Active learning for HPSG parse selection

Jason Baldridge and Miles Osborne
School of Informatics
University of Edinburgh
Edinburgh EH8 9LW, UK
{jmb,osborne}@cogsci.ed.ac.uk

Abstract

We describe new features and algorithms for HPSG parse selection models and address the task of creating annotated material to train them. We evaluate the ability of several sample selection methods to reduce the number of annotated sentences necessary to achieve a given level of performance. Our best method achieves a 60% reduction in the amount of training material without any loss in accuracy.

1 Introduction

Even with significant resources such as the Penn Treebank, a major bottleneck for improving statistical parsers has been the lack of sufficient annotated material from which to estimate their parameters. Most statistical parsing research, such as Collins (1997), has centered on training probabilistic context-free grammars using the Penn Treebank. For richer linguistic frameworks, such as Head-Driven Phrase Structure Grammar (HPSG), there is even less annotated material available for training stochastic parsing models. There is thus a pressing need to create significant volumes of annotated material in a logistically efficient manner. Even if it were possible to bootstrap from the Penn Treebank, it is still unlikely that there would be sufficient quantities of high quality material.

There has been a strong focus in recent years on using the active learning technique of *selective sampling* to reduce the amount of human-annotated training material needed to train models for various natural language processing tasks. The aim of selective sampling is to identify the most informative examples, according to some selection method, from a large pool of unlabelled material. Such selected examples are then manually labelled. Selective sampling has typically been applied to classification tasks, but has also been shown to reduce the number of examples needed for inducing Lexicalized Tree Insertion Grammars from the Penn Treebank (Hwa, 2000).

The suitability of active learning for HPSG-type grammars has as yet not been explored. This paper addresses the problem of minimizing the human effort expended in creating annotated training material for HPSG parse selection by using selective sampling. We do so in the context of Redwoods (Oepen et al., 2002), a treebank that contains HPSG analyses for sentences from the Verbmobil appointment scheduling and travel planning domains. We show that sample selection metrics based on tree entropy (Hwa, 2000) and disagreement between two different parse selection models significantly reduce the number of annotated sentences necessary to match a given level of performance according to random selection. Furthermore, by combining these two methods as an ensemble selection method, we require even fewer examples — achieving a 60% reduction in the amount of annotated training material needed to outperform a model trained on randomly selected material. These results suggest that significant reductions in human effort can be realized through selective sampling when creating annotated material for linguistically rich grammar formalisms.

As the basis of our active learning approach, we create both log-linear and perceptron models, the latter of which has not previously been used for feature-based grammars. We show that the different biases of the two types of models is sufficient to create diverse members for a committee, even when they use exactly the same features. With respect to the features used to train models, we demonstrate that a very simple feature selection strategy that ignores the proper structure of trees is competitive with one that fully respects tree configurations.

The structure of the paper is as follows. In sections 2 and 3, we briefly introduce active learning and the Redwoods treebank. Section 4 discusses the parse selection models that we use in the experiments. In sections 5 and 6, we explain the different selection methods that we use for active learning and explicate the setup in which the experiments were conducted. Finally, the results of the experiments are presented and discussed in section 7.

2 Active Learning

Active learning attempts to reduce the number of examples needed for training statistical models by allowing the machine learner to directly participate in creating the corpus it uses. There are a several approaches to active learning; here, we focus on selective sampling (Cohn et al., 1994), which involves identifying the most informative examples from a pool of unlabelled data and presenting only these examples to a human expert for annotation. The two main flavors of selective sampling are *certainty-based* methods and *committee-based* methods (Thompson et al., 1999). For certainty-based selection, the examples chosen for annotation are those for which a single learner is least confident, as determined by some criterion. Committee-based selection involves groups of learners that each maintain different hypotheses about the problem; examples on which the learners disagree in some respect are typically regarded as the most informative.

