds as follows (Schabes, 1992; Resnik, 1992). An initial tree is selected with probability P_{init} and other trees selected by words in the sentence are combined using the operations of substitution and adjoining. These operations are explained below with examples. Each of these operations is performed with probability P_{attach}.

For each τ that can be valid start of a derivation:

$$\sum_{\tau} P_{init}(\tau) = 1$$

Substitution is defined as rewriting a node in the frontier of a tree with probability P_{attach} which is said to be proper if:

$$\sum_{\tau'} P_{attach}(\tau, \eta \rightarrow \tau') = 1$$

where $\tau, \eta \rightarrow \tau'$ indicates that tree τ' is substituting into node η in tree τ. An example of the operation of substitution is shown in Figure 4.

Adjoining is defined as rewriting any internal node of a tree by another tree. This is a recursive rule and each adjoining operation is performed with probability P_{attach} which is proper if:

$$P_{attach}(\tau, \eta \rightarrow \text{NA}) + \sum_{\tau'} P_{attach}(\tau, \eta \rightarrow \tau') = 1$$

P_{attach} here is the probability that τ' rewrites an internal node η in tree τ or that no adjoining (NA) occurs at node η in τ. The additional factor that accounts for no adjoining at a node is required for the probability to be well-formed. An example of the operation of adjoining is shown in Figure 5.

Each LTAG derivation $\mathcal{D}$ which was built starting from tree α with n subsequent attachments has the probability:

$$Pr(\mathcal{D}) = P_{init}(\alpha) \prod_{1 \leq i \leq n} P_{attach}(\tau, \eta \rightarrow \tau_i')$$

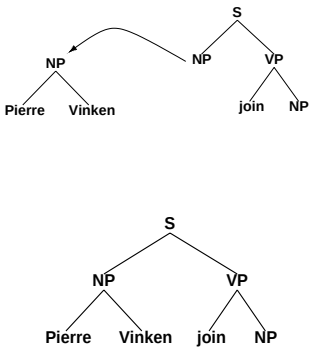

Figure 4: Example substitution of the tree for *Pierre Vinken* into the tree for *join*: $\tau(\text{join}), \text{NP} \rightarrow \tau'(\text{Pierre Vinken})$.

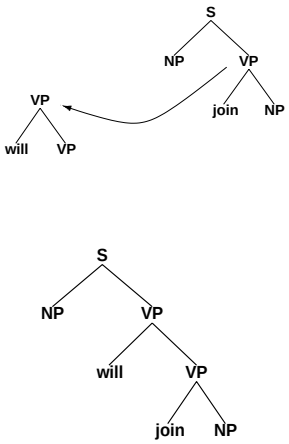

Figure 5: Example adjoining of the tree for *will* into the tree for *join*: $\tau(\text{join}), \text{VP} \rightarrow \tau'(\text{will})$.

Note that assuming each tree is lexicalized by one word the derivation $\mathcal{D}$ corresponds to a sentence of $n+1$ words.

In the next section we show how to exploit this notion of tag dictionary to the problem of statistical parsing.

3 Co-Training methods for parsing

Many supervised methods of learning from a Treebank have been studied. The question we want to pursue in this paper is whether unlabeled data can be used to improve the performance of a statistical parser and at the same time reduce the amount of labeled training data necessary for good performance. We will assume the

data that is input to our method will have the following characteristics:

1. A small set of sentences labeled with corrected parse trees and large set of unlabeled data.
2. A pair of probabilistic models that form parts of a statistical parser. This pair of models must be able to mutually constrain each other.
3. A tag dictionary (used within a backoff smoothing strategy) for labels are not covered in the labeled set.

The pair of probabilistic models can be exploited to bootstrap new information from unlabeled data. Since both of these steps ultimately have to agree with each other, we can utilize an iterative method called Co-Training that attempts to increase agreement between a pair of statistical models by exploiting mutual constraints between their output.

Co-Training has been used before in applications like word-sense disambiguation (Yarowsky, 1995), web-page classification (Blum and Mitchell, 1998) and named-entity identification (Collins and Singer, 1999). In all of these cases, using unlabeled data has resulted in performance that rivals training solely from labeled data. However, these previous approaches were on tasks that involved identifying the right label from a small set of labels (typically 2–3), and in a relatively small parameter space. Compared to these earlier models, a statistical parser has a very large parameter space and the labels that are expected as output are parse trees which have to be built up recursively. We discuss previous work in combining labeled and unlabeled data in more detail in Section 7.

