

# Improved Fully Unsupervised Parsing with Zoomed Learning

**Roi Reichart**

ICNC  
The Hebrew University  
roi@cs.huji.ac.il

**Ari Rappoport**

Institute of computer science  
The Hebrew University  
arir@cs.huji.ac.il

## Abstract

We introduce a novel training algorithm for unsupervised grammar induction, called *Zoomed Learning*. Given a training set  $T$  and a test set  $S$ , the goal of our algorithm is to identify subset pairs  $T_i, S_i$  of  $T$  and  $S$  such that when the unsupervised parser is trained on a training subset  $T_i$  its results on its paired test subset  $S_i$  are better than when it is trained on the entire training set  $T$ . A successful application of zoomed learning improves overall performance on the full test set  $S$ .

We study our algorithm's effect on the leading algorithm for the task of fully unsupervised parsing (Seginer, 2007) in three different English domains, WSJ, BROWN and GENIA, and show that it improves the parser F-score by up to 4.47%.

## 1 Introduction

Grammar induction is the task of learning grammatical structure from plain text without human supervision. The task is of great importance both for the understanding of human language acquisition and since its output can be used by NLP applications, avoiding the costly and error prone creation of manually annotated corpora. Many recent works have addressed the task (e.g. (Klein and Manning, 2004; Seginer, 2007; Cohen and Smith, 2009; Headen et al., 2009)) and its importance has increased due to the recent availability of huge corpora.

A basic challenge to this research direction is how to utilize training data in the best possible way. Klein and Manning (2004) report results for

their dependency model with valence (DMV) for unsupervised dependency parsing when it is trained and tested on the same corpus (both when sentence length restriction is imposed, such as for WSJ10, and when it is not, such as for the entire WSJ). Today's best unsupervised dependency parsers, which are rooted in this model, train on short sentences only: both Headen et al., (2009) and Cohen and Smith (2009) train on WSJ10 even when the test set includes longer sentences.

Recently, Spitkovsky et al., (2010) demonstrated that training the DMV model on sentences of up to 15 words length yields better results on the entire section 23 of WSJ (with no sentence length restriction) than training with the entire WSJ corpus.

In contrast to these dependency models, the Seginer constituency parser achieves its best performance when trained on the entire WSJ corpus either if sentence length restriction is imposed on the test corpus or not. The sentence length restriction training protocol of (Spitkovsky et al., 2010), harms this parser. When the parser is trained with the entire WSJ corpus its F-score performance on the WSJ10, WSJ20 and the entire WSJ corpora are 76, 64.8 and 56.7 respectively. When training is done with WSJ10 (WSJ20) performance degrades to 60 (72.2), 37.4 (61.9) and 29.7 (48) respectively.

In this paper we introduce the *Zoomed Learning* (ZL) technique for unsupervised parser training: given a training set  $T$  and a test set  $S$ , it identifies subset pairs  $T_i, S_i$  of  $T$  and  $S$  such that when the unsupervised parser is trained on a training subset  $T_i$  its results on its paired test subset  $S_i$  are better than when it is trained on the entire training set  $T$ . A

successful application of zoomed learning improves performance on the full test set  $S$ .

We describe ZL algorithms of increasing sophistication. In the simplest algorithm the subsets are randomly selected while in the more sophisticated versions subset selection is done using a fully unsupervised measure of constituency parse tree quality.

We apply ZL to the Seginer parser, the best algorithm for fully unsupervised constituency parsing. The input is a plain text corpus without any annotation, not even POS tagging<sup>1</sup>, and the output is an unlabeled bracketing for each sentence.

We experiment in three different English domains: WSJ (economic newspaper), GENIA (biological articles) and BROWN (heterogeneous domains), and show that ZL improves the parser F-score by as much as 4.47%.

## 2 Related Work

Unsupervised parsing has attracted researchers for over a quarter of a century (see (Clark, 2001; Klein, 2005) for reviews). In recent years efforts have been made to evaluate the algorithms on manually annotated corpora such as the WSJ PennTreebank. Recent works on unlabeled bracketing or dependencies induction include (Klein and Manning, 2002; Klein and Manning, 2004; Dennis, 2005; Bod, 2006a; Bod, 2006b; Bod, 2007; Smith and Eisner, 2006; Seginer, 2007; Cohen et al., 2008; Cohen and Smith, 2009; Headden et al., 2009). Most of the works above use POS tag sequences, created either manually or by a supervised algorithm, as input. The only exception is Seginer’s parser, which induces bracketing from plain text.

Our confidence-based ZL algorithms use the PUPA unsupervised parsing quality score (Reichart and Rappoport, 2009b). As far as we know, PUPA is the only unsupervised quality assessment algorithm for syntactic parsers that has been proposed. Combining PUPA with Seginer’s parser thus preserves the fully unsupervised nature of the task.

Quality assessment of a learning algorithm’s output has been addressed for supervised algorithms

(see (Caruana and Niculescu-Mizil, 2006) for a survey) and specifically for supervised syntactic parsing (Yates et al., 2006; Reichart and Rappoport, 2007; Ravi et al., 2008; Kawahara and Uchimoto, 2008). All these algorithms are based on manually annotated data and thus do not preserve the unsupervised nature of the task addressed in this paper.