Active learning has been successfully applied to a number of natural language oriented tasks, including text categorization (Lewis and Gale, 1994) and part-of-speech tagging (Engelson and Dagan, 1996). Hwa (2000) shows that certainty-based selective sampling can reduce the amount of training material needed for inducing Probabilistic Lexicalized Tree Insertion Grammars by 36% without degrading the quality of the grammars. Like Hwa, we investigate active learning for parsing and thus seek informative sentences; however, rather than inducing grammars, our task is to select the best parse from the output of an existing hand-crafted grammar by using the Redwoods treebank.

3 The Redwoods Treebank

The English Resource Grammar (ERG, Flickinger (2000)) is a broad coverage HPSG grammar that provides deep semantic analyses of sentences but has no means to prefer some analyses over others because of its purely symbolic nature. To address this limitation, the Redwoods treebank has been created to provide annotated training material to permit statistical models for ambiguity resolution to be combined with the precise interpretations produced by the ERG (Oepen et al., 2002).

Whereas the Penn Treebank has an implicit grammar underlying its parse trees, Redwoods uses the ERG explicitly. For each utterance, Redwoods enumerates the set of analyses, represented as derivation trees, licensed by the ERG and identifies which analysis is the preferred one. For example, Figure 1 shows the preferred derivation tree, out of three ERG analyses, for *what can I do for you?*. From such derivation trees, the parse trees and semantic interpretations can be recovered using an HPSG parser.

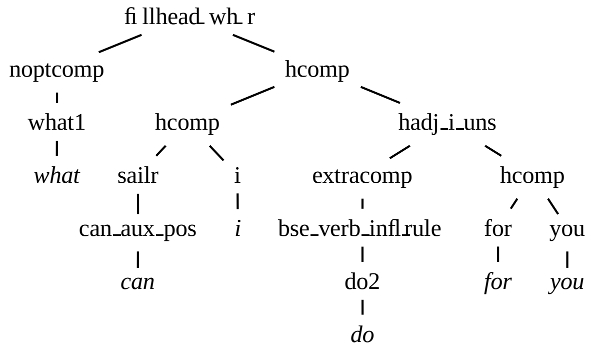


Figure 1: Redwoods derivation tree for the sentence *what can I do for you?* The node labels are the names of the ERG rules used to build the analysis.

Redwoods is (semi-automatically) updated after changes have been made to the ERG, and it has thus far gone through three growths. Some salient characteristics of the first and third growths are given in Table 1 for utterances for which a unique preferred parse has been identified and for which there are at least two analyses.[1] The ambiguity increased considerably between the first and third growths, reflecting the increased coverage of the ERG for more difficult sentences.

corpus	sentences	length	parses
Redwoods-1	3799	7.9	9.7
Redwoods-3	5302	9.3	58.0

Table 1: Characteristics of subsets of Redwoods versions used for the parse selection task. The columns indicate the number of sentences in the subset, their average length, and their average number of parses.

The small size of the treebank makes it essential to explore the possibility of using methods such as active learning to speed the creation of more annotated material for training parse selection models.

4 Parse Selection

Committee-based active learning requires multiple learners which have different biases that cause them to make different predictions sometimes. As in co-training, one

[1]There are over 1400 utterances in both versions for which the ERG produces only one analysis and which therefore are irrelevant for parse selection. They contain no discriminating information and are thus not useful for the machine learning algorithms discussed in the next section.

uni $<)^{2}(^{1}$hcomp$>$
bi $<)^{0}(^{1}$what1$>$ $<)^{2}(^{1}$hcomp$>$
tri $<)^{0}(^{1}$noptcomp$>$ $<)^{0}(^{1}$what1$>$ $<)^{2}(^{1}$hcomp$>$

Figure 2: Three example ngram features based on the derivation tree in Figure 1.

way such diverse learners can be created is by using independent or partially independent feature sets to reduce the error correlation between the learners. Another way is to use different machine learning algorithms trained on the same feature set. In this section, we discuss two feature sets and two machine learning algorithms that are used to produce four distinct models and we give their overall performance on the parse selection task.

