On Minimizing Training Corpus for Parser Acquisition *

Rebecca Hwa
Institute for Advanced Computer Studies,
University of Maryland
College Park, MD 20742 USA
hwa@umiacs.umd.edu

Abstract

Many corpus-based natural language processing systems rely on using large quantities of annotated text as their training examples. Building this kind of resource is an expensive and labor-intensive project. To minimize effort spent on annotating examples that are not helpful the training process, recent research efforts have begun to apply active learning techniques to selectively choose data to be annotated. In this work, we consider selecting training examples with the *tree-entropy* metric. Our goal is to assess how well this selection technique can be applied for training different types of parsers. We find that tree-entropy can significantly reduce the amount of training annotation for both a history-based parser and an EM-based parser. Moreover, the examples selected for the history-based parser are also good for training the EM-based parser, suggesting that the technique is parser independent.

1 Introduction

In recent years, large collections of text in machine readable format have become readily available. These ought be valuable resources for training natural language processing system. Unfortunately, most systems cannot take advantage of the data in their raw text form; typically, the data must be annotated by a human to become effective training examples. For instance, consider the task of inducing a grammar to parse English sentences. Studies have shown that a grammar trained on sentences annotated with their constituent trees produces much better parses than one trained on just the sentences alone (Pereira and Schabes, 1992). Recent state-of-the-art parsers developed by Collins (1997) and Charniak (1999) are all trained from hand-annotated corpora such as those from the Penn Treebank Project (Marcus et al., 1993). However, building an annotated corpus is a human labor-intensive project; therefore, it is important to find ways to minimize the size of the corpus.

Out of a large pool of raw text, what subset should be annotated and added to the training set? Recent studies have begun to address this question using *sample selection*, in which training process is seen as interactive session between the learning system and the human annotator (Lewis and Gale, 1994), (Engelson and Dagan, 1996), (Fujii et al., 1998), (Thompson et al., 1999), and (Ngai and Yarowsky, 2000). The system actively influences its learning progress by evaluating potential candidates from the pool of raw text and selecting those with high *Training Utility Values* (TUV) for humans to annotate. As the learning process continues, the system should become better at identifying good training candidates so that the annotators would not

*This material is based upon work supported by the National Science Foundation under Grant No. IRI 9712068 and DARPA contract N6600197C8540. We thank Michael Collins for the use of his parser; and Ric Crabbe and the anonymous reviewers for their comments on the paper.

need to waste time on processing uninformative examples.

This work considers the problem of applying sample selection techniques to the task of training statistical parsers. Our primary challenge is in designing a function that can accurately estimate an unlabeled candidate's potential utility for training a parser. In a previous study (Hwa, 2000b), we have applied sample selection to an induction algorithm based on the expectation-maximization (EM) principle that induces Probabilistic Lexicalized Tree Insertion Grammars (PLTIGs). In that work, we proposed an uncertainty-based evaluation function to estimate the TUV of unlabeled candidates called *tree entropy*. We have empirically shown that sample selection with tree entropy can reduce the size of the training corpus significantly. However, because only an EM-based learner was used, it is unknown whether the evaluation function would be general enough to be applicable to other types of learners. The goal of this work is to assess the robustness of the tree-entropy evaluation function. We have performed experiments to evaluate how well the metric selects training examples for different types of parsers and to determine whether examples selected for one type of parser might be good for training a different type of parser. Our experimental results show that the tree-entropy metric can reduce the amount of training annotation by 23% for a history-based lexical statistical parser, the Model 2 parser described by Collins (1997). Moreover, we found that the data selected for training the Collins Parser also make good training examples for inducing the EM-based PLTIG parser, suggesting that the tree-entropy evaluation function is parser independent.

2 The Learning Framework

There are two types of sample selection algorithms: *committee based* or *single learner*. A committee-based selection algorithm works with multiple learners, each maintaining a different hypothesis (perhaps pertaining to different aspects of the problem). The candidate examples that lead to the most disagreements among the different learners are considered to have the highest TUV. (Cohn et al., 1994; Freund et al., 1997). For computationally intensive problems, keeping multiple learners may be impractical. In this work, we focus on sample selection algorithms that use only a single learner that keeps just one working hypothesis. Without access to multiple hypotheses, the selection algorithm can nonetheless estimate the TUV of an example. We categorize some possible ranking criteria into the following three classes:

U is a set of unlabeled candidates.
L is a set of labeled training examples.
M is the current model.
Initialize:
  $M \leftarrow Train(L)$.
Repeat
  $N \leftarrow Select(n, U, M, f)$.
  $U \leftarrow U - N$.
  $L \leftarrow L \cup Label(N)$.
  $M \leftarrow Train(L)$.
Until $(M \approx M_{true})$ **or**
  $(U = \emptyset)$ **or** (human stops).