Co-training (Blum and Mitchell, 1998; Yarowsky, 1995) can be informally described in the following manner:

- Pick two (or more) "views" of a classification problem.
- Build separate models for each of these "views" and train each model on a small set of labeled data.
- Sample an unlabeled data set and to find examples that each model independently labels with high confidence. (Nigam and Ghani, 2000)
- Confidently labeled examples can be picked in various ways. (Collins and Singer, 1999; Goldman and Zhou, 2000)
- Take these examples as being valuable as training examples and iterate this procedure until the unlabeled data is exhausted.

Effectively, by picking confidently labeled data from each model to add to the training data, one model is labeling data for the other model.

3.1 Lexicalized Grammars and Mutual Constraints

In the representation we use, parsing using a lexicalized grammar is done in two steps:

1. Assigning a set of lexicalized structures to each word in the input sentence (as shown in Figure 2).
2. Finding the correct attachments between these structures to get the best parse (as shown in Figure 1).

Each of these two steps involves ambiguity which can be resolved using a statistical model. By explicitly representing these two steps independently, we can pursue independent statistical models for each step:

1. Each word in the sentence can take many different lexicalized structures. We can introduce a statistical model that disambiguates the lexicalized structure assigned to a word depending on the local context.
2. After each word is assigned a certain set of lexicalized structures, finding the right parse tree involves computing the correct attachments between these lexicalized structures. Disambiguating attachments correctly using an appropriate statistical model is essential to finding the right parse tree.

These two models have to agree with each other on the trees assigned to each word in the sentence. Not only do the right trees have to be assigned as predicted by the first model, but they also have to fit together to cover the entire sentence as predicted by the second model[2]. This represents the mutual constraint that each model places on the other.

3.2 Tag Dictionaries

For the words that appear in the (unlabeled) training data, we collect a list of part-of-speech labels and trees that each word is known to select in the training data. This information is stored in a POS tag dictionary and a tree dictionary. It is important to note that no frequency or any other distributional information is stored. The only information stored in the dictionary is which tags or trees can be selected by each word in the training data.

We use a count cutoff for trees in the labeled data and combine observed counts into an *unobserved* tree count. This is similar to the usual technique of assigning the token *unknown* to infrequent word tokens. In this way, trees unseen in the labeled data but in the tag dictionary are assigned a probability in the parser.

The problem of lexical coverage is a severe one for unsupervised approaches. The use of tag dictionaries is a way around this problem. Such an approach has already been used for unsupervised part-of-speech tagging in (Brill, 1997) where seed data of which POS tags can be selected by each word is given as input to the unsupervised tagger.

[2] See §7 for a discussion of the relation of this approach to that of SuperTagging (Srinivas, 1997)

In future work, it would be interesting to extend models for unknown-word handling or other machine learning techniques in clustering or the learning of subcategorization frames to the creation of such tag dictionaries.

4 Models

As described before, we treat parsing as a two-step process. The two models that we use are:

1. H1: selects trees based on previous context (tagging probability model)
2. H2: computes attachments between trees and returns best parse (parsing probability model)

4.1 H1: Tagging probability model

We select the most likely trees for each word by examining the local context. The statistical model we use to decide this is the trigram model that was used by B. Srinivas in his SuperTagging model (Srinivas, 1997). The model assigns an n-best lattice of tree assignments associated with the input sentence with each path corresponding to an assignment of an elementary tree for each word in the sentence. (for further details, see (Srinivas, 1997)).

$$
\begin{aligned}
&P(\mathbf{T}|\mathbf{W}) \\
&\quad = P(T_0 \ldots T_n | W_0 \ldots W_n) && (1)\\
&\quad = \frac{P(T_0 \ldots T_n) \times P(W_0 \ldots W_n | T_0 \ldots T_n)}{P(W_0 \ldots W_n)} && (2)\\
&\quad \approx P(T_i | T_{i-2} T_{i-1}) \times P(W_i | T_i) && (3)
\end{aligned}
$$

where $T_0 \ldots T_n$ is a sequence of elementary trees assigned to the sentence $W_0 \ldots W_n$.

We get (2) by using Bayes theorem and we obtain (3) from (2) by ignore the denominator and by applying the usual Markov assumptions.

The output of this model is a probabilistic ranking of trees for the input sentence which is sensitive to a small local context window.