4.1 Features

Our two feature sets are created by using only the derivation trees made available in Redwoods. The ***configurational*** set is loosely based on the derivation tree features given by Toutanova and Manning (2002), and thus encodes standard relations such as grandparent-of and left-sibling for the nodes in the tree. The ***ngram*** set is created by flattening derivation trees and treating them as strings of rule names over which ngrams are extracted, taking up to four rule names at a time and including the number of intervening parentheses between them. We ignore orthographic values for both feature sets.

As examples of typical ngram features, the derivation tree given in Figure 1 generates features such as those depicted in Figure 2. Such features provide a reasonable approximation of trees that implicitly encodes many of the interesting relationships that are typically gathered from them, such as grandparent and sibling relations. They also capture further relationships that cross the brackets of the actual tree, providing some more long-distance relationships than the configurational features.

4.2 Algorithms

We use both log-linear and perceptron algorithms to create parse selection models. Both frameworks use iterative procedures to determine the weights $\langle w_1, \ldots, w_m \rangle$ of a corresponding set of features $\{f_1, \ldots, f_m\}$ produced from annotated training material. Though they are otherwise quite different, this commonality facilitates their use in a committee since they can work with the same training material. When preparing the training material, we record observations about the distribution of analyses with a binary distinction that simply identifies the preferred parse, rather than using a full regression approach that recognizes similarities between the preferred parse and some of the dispreferred ones.

Log-linear models have previously been used for stochastic unification-based grammars by Johnson et al. (1999) and Osborne (2000). Using Redwoods-1, Toutanova and Manning (2002) have shown that log-linear models for parse selection considerably outperform PCFG models trained on the same features. By using features based on both derivation trees and semantic dependency trees, they achieved 83.32% exact match whole-sentence parse selection with an an ensemble of log-linear models that used different subsets of the feature space.

As standard for parse selection using log-linear modelling, we model the probability of an analysis t_i given a sentence with a set of analyses $\tau = \{t_1 \ldots t_k\}$ as follows:

$$P(t_i|s) = \frac{exp(\sum_{j=1}^{m} f_j(t_i)w_j)}{Z(s)}$$

where $f_j(t)$ returns the number of times feature j occurs in analysis t and $Z(s)$ is a normalization factor for the sentence. The parse with the highest probability is taken as the preferred parse for the model.[2] We use the limited memory variable metric algorithm (Malouf, 2002) to determine the weights.

Perceptrons have been used by Collins and Duffy (2002) to re-rank the output of a PCFG, but have not previously been applied to feature-based grammars. Standard perceptrons assign a score rather than probability to each analysis. Scores are computed by taking the inner product of the analysis' feature vector with the parameter vector:

$$score(t_i) = \sum_{j=1}^{m} f_j(t_i)w_j$$

The preferred parse is that with the highest score out of all analyses of a sentence.

4.3 Performance

Using the two feature sets (configurational and ngram) with both log-linear and perceptron algorithms, we create the four models shown in Table 2. To test their overall accuracy, we measured performance using exact match. This means we award a model a point if it picks some parse for a sentence and that parse happens to be the best analysis. We averaged performance over ten runs using a cross-validation strategy. For each run, we randomly split the corpus into ten roughly equally-sized subsets and tested the accuracy for each subset after training a model on the other nine. The accuracy when a model ranks m parses highest is given as $1/m$.

The results for the four models on both Redwoods-1 and Redwoods-3 are given in Table 3, along with a baseline of randomly selecting parses. As can be seen, the increased ambiguity in the later version impacts the ac-

Model	Algorithm	Feature set
LL-CONFIG	log-linear	configurational
LL-NGRAM	log-linear	ngram
PT-CONFIG	perceptron	configurational
PT-NGRAM	perceptron	ngram

Table 2: Parse selection models.