Figure 1: The pseudo-code for the sample selection learning algorithm

Problem-space: Knowledge about the problem-space may help to locate good training canidates. For example, knowing the distribution of the pool, we might select the most frequently occuring instances.

Performance of the hypothesis: Testing the candidates on the current hypothesis may show the type of data on which the hypothesis performs weakly (Lewis and Catlett, 1994).

Parameters of the hypothesis: Estimating the potential impact of the candidates will have on the parameters of the current working hypothesis locates those examples that will change the current hypothesis the most.

Figure 1 outlines the single-learner sample selection training loop in pseudo-code. Initially, the training set, L, consists of a small number of labeled examples. The learner uses L to train an initial model M. Also available to the learner is a large pool of unlabeled training candidates, U. In each iteration, the selection algorithm, $Select(n, U, M, f)$, uses an evaluation function f to compute the expected TUV of each candidate in U and returns the n candidates with the highest values. The set of the n chosen candidates are then labeled by human experts and added to the existing training set. Training on the updated set L, the system modifies the model so that it is consistent with all the examples seen thus far. The loop continues until one of the stopping conditions is met: the model is considered to be good enough, all candidates are labeled, or all human resources are used up.

2.1 The Evaluation Function

At the heart of the sample selection algorithm is the evaluation function that predicts each unlabeled candidate's training utility. Our proposed function ranks candidates based on the "performance of hypothesis." In other words, we wish to find the set of sentences that the current parsing model is the most uncertain about. One way to measure the parser's uncertainty is to compute the *tree entropy* over the distribution of parsing probabilities of the set of trees produced by the parser. More specifically, the tree entropy for a sentence u is:

$$TE(u, M) = -\sum_{t \in \mathcal{T}} Pr(t|u, M) \log_2 Pr(t|u, M),$$

where $\mathcal{T}$ is the set of possible trees that M generated for u. Details of computing tree entropy have been discussed previously (Hwa, 2000b). Our proposed function evaluates each candidate by measuring the similarity between the tree entropy of the candidate and the uniform distribution for the same number of trees. That is,

$$f_{te}(u, M) = \frac{TE(u, M)}{\log_2 |\mathcal{T}|}$$

2.2 Parsing Models

To test the robustness of the tree-entropy evaluation function, we use it to select training examples for the Collins Parser and the PLTIG parser. Although both are lexicalized and statistical parsers, their learning algorithms are different. The Collins Parser is a fully-supervised, history-based learner that models the parameters of the parser by taking statistics directly from the training data. In contrast, PLTIG's EM-based induction algorithm (Hwa, 2000a) is partially-supervised; the model's parameters are estimated indirectly from the training data. Our goal for this study is to determine whether the success of the tree-entropy metric is learner dependent.

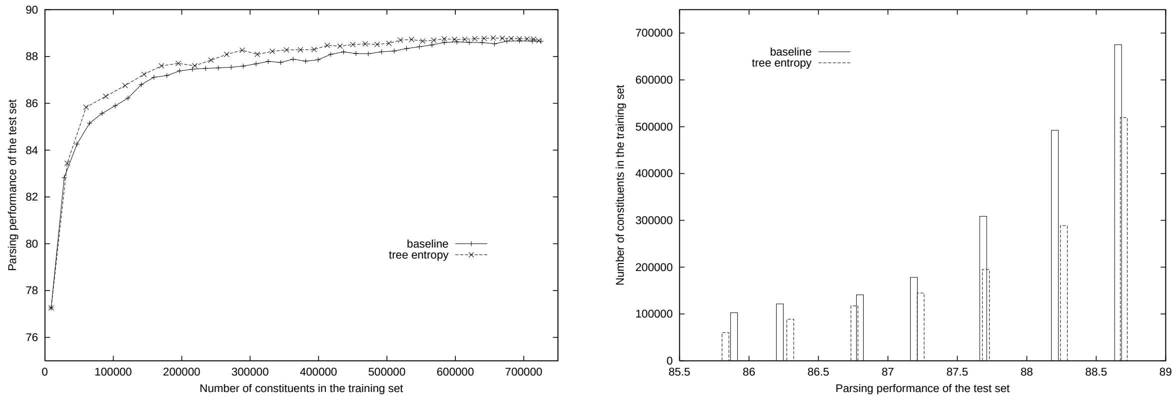
3 Experimental Setup and Results

Two experiments are performed. The first experiment assesses whether the tree-entropy evaluation function can select good examples for a history-based learner. The second experiment is a preliminary study on whether the examples selected for a history-based learner are also good training examples for a EM-based learner.