4.2 H2: Parsing probability model

Once the words in a sentence have selected a set of elementary trees, parsing is the process of attaching these trees together to give us a consistent bracketing of the sentences. Notation: Let τ stand for an elementary tree which is lexicalized by a word: w and a part of speech tag: p.

Let P_{init} (introduced earlier in 2.1) stand for the probability of being root of a derivation tree defined as follows:

$$\sum_{\tau} P_{\mathit{init}}(\tau) = 1$$

including lexical information, this is written as:

$$
\begin{aligned}
&\Pr(\tau, w, p | top = 1) = \\
&\quad \Pr(\tau | top = 1) \times && (4)\\
&\quad \Pr(p | \tau, top = 1) \times && (5)\\
&\quad \Pr(w | \tau, p, top = 1); && (6)
\end{aligned}
$$

where the variable top indicates that τ is the tree that begins the current derivation. There is a useful approximation for P_{init}:

$$\Pr(\tau, w, p | top = 1) \approx \Pr(label | top = 1)$$

where $label$ is the label of the root node of τ.

$$\hat{\Pr}(label | top = 1) = \frac{Count(top = 1, label) + \alpha}{Count(top = 1) + N\alpha} \quad (7)$$

where N is the number of bracketing labels and α is a constant used to smooth zero counts.

Let P_{attach} (introduced earlier in 2.1) stand for the probability of attachment of τ' into another τ:

$$P_{\mathit{attach}}(\tau, \eta \rightarrow \text{NA}) + \sum_{\tau'} P_{\mathit{attach}}(\tau, \eta \rightarrow \tau') = 1$$

including lexical information, this is written as:

$$
\begin{aligned}
&\Pr(\tau', p', w' | Node, \tau, w, p) && (8)\\
&\Pr(\text{NA} | Node, \tau, w, p) && (9)
\end{aligned}
$$

We decompose (8) into the following components:

$$
\begin{aligned}
&\Pr(\tau', p', w' | Node, \tau, w, p) = \\
&\quad \Pr(\tau' | Node, \tau, w, p) \times && (10)\\
&\quad \Pr(p' | \tau', Node, \tau, w, p) \times && (11)\\
&\quad \Pr(w' | p', \tau', Node, \tau, w, p); && (12)
\end{aligned}
$$

We do a similar decomposition for (9).

For each of the equations above, we use a backoff model which is used to handle sparse data problems. We compute a backoff model as follows:

Let e_1 stand for the original lexicalized model and e_2 be the backoff level which only uses part of speech information:

e_1: $Node, \tau, w, p$

e_2: $Node, \tau, p$

For both P_{init} and P_{attach}, let $c = Count(e_1)$. Then the backoff model is computed as follows:

$$\lambda(c) e_1 + (1 - \lambda(c)) e_2$$

where $\lambda(c) = \frac{c}{(c+D)}$ and D is the diversity of e_1 (i.e. the number of distinct counts for e_1).

For P_{attach} we further smooth probabilities (10), (11) and (12). We use (10) as an example, the other two are handled in the same way.

$$\hat{\Pr}(\tau'|Node, \tau, w, p) = \frac{(Count(Node, \tau, w, p, \tau') + \alpha)}{(Count(Node, \tau, w, p) + k\alpha)} \quad (13)$$

$$Count(Node, \tau, w, p) = \sum_{y \in \mathcal{T}'} Count(Node, \tau, w, p, y) \quad (14)$$

where k is the diversity of adjunction, that is: the number of different trees that can attach at that node. $\mathcal{T}'$ is the set of all trees τ' that can possibly attach at Node in tree τ.

For our experiments, the value of α is set to $\frac{1}{100,000}$.

5 Co-Training algorithm

We are now in the position to describe the Co-Training algorithm, which combines the models described in Section 4.1 and in Section 4.2 in order to iteratively label a large pool of unlabeled data.

We use the following datasets in the algorithm:

labeled a set of sentences bracketed with the correct parse trees.

cache a small pool of sentences which is the focus of each iteration of the Co-Training algorithm.

unlabeled a large set of unlabeled sentences. The only information we collect from this set of sentences is a tree-dictionary: *tree-dict* and part-of-speech dictionary: *pos-dict*. Construction of these dictionaries is covered in Section 3.2.

In addition to the above datasets, we also use the usual development test set (termed *dev* in this paper), and a test set (called *test*) which is used to evaluate the bracketing accuracy of the parser.

The Co-Training algorithm consists of the following steps which are repeated iteratively until all the sentences in the set *unlabeled* are exhausted.