Model	Redwoods-1	Redwoods-3
RANDOM	25.71	22.70
LL-CONFIG	81.84	74.90
LL-NGRAM	81.35	74.05
PT-CONFIG	79.92	71.76
PT-NGRAM	79.92	72.75

Table 3: Parse selection accuracy.

curacy heavily.

The performance of LL-CONFIG on Redwoods-1 matches the accuracy of the best stand-alone log-linear model reported by Toutanova and Manning (2002), which uses essentially the same features. The log-linear model that utilizes the ngram features is not far behind, indicating that these simple features do indeed capture important generalizations about the derivation trees.

The perceptrons both perform worse than the log-linear models. However, what is more important is that each model disagrees with all of the others on roughly 20% of the examples, indicating that differentiation by using either a different feature set or a different machine learning algorithm is sufficient to produce models with different biases. This is essential for setting up committee-based active learning and could also make them informative members in an ensemble for parse selection.

5 Selecting Examples for Annotation

In applying active learning to parse selection, we investigate two primary sample selection methods, one certainty-based and the other committee-based, and compare them to several baseline methods.

The single-learner method uses ***tree entropy*** (Hwa, 2000), which measures the uncertainty of a learner based on the conditional distribution it assigns to the parses of a given sentence. Following Hwa, we use the following evaluation function to quantify uncertainty based on tree entropy:

$$f_{te}(s,\tau) = \frac{-\sum_{t\in\tau} p(t|s) log_2(p(t|s))}{length(s)}$$

where τ denotes the set of analyses produced by the ERG for the sentence. Higher values of $f_{te}(s,\tau)$ indicate examples on which the learner is most uncertain and thus presumably are more informative. The intuition behind tree entropy is that sentences should have a skewed distribution over their parses and that deviation from this signals learner uncertainty. Calculating tree entropy is trivial with the conditional log-linear models described in section 4. Of course, tree entropy cannot be straightforwardly used with standard perceptrons since they do not determine a distribution over the parses of a sentence.

The second sample selection method is inspired by the Query by Committee algorithm (Freund et al., 1997; Argamon-Engelson and Dagan, 1999) and co-testing (Muslea et al., 2000). Using a fixed committee consisting of two distinct models, the examples we select for annotation are those for which the two models disagree on the preferred parse. We will refer to this method as ***preferred parse disagreement***. The intuition behind this method is that the different biases of each of the learners will lead to different predictions on some examples and thus identify examples for which at least one of them is uncertain.

We compare tree entropy and disagreement with the following three baseline selection methods to ensure the significance of the results:

- ***random***: randomly select sentences
- ***ambiguity***: select sentences with a higher number of parses
- ***length***: select longer sentences

6 Experimental Setup

The pseudo-code for committee-based active learning with two members is given in Figure 3.[3] Starting with a small amount of initial annotated training material, the learners on the committee are used to select examples, based on the method being used. These examples are subsequently manually annotated and added to the set of labelled training material and the learners are retrained on the extended set. This loop continues until all available unannotated examples are exhausted, or until some other pre-determined condition is met.

As is standard for active learning experiments, we quantify the effect of different selection techniques by using them to select subsets of the material already annotated in Redwoods-3. For the experiments, we used tenfold cross-validation by moving a fixed window of 500 sentences through Redwoods-3 for the test set and selecting samples from the remaining 4802 sentences. Each run of active learning begins with 50 randomly chosen, annotated seed sentences. At each round, new examples

[2]When only an absolute ranking of analyses is required, it is unnecessary to exponentiate and compute $Z(s)$.