3.1 Experiment 1

We use the Collins Parser as the basic learning model M in the sample selection framework described in Figure 1. To simulate the interactive process, we create a large unlabeled candidate pool U by stripping all annotated information from sections 02 through 21 of the Wall Street Journal corpus. Initially, L, the set of labeled training data, consists of 500 parsed sentences. In each iteration, $n = 1000$ new sentences are picked from U to be added to L. Then, a new parser is trained from the updated L and tested on section 00 to chart the learning progress.

We compare the learning rate of the parser trained on examples selected by the tree entropy evaluation function, f_{te} with a baseline in which the model was trained with examples sequentially selected. The experimental results are graphed in Figure 2(a). The



(a) (b)

Figure 2: (a) A graph comparing the learning rates of the Collins Parser under two training conditions: "baseline" shows the progress of sequential training and "tree entropy" shows the progress of sample selection training. (b) A graph showing the relative amounts of annotated training data used to achieve the same performance level by the two evaluation functions.

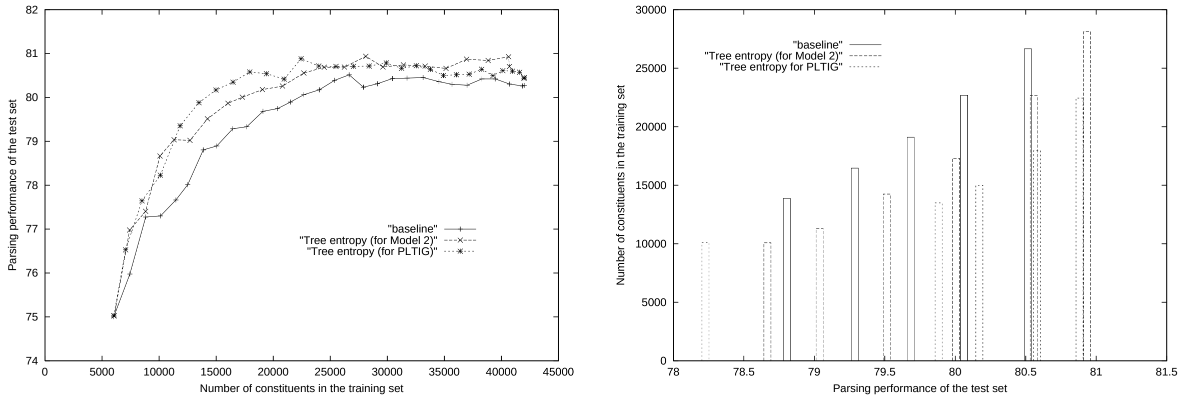
parsing performance on the test sentences (using the combined labeled precision and label recall score[1] as the metric (Van Rijsbergen, 1979)) is graphed as a function of the number of labeled constituents in the training data. We use the number of *constituents* rather than the number of sentences because it is a better indicator of the effort spent by the human annotator. Longer sentences tend to require more annotation than short ones, and thus take more time to analyze.

Our results suggest that the parser learns faster when trained from examples selected by f_{te}. The learning rates of the parser under the two training conditions are plotted in Figure 2(a). The graph shows that, for a comparable amount of annotated constituents in the training data, the parser trained on examples selected by f_{te} typically performs better on unseen test data than the baseline. Another way of interpreting the results is to say that the same parsing performance can be achieved using fewer annotated training examples. Figure 2(b) graphs the amount of reduction in annotated training constituents that f_{te} offers from the baseline given comparable parsing performances. For the final parsing performance of 88.7%, the parser requires a baseline training set of 36,500 sentences annotated with about 675,000 constituents. In contrast, the same performance can be achieved using a training set of 520,000 constituents in the 23,500 sentences selected by f_{te}, reducing the number of annotated constituents by 23%.

3.2 Experiment 2

To determine the suitability of the selected training examples across different learners, we now use PLTIG as the basic learning model and compare the parsing performances of three PLTIGs induced from different sets of training sentences: those selected by the tree-entropy evaluation function for a PLTIG model, $f_{te}(u, M_{PLTIG})$, those selected for the Collins Parser in the previous experiment, $f_{te}(u, M_{Collins})$, and the baseline of sequential selection. Some modifications to the experimental setup of the previous experiments are necessary to accommodate the EM-based induction algorithm for PLTIG. Because EM-based grammar induction is computationally expensive, this experiment is limited to using an unlabeled pool of 3600 sentences, and the grammars are lexicalized to part-of-speech tags rather than words. Moreover, because the algorithm induces grammars that generate binary branching trees, we evaluate the parsing accuracy of the test sentences with the

[1] $F_{\beta=1} = \frac{2\times LR\times LP}{LR+LP}$, where LR is the labeled recall score and LP is the labeled precision score.