1. Input: *labeled* and *unlabeled*
2. Update cache
   - Randomly select sentences from *unlabeled* and refill *cache*
   - If *cache* is empty; exit
3. Train models H1 and H2 using *labeled*
4. Apply H1 and H2 to cache.
5. Pick most probable n from H1 (run through H2) and add to *labeled*.
6. Pick most probable n from H2 and add to *labeled*
7. $n = n + k$; Go to Step 2

For the experiment reported here, $n = 10$, and k was set to be n in each iteration. We ran the algorithm for 12 iterations (covering 20480 of the sentences in *unlabeled*) and then added the best parses for all the remaining sentences.

6 Experiment

6.1 Setup

The experiments we report were done on the Penn Treebank WSJ Corpus (Marcus et al., 1993). The various settings for the Co-Training algorithm (from Section 5) are as follows:

- *labeled* was set to Sections 02-06 of the Penn Treebank WSJ (9625 sentences)
- *unlabeled* was 30137 sentences (Section 07-21 of the Treebank stripped of all annotations).
- A tag dictionary of all lexicalized trees from *labeled* and *unlabeled*.
- Novel trees were treated as unknown tree tokens.
- The *cache* size was 3000 sentences.

While it might seem expensive to run the parser over the cache multiple times, we use the pruning capabilities of the parser to good use here. During the iterations we set the beam size to a value which is likely to prune out all derivations for a large portion of the cache except the most likely ones. This allows the parser to run faster, hence avoiding the usual problem with running an iterative algorithm over thousands of sentences. In the initial runs we also limit the length of the sentences entered into the cache because shorter sentences are more likely to beat out the longer sentences in any case. The beam size is reset when running the parser on the test data to allow the parser a better chance at finding the most likely parse.

6.2 Results

We scored the output of the parser on Section 23 of the Wall Street Journal Penn Treebank. The following are some aspects of the scoring that might be useful for comparision with other results: No punctuations are scored, including sentence final punctuation. Empty elements are not scored. We used EVALB (written by Satoshi Sekine and Michael Collins) which scores based on PARSEVAL (Black et al., 1991); with the standard parameter file (as per standard practice, part of speech brackets were not part of the evaluation). Also, we used Adwait Ratnaparkhi's part-of-speech tagger (Ratnaparkhi, 1996) to tag unknown words in the test data.

We obtained 80.02% and 79.64% labeled bracketing precision and recall respectively (as defined in (Black et al., 1991)). The baseline model which was only trained on the 9695 sentences of labeled data performed at 72.23% and 69.12% precision and recall. These results show that training a statistical parser using our Co-training method to combine labeled and unlabeled data strongly outperforms training only on the labeled data.

It is important to note that unlike previous studies, our method of moving towards unsupervised parsing are directly compared to the output of supervised parsers.

Certain differences in the applicability of the usual methods of smoothing to our parser cause the lower accuracy as compared to other state of the art statistical parsers. However, we have consistently seen increase in performance when using the Co-Training method over the baseline across several trials. It should be emphasised that this is a result based on less than 20% of data that is usually used by other parsers. We are experimenting with the use of an even smaller set of labeled data to investigate the learning curve.

7 Previous Work: Combining Labeled and Unlabeled Data

The two-step procedure used in our Co-Training method for statistical parsing was incipient in the SuperTagger (Srinivas, 1997) which is a statistical model for tagging sentences with elementary lexicalized structures. This was particularly so in the Lightweight Dependency Analyzer (LDA), which used shortest attachment heuristics after an initial SuperTagging stage to find syntactic dependencies between words in a sentence. However, there was no statistical model for attachments and the notion of mutual constraints between these two steps was not exploited in this work.

Previous studies in unsupervised methods for parsing have concentrated on the use of inside-outside algorithm (Lari and Young, 1990; Carroll and Rooth, 1998). However, there are several limitations of the inside-outside algorithm for unsupervised parsing, see (Marcken, 1995) for some experiments that draw out the mismatch between minimizing error rate and iteratively increasing the likelihood of the corpus. Other approaches have tried to move away from phrase structural representations into dependency style parsing (Lafferty et al., 1992; Fong and Wu, 1996). However, there are still inherent computational limitations due to the vast search space (see (Pietra et al., 1994) for discussion). None of these approaches can even be realistically compared to supervised parsers that are trained and tested on the kind of representations and the complexity of sentences that are found in the Penn Treebank.