[3]The code for a single-learner is essentially the same.

```
$A$ and $B$ are two different learners.
$M_A^i$ and $M_B^i$ are models of $A$ and $B$ at step $i$.
$U$ is a pool of unlabelled examples.
$L$ is the manually labelled seed data.
**Initialize:**
    $M_A^0 \leftarrow Train(A, L^0)$
    $M_B^0 \leftarrow Train(B, L^0)$
**Loop:**
    $N \leftarrow$ Select $n$ examples using $M_A^i$ and $M_B^i$
        according to some selection method $S$
    $U \leftarrow U - N$
    $L^{i+1} \leftarrow L^i \cup Label(N)$
    $M_A^{i+1} \leftarrow Train(A, L^{i+1})$
    $M_B^{i+1} \leftarrow Train(B, L^{i+1})$
**Until:**
    $(U = \emptyset)$ or (human stops)
```

Figure 3: Pseudo-code for committee-based active learning.

are selected for annotation from a randomly chosen subset according to the operative selection method until the total amount of annotated training material made available to the learners reaches 3000. We select 25 examples at time until the training set contains 1000 examples, then 50 at a time until it has 2000, and finally 100 at a time until it has 3000. The results for each selection method are averaged over four tenfold cross-validation runs.

Whereas Hwa (Hwa, 2000) evaluated the effectiveness of selective sampling according to the number of brackets which were needed to create the parse trees for selected sentences, we compare selection methods based on the absolute number of sentences they select. This is realistic in the Redwoods setting since the derivation trees are created automatically from the ERG, and the task of the human annotator is to select the best from all licensed parses. Annotation in Redwoods uses an interface that presents local discriminants which disambiguate large portions of the parse forest, so options are narrowed down quickly even for sentences with a large number of parses.

7 Results

Figure 4 shows the performance of the LL-CONFIG model as more examples are chosen according to the different selection methods. As can be seen, both tree entropy and disagreement are equally effective and significantly improve on random selection.[4] Selection by sentence length is worse than random until 2100 examples have been annotated. Selecting more ambiguous sentences does eventually perform significantly better than random, but its accuracy does not rise nearly as steeply as tree entropy and disagreement selection. Table 4 shows the precise values for all methods using different amounts of annotated sentences. The accuracies for entropy and disagreement are statistically significant improvements over random selection. Using a pair-wise t-test, the values for 500, 1000, and 2000 are significant at 99% confidence, and those for 3000 are significant at 95% confidence.[5]

	500	1000	2000	3000
random	65.87	68.76	71.39	72.82
disagree	68.52	71.60	74.31	74.63
entropy	69.01	71.90	74.10	74.85
ambiguity	64.65	68.54	72.25	74.54
length	64.82	66.41	70.37	73.51

Table 4: Accuracy for different selection methods with different amounts of training data.

Table 5 shows that when compared to random selection using 3000 examples, tree entropy and disagreement achieve higher accuracy while reducing the number of training examples needed by more than one half. Though selection by ambiguity does provide a reduction over random selection, it does not enjoy the same rapid increase as tree entropy and disagreement, and it performs roughly equal to or worse than random until 1100 examples, as is evident in Figure 4(b).

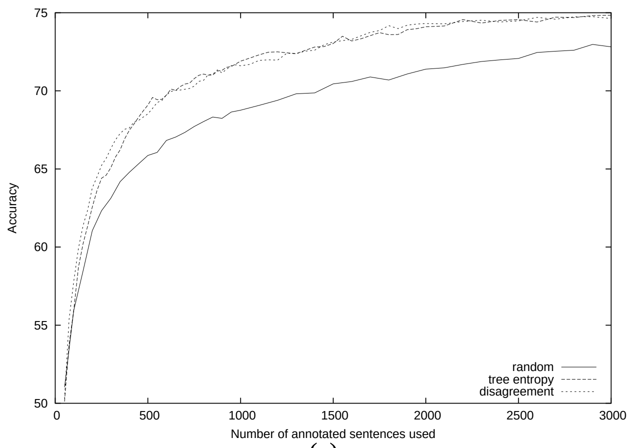
	# examples	avg. score	reduction
random	3000	72.82	N/A
disagree	1450	72.95	51.7
entropy	1450	72.84	51.7
ambiguity	2300	72.95	23.3
length	2600	73.70	12.0

Table 5: Number of examples needed for different selection methods to outperform random selection with 3000 examples. The final column gives the percentage reduction in the number of examples used.