(a) (b)

Figure 3: (a) A graph comparing the learning rates of three PLTIGs induced from different sets of training examples. (b) A graph showing the relative amounts of annotated training data used to induce the PLTIGs.

consistent bracketing metric (i.e., the percentage of constituents in the proposed parse not crossing constituents of the true parse) rather than the average precision and recall metric[2].

Currently, one trial of this experiment has been completed. While a more comprehensive analysis is required, our initial results suggest that the examples selected by the tree-entropy metric are informative independent of the underlying learning model. Comparing the learning rates of the three PLTIGs graphed in Figure 3(a), we see that although learning rate of the grammar trained on examples selected specifically for the PLTIG model is faster than the one trained on examples selected for the Collins Parser, both are better than the baseline. The bar graph in Figure 3(b) shows the relative amounts of annotation used to train each grammar. To achieve a parsing level comparable to the baseline's best performance, the grammar trained on examples selected for the Collins Parser needed about 15% less annotation than the baseline, and the grammar trained on examples selected for itself needed about 33%[3] less annotation than the baseline. Both induced grammars achieved slightly higher parsing accuracy than the baseline when trained on all examples.

4 Conclusion and Future Work

In this paper, we have assessed the robustness of the tree-entropy evaluation function as a metric of training utility values for different types of parsers. We have empirically shown that tree-entropy can select informative training examples and reduce the amount of training annotation for both a history-based learner and an EM-based learner. Moreover, we have found that the training examples selected for the history-based parser are also informative for training the EM-based parser.

In addition to the tree-entropy evaluation function, which uses the performance of the hypothesis as the ranking criterion, we are exploring alternative evaluation functions that use problem-space based and parameter-confidence based ranking criteria.

References

Eugene Charniak. 1999. A maximum-entropy inspired parser. Technical Report CS-99-12, Brown University.

[2] The number of proposed constituents in a binary branching tree is always one fewer than the length of the sentence. The WSJ corpus, on the other hand, favors a more flattened tree structure with considerably fewer brackets per sentence. The consistent bracketing metric does not unfairly penalize a proposed parse tree for being binary branching.

[3] The figure reported in our previous study of 36% was the average of 10 trials.

David Cohn, Les Atlas, and Richard Ladner. 1994. Improving generalization with active learning. *Machine Learning*, 15(2):201–221.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the ACL*, pages 16–23, Madrid, Spain.

Sean P. Engelson and Ido Dagan. 1996. Minimizing manual annotation cost in supervised training from copora. In *Proceedings of the 34th Annual Meeting of the ACL*, pages 319–326.

Yoav Freund, H. Sebastian Seung, Eli Shamir, and Naftali Tishby. 1997. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2-3):133–168.

Atsushi Fujii, Kentaro Inui, Takenobu Tokunaga, and Hozumi Tanaka. 1998. Selective sampling for example-based word sense disambiguation. *Computational Linguistics*, 24(4):573–598, December.

Rebecca Hwa. 2000a. *Learning Probabilistic Lexicalized Grammars for Natural Language Processing*. Ph.D. thesis, Harvard University.

Rebecca Hwa. 2000b. Sample selection for statistical grammar induction. In *Proceedings of the 2000 Joint SIGDAT Conference on EMNLP and VLC*, pages 45–52, Hong Kong, China, October.

David D. Lewis and Jason Catlett. 1994. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 148–156.

David D. Lewis and William A. Gale. 1994. A sequential algorithm for training text classifiers. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Grace Ngai and David Yarowsky. 2000. Rule writing or annotation: Cost-efficient resource usage for base noun phrase chunking. In *Proceedings of the 38th Annual Meeting of the ACL*, pages 117–125, Hong Kong, China, October.

Fernando Pereira and Yves Schabes. 1992. Inside-Outside reestimation from partially bracketed corpora. In *Proceedings of the 30th Annual Meeting of the ACL*, pages 128–135, Newark, Delaware.

Cynthia A. Thompson, Mary Elaine Califf, and Raymond J. Mooney. 1999. Active learning for natural language parsing and information extraction. In *Proceedings of ICML-99*, pages 406–414, Bled, Slovenia.

C. J. Van Rijsbergen. 1979. *Information Retrieval*. Butterworth.