(Chelba and Jelinek, 1998) combine unlabeled and labeled data for parsing with a view towards language modeling applications. The goal in their work is not to get the right bracketing or dependencies but to reduce the word error rate in a speech recognizer.

Our approach is closely related to previous Co-Training methods (Yarowsky, 1995; Blum and Mitchell, 1998; Goldman and Zhou, 2000; Collins and Singer, 1999). (Yarowsky, 1995) first introduced an iterative method for increasing a small set of seed data used to disambiguate dual word senses by exploiting the constraint that in a segment of discourse only one sense of a word is used. This use of unlabeled data improved performance of the disambiguator above that of purely supervised methods. (Blum and Mitchell, 1998) further embellish this approach and gave it the name of Co-Training. Their definition of Co-Training includes the notion (exploited in this paper) that different models can constrain each other by exploiting different 'views' of the data. They also prove some PAC results on learnability. They also discuss an application of classifying web pages by using their method of mutually constrained models. (Collins and Singer, 1999) further extend the use of classifiers that have mutual constraints by adding terms to AdaBoost which force the classifiers to agree (called Co-Boosting). (Goldman and Zhou, 2000) provide a variant of Co-Training which is suited to the learning of decision trees where the data is split up into different equivalence classes for each of the models and they use hypothesis testing to determine the agreement between the models. In future work we would like to experiment whether some of these ideas could be incorporated into our model.

In future work we would like to explore use of the entire 1M words of the WSJ Penn Treebank as our labeled data and to use a larger set of unbracketed WSJ data as input to the Co-Training algorithm. In addition, we plan to explore the following points that bear on understanding the nature of the Co-Training learning algorithm:

- The contribution of the dictionary of trees extracted from the unlabeled set is an issue that we would like to explore in future experiments. Ideally, we wish to design a co-training method where no such information is used from the unlabeled set.
- The relationship between co-training and EM bears investigation. (Nigam and Ghani, 2000) is a study which tries to separate two factors: (1) The gradient descent aspect of EM vs. the iterative nature of co-training and (2) The generative model used in EM vs. the conditional independence between the features used by the two models that is exploited in co-training. Also, EM has been used successfully in text classification in combination of labeled and unlabeled data (see (Nigam et al., 1999)).
- In our experiments, unlike (Blum and Mitchell, 1998) we do not balance the label priors when picking new labeled examples for addition to the training data. One way to incorporate this into our algorithm would be to incorporate some form of sample selection (or active learning) into the selection of examples that are considered as labeled with high

confidence (Hwa, 2000).

8 Conclusion

In this paper, we proposed a new approach for training a statistical parser that combines labeled with unlabeled data. It uses a Co-Training method where a pair of models attempt to increase their agreement on labeling the data. The algorithm takes as input a small corpus of 9695 sentences (234467 word tokens) of bracketed data, a large pool of unlabeled text and a tag dictionary of lexicalized structures for each word in this training set (based on the LTAG formalism). The algorithm presented iteratively labels the unlabeled data set with parse trees. We then train a statistical parser on the combined set of labeled and unlabeled data.

We obtained 80.02% and 79.64% labeled bracketing precision and recall respectively. The baseline model which was only trained on the 9695 sentences of labeled data performed at 72.23% and 69.12% precision and recall. These results show that training a statistical parser using our Co-training method to combine labeled and unlabeled data strongly outperforms training only on the labeled data.

It is important to note that unlike previous studies, our method of moving towards unsupervised parsing can be directly compared to the output of supervised parsers. Unlike previous approaches to unsupervised parsing our method can be trained and tested on the kind of representations and the complexity of sentences that are found in the Penn Treebank.

In addition, as a byproduct of our representation we obtain more than the phrase structure of each sentence. We also produce a more embellished parse in which phenomena such as predicate-argument structure, subcategorization and movement are given a probabilistic treatment.

References

E. Black, S. Abney, D. Flickinger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski. 1991. A procedure for quantitatively comparing the syntactic coverage of english grammars. In *Proc. DARPA Speech and Natural Language Workshop*, pages 306–311. Morgan Kaufmann.

A. Blum and T. Mitchell. 1998. Combining Labeled and Unlabeled Data with Co-Training. In *Proc. of 11th Annual Conf. on Comp. Learning Theory (COLT)*, pages 92–100.

E. Brill. 1997. Unsupervised learning of disambiguation rules for part of speech tagging. In *Natural Language Processing Using Very Large Corpora*. Kluwer Academic Press.