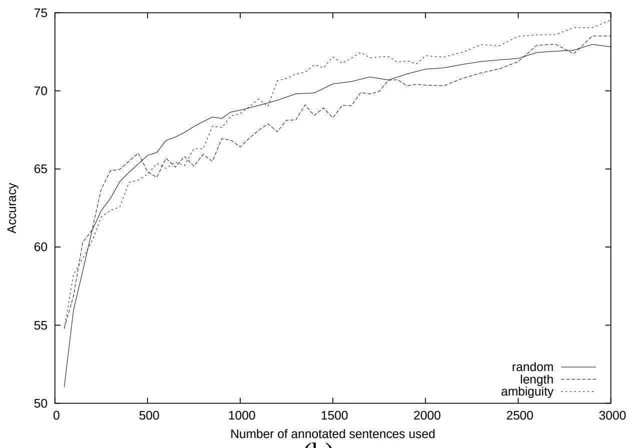
We also tested preferred parse disagreement by pairing LL-CONFIG with the perceptrons. The performance in these cases was nearly identical to that given for selection by disagreement in Figure 4, which used LL-CONFIG and LL-NGRAM for the committee. This indicates that differences either in terms of the algorithm or the feature set used are enough to bias the learners sufficiently for them to disagree on informative examples. This provides flexibility for applying selection by disagreement in different contexts where it may be easier to employ different

[4] LL-CONFIG was paired with LL-NGRAM for preferred parse disagreement in Figure 4(a).

[5] The slightly lower confi dence for 3000 examples indicates the fact that the small size of the corpus leaves the selection techniques with fewer informative examples to choose from and thereby differentiate itself from random selection.



(a)



(b)

Figure 4: Accuracy as more examples are selected according to (a) random, tree entropy, and disagreement, and (b) random, ambiguity, and sentence length.

feature sets than different algorithms, or vice versa. The fact that using the same feature set with different algorithms is effective for active learning is interesting and is echoed by similar findings for co-training (Goldman and Zhou, 2000).

Given the similar performance of tree entropy and preferred parse disagreement, it is interesting to see whether they select essentially the same examples. One case where they might not overlap is a distribution with two sharp spikes, which would be likely to provide excellent discriminating information. Though such a distribution has low entropy, each model might be biased toward a different spike and they would select the example by disagreement.

To test this, we ran a further experiment with a ***combined*** selection method that takes the intersection of tree entropy and disagreement. At each round, we randomly choose examples from the pool of unannotated sentences and sort them according to tree entropy, from highest to lowest. From the first 100 of these examples, we take the first n examples that are also selected by disagreement, varying the number selected in the same manner as for the previous experiments. When the size of the intersection is less than the number to be selected, we select the remainder according to tree entropy.

The performance for combined selection is compared with entropy and random selection in Figure 5 and Table 6. There is an slight, though not significant improvement over entropy on its own. The improvement over random is significant for all values, using a pair-wise t-test at 99% confidence. The combined approach requires 1200 examples on average to outperform random selection with 3000 examples, a 60.0% reduction that improves on either method on its own.

Tracking the examples chosen by tree entropy and disagreement at each round verifies that they do not select precisely the same examples. It thus appears that disagreement-based selection helps tease out examples that contain better discriminating information than other examples with higher entropy. This may in effect be approximating a more general method that could directly identify such examples.

	500	1000	2000	3000
random	65.87	68.76	71.39	72.82
entropy	69.01	71.90	74.10	74.85
combined	69.56	71.98	74.43	75.26

Table 6: Accuracy for random, tree entropy and combined selection selection with different amounts of training data.

The accuracy of LL-CONFIG when using all 4802 available training examples for the tenfold cross-validation is 74.80%, and combined selection improves on this by reaching 75.26% (on average) with 3000 training examples. Furthermore, though active learning was halted at 3000 examples, the accuracy for all the selection methods was still increasing at this point, and it is likely than even higher accuracy would be achieved by allowing more examples to be selected. Sample selection thus appears to identify highly informative subsets as well as reduce the number of examples needed.

Finally, we considered one further question regarding the behavior of sample selection under different conditions: can an impoverished model select informative examples for a more capable one? Thus, if active learning is actually used to extend a corpus, will the examples selected for annotation still be of high utility if we later devise a better feature selection strategy that gives rise

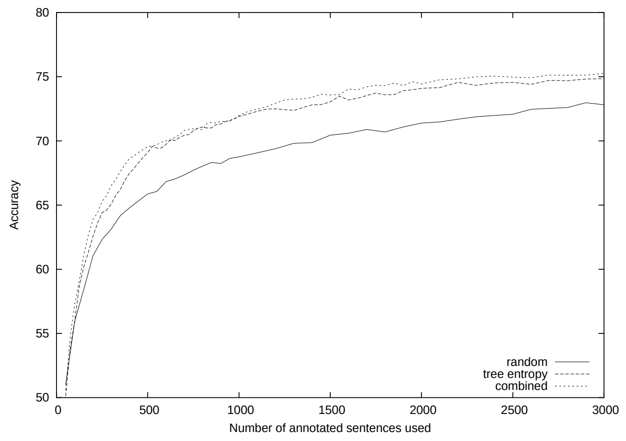


Figure 5: Accuracy as more examples are selected based on tree entropy alone and tree entropy combined with preferred parse disagreement.

to better models? To test this, we created a log-linear model that uses only bigrams, used it to select examples by tree entropy, and simultaneously trained and tested LL-CONFIG on those examples. Utilizing all training material, the bigram model performs much worse than LL-CONFIG overall: 71.43% versus 74.80%.

LL-CONFIG is thus a sort of passenger of the weaker bigram model, which drives the selection process. Figure 6 compares the accuracy of LL-CONFIG under this condition (which only involved one tenfold cross-validation run) with the accuracy when LL-CONFIG itself chooses examples according to tree entropy. Random selection is also included for reference.

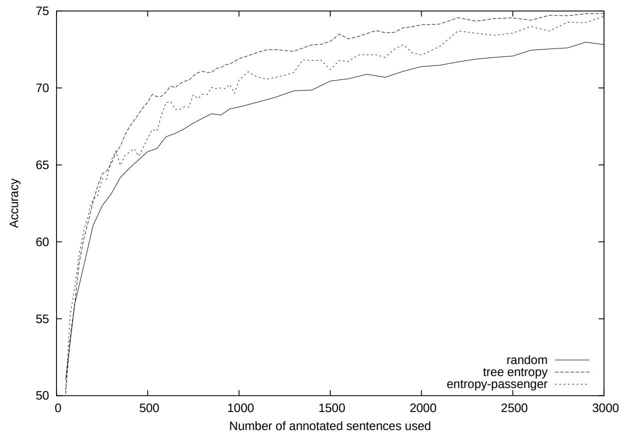


Figure 6: Accuracy as more examples are selected based tree entropy according to LL-CONFIG itself and when LL-CONFIG is the passenger of an impoverished model.

This experiment demonstrates that although accuracy does not rise as quickly as when LL-CONFIG itself selects examples, it is still significantly better than random (at 95% confidence) despite the bigram model's poorer performance. We can thus expect samples chosen by the current best model to be informative, though not necessarily optimal, for improved models in the future.