G. Carroll and M. Rooth. 1998. Valence Induction with a Head-Lexicalized PCFG. `http://xxx.lanl.gov/abs/cmp-lg/9805001`, May.

C. Chelba and F. Jelinek. 1998. Exploiting syntactic structure for language modeling. In *Proc. of COLING-ACL '98*, pages 225–231, Montreal.

M. Collins and Y. Singer. 1999. Unsupervised Models for Named Entity Classification. In *Proc. of WVLC/EMNLP-99*, pages 100–110.

D. Cutting, J. Kupiec, J. Pedersen, and P. Sibun. 1992. A practical part-of-speech tagger. In *Proc. of 3rd ANLP Conf.*, Trento, Italy. ACL.

D. Elworthy. 1994. Does baum-welch re-estimation help taggers? In *Proc. of 4th ANLP Conf.*, pages 53–58, Stuttgart, October 13-15.

E. W. Fong and D. Wu. 1996. Learning restricted probabilistic link grammars. In S. Wermter, E. Riloff, and G. Scheler, editors, *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing*, pages 173–187. Springer-Verlag.

S. Goldman and Y. Zhou. 2000. Enhancing supervised learning with unlabeled data. In *Proc. of ICML'2000*, Stanford University, June 29–July 2.

Rebecca Hwa. 2000. Sample selection for statistical grammar induction. In *Proceedings of EMNLP/VLC-2000*, pages 45–52.

A. K. Joshi and Y. Schabes. 1992. Tree-adjoining grammar and lexicalized grammars. In M. Nivat and A. Podelski, editors, *Tree automata and languages*, pages 409–431. Elsevier Science.

A. K. Joshi, L. Levy, and M. Takahashi. 1975. Tree Adjunct Grammars. *Journal of Computer and System Sciences*.

A. K. Joshi. 1985. Tree Adjoining Grammars: How much context Sensitivity is required to provide a reasonable structural description. In D. Dowty, I. Karttunen, and A. Zwicky, editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press, Cambridge, U.K.

J. Lafferty, D. Sleator, and D. Temperley. 1992. Grammatical trigrams: A probabilistic model of link grammar. In *Proc. of the AAAI Conf. on Probabilistic Approaches to Natural Language*.

K. Lari and S. J. Young. 1990. The estimation of stochastic context-free grammars using the Inside-Outside algorithm. *Computer Speech and Language*, 4:35–56.

C. de Marcken. 1995. Lexical heads, phrase structure and the induction of grammar. In D. Yarowsky and K. Church, editors, *Proc. of 3rd WVLC*, pages 14–26, MIT, Cambridge, MA.

M. Marcus, B. Santorini, and M. Marcinkiewiecz. 1993. Building a large annotated corpus of english. *Computational Linguistics*, 19(2):313–330.

B. Merialdo. 1994. Tagging english text with a probabilistic model. *Computational Linguistics*, 20(2):155–172.

Kamal Nigam and Rayid Ghani. 2000. Analyzing the effectiveness and applicability of co-training. In *Proc. of Ninth International Conference on Information and Knowledge (CIKM-2000)*.

Kamal Nigam, Andrew McCallum, Sebastian Thrun, and Tom Mitchell. 1999. Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning*, 1(34).

S. Della Pietra, V. Della Pietra, J. Gillett, J. Lafferty, H. Printz, and L. Ureš. 1994. Inference and estimation of a long-range trigram model. In R. Carrasco and J. Oncina, editors, *Proc. of ICGI-94*. Springer-Verlag.

A. Ratnaparkhi. 1996. A Maximum Entropy Part-Of-Speech Tagger. In *Proc. of EMNLP-96*, University of Pennsylvania.

P. Resnik. 1992. Probabilistic tree-adjoining grammars as a framework for statistical natural language processing. In *Proc. of COLING '92*, volume 2, pages 418–424, Nantes, France.

Y. Schabes. 1992. Stochastic lexicalized tree-adjoining grammars. In *Proc. of COLING '92*, volume 2, pages 426–432, Nantes, France.

B. Srinivas. 1997. *Complexity of Lexical Descriptions and its Relevance to Partial Parsing*. Ph.D. thesis, Department of Computer and Information Sciences, University of Pennsylvania.

F. Xia, M. Palmer, and A. Joshi. 2000. A Uniform Method of Grammar Extraction and its Applications. In *Proc. of EMNLP/VLC-2000*.

D. Yarowsky. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proc. 33rd Meeting of the ACL*, pages 189–196, Cambridge, MA.