8 Conclusion

We have shown that sample selection according to both tree entropy and preferred parse disagreement significantly reduce the number of examples needed to train models for HPSG parse selection, when compared to several baseline selection metrics. Furthermore, performance improves further when these these two methods are combined, resulting in a 60% reduction in the amount of training material without any degradation in parse selection accuracy. Another interesting result is that, for this data set, higher accuracy is attainable by *not* using all of the available training material. We have also shown that an impoverished learner can effectively choose samples that are informative for a better model.

Because tree entropy requires only one learner, it is simpler and more efficient than preferred parse disagreement. However, it requires the learner to be probabilistic, and thus cannot be straightforwardly used with machine learning algorithms such as standard perceptrons and support vector machines.

A more important difference between tree entropy and disagreement is that the latter leads naturally to a combined approach using both active learning and co-training. Rather than comparing the two learners on whether they categorically select the same preferred parse on a number of examples, we can view active learning as the inverse of agreement-based co-training (Abney, 2002). We can then explore thresholds for which we can determine that certain examples need to be human annotated and others can be confidently machine labelled.

In future work, we will explore the effect of using further models that utilize the semantic information in Redwoods for sample selection, and we will apply active learning to both expand Redwoods and add discourse-level annotations.

Acknowledgements. We would like to thank Markus Becker, Chris Callison-Burch, Dan Flickinger, Alex Lascarides, Chris Manning, Stephan Oepen, Kristina Toutanova, and the anonymous reviewers. This work was supported by Edinburgh-Stanford Link R36763, ROSIE project.

References

Steven Abney. 2002. Bootstrapping. In *Proceedings of the 40th Annual Meeting of the ACL*, pages 360–367, Philadelphia, PA.

Shlomo Argamon-Engelson and Ido Dagan. 1999. Committee-based sample selection for probabilistic classifiers. *Journal of Artificial Intelligence Research*, 11:335–360.

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

Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures and the voted perceptron. In *Proceedings of the 40th Annual Meeting of the ACL*, pages 263–270, Philadelphia, Pennsylvania.

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.

Dan Flickinger. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering*, 6(1):15–28. Special Issue on Efficient Processing with HPSG.

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.

Sally Goldman and Yan Zhou. 2000. Enhancing supervised learning with unlabeled data. In *Proceedings of the 17th International Conference on Machine Learning*, Stanford, CA.

Rebecca Hwa. 2000. 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.

Mark Johnson, Stuart Geman, Stephen Cannon, Zhiyi Chi, and Stephan Riezler. 1999. Estimators for Stochastic "Unification-Based" Grammars. In *37th Annual Meeting of the ACL.*

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.

Robert Malouf. 2002. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the Sixth Workshop on Natural Language Learning*, pages 49–55, Taipei, Taiwan.

Ion Muslea, Steven Minton, and Craig Knoblock. 2000. Selective sampling with redundant views. In *Proceedings of National Conference on Artificial Intelligence (AAAI-2000)*, pages 621–626.

Stephan Oepen, Kristina Toutanova, Stuart Shieber, Christopher Manning, Dan Flickinger, and Thorsten Brants. 2002. The LinGO Redwoods Treebank: Motivation and preliminary applications. In *Proceedings of the 19th International Conference on Computational Linguistics*, Taipei, Taiwan.

Miles Osborne. 2000. Estimation of Stochastic Attribute-Value Grammars using an Informative Sample. In *The* 18^{th} *International Conference on Computational Linguistics*, Saarbrücken.

Cynthia A. Thompson, Mary Elaine Califf, and Raymond J. Mooney. 1999. Active learning for natural language parsing and information extraction. In *Proc. 16th International Conf. on Machine Learning*, pages 406–414. Morgan Kaufmann, San Francisco, CA.

Kristina Toutanova and Chris Manning. 2002. Feature selection for a rich HPSG grammar using decision trees. In *Proceedings of the 6th Conference on Natural Language Learning*, Taipei, Taiwan.