We experiment with the Seginer parser for two reasons. First, this is the best algorithm for the task of fully unsupervised parsing which motivates us to improve its performance. Second, this is the only publicly available unsupervised parser that induces constituency trees. The PUPA score we use in our confidence-based algorithms is applicable for constituency trees only. When additional constituency parsers will be made available, we will test ZL with them as well. Interestingly, the results reported for other constituency models (the CCM model (Klein and Manning, 2002) and the U-DOP model (Bod, 2006a; Bod, 2006b)) are reported when the parser is trained on its test corpus even if the sentences in that corpus are of bounded length (e.g. WSJ10). This raises the question if using more training data (e.g. the entire WSJ) wisely can enhance these models.

Recently, Spitkovsky et al., (2010) proposed three approaches for improvement of unsupervised grammar induction by considering the complexity of the training data. The approaches have been applied to the DMV unsupervised dependency parser (Klein and Manning, 2004) and improved its performance. One of these approaches is to train the model with sentences whose length is up to 15 words. As noted above, such a training protocol fails to improve the performance of the Seginer parser.

The other approaches in that paper, bootstrapping via iterated learning of increasingly longer sentences and a combination of the bootstrapping and the short sentences approaches, are not directly applicable to the Seginer parser since its training method cannot be trivially bootstrapped with parses created in former steps (Seginer, 2007).

**Related machine learning methods.** ZL is related to ensemble methods. Both ZL and such methods produce multiple learners, each of them trained on a different subset of the training data, and decide which learner to use for a particular test instance. Bagging (Breiman, 1996) and boosting (Freund and Schapire, 1996), where the experts utilize the same

---

<sup>1</sup>For clarity of exposition, we still refer to this corpus as our *training corpus*. In the algorithms presented in this paper, the test set is included in the training set which is a common practice in unsupervised parsing.

learning algorithm and differ in the sample of the training data they use for its training, were applied to supervised parsing (Henderson and Brill, 2000; Becker and Osborne, 2005). In Section 3 we discuss the connection of ZL to boosting.

Owing to the fact that ZL produces different learners, it is natural to use it in conjunction with an ensemble method, which is what we do in this paper with our EZL model (Section 3).

ZL is also related to active learning (AL) (Cohn and Ladner, 1994). AL also uses training subset selection, with the goal of obtaining a faster learning curve for an algorithm. AL is done in supervised settings, usually in order to minimize human annotation costs. AL algorithms providing faster learning than random subset selection for parsing have been proposed (Reichart and Rappoport, 2009a; Hwa, 2004). However, we are not aware of AL applications in which the *overall* performance on the test set has been improved. In addition, our application here is to an unsupervised problem.

Algorithms that utilize unsupervised clustering for class decomposition in order to improve classifiers’ performance (e.g. (Vilalta and Rish, 2003)) are related to ZL. In such methods, examples that belong to the same class are clustered, and the induced clusters are considered as separate classes. These methods, however, have been applied only to supervised classification in contrast to our work that addresses unsupervised structured learning. Moreover, after class decomposition a classifier is trained with the entire training data while the subsets identified by a ZL algorithm are parsed by a parser trained only with the sentences they contain.

### 3 Zoomed Learning Algorithms

Zoomed Learning proposes that performance on a particular test instance might improve if training is done on a proper *subset* of the training set. The ZL view is clearly applicable when the training data is comprised of subsets originating from different sources having different natures. If the test data is also similarly composed, performance on any particular test instance might improve if training is done on a training subset coming from the same source. However, even when the training and test data are from the same source, a ZL algorithm may capture

fine differences between subsets.

The ZL idea is therefore related to the notions of *in-domain* and *out-of-domain* (domain adaptation). In the former, the training and test data are assumed to originate from the same domain. In the latter, the test data comes from a different domain, and therefore has different statistics from the training data. Indeed, the performance of NLP algorithms in domain adaptation scenarios is markedly lower than in in-domain ones (McClosky et al., 2006).

ZL takes this observation to the extreme, assuming that a similar situation might exist *even in in-domain scenarios*. After all, a ‘domain’ is only a coarse qualification of the nature of a data set. In NLP, a domain is usually specified as the genre of the text involved (e.g., ‘newspapers’). However, there are additional axes that might influence the statistics obtained from training data, e.g., the syntactic nature of sentences.

This section presents our ZL algorithms. We start with the simplest possible ZL algorithm where the subsets are randomly selected. We then describe ZL algorithms based on quality-based parse selection. We first detail a basic version and then an extended version consisting of another level of parse selection. Finally, we briefly discuss the PUPA quality measure that we use to evaluate the quality of a parse tree.

In all versions of the algorithm the input consists of a set  $T$  of  $N$  training sentences, a set  $S \subseteq T$  of test sentences, and an integer number  $N_H \leq N$ .

**Zoomed Learning with Random Selection (RZL).** The simplest ZL algorithm randomly assigns each of the training sentences to one of  $n$  sets ( $n = 2$  in this paper). More explicitly, the set number is drawn from a uniform distribution on  $\{1, 2, \dots, n\}$ . Each set is then parsed by a parser that is trained only with the sentences contained in that set.

The intuition behind this algorithm is that different sets of sentences are likely to manifest different syntactic patterns. Consequently, the best way to learn the syntactic patterns of any given set of sentences might be to train the parser on the sentences contained in the set.

While simple, in Section 5 it is shown to improve the performance of the Seginer parser.

**The Basic Quality-Based Algorithm (BZL).** The idea of the basic ZL algorithm is that sentences for

which the parser provides low quality parses manifest different syntactic patterns than the sentences for which the parser provides high quality parses. The main challenge is therefore to estimate the quality of the produced parses without supervision.

The algorithm has three stages. In the first, we create the fully-trained model by training the parser using all of the  $N$  sentences of  $T$ . We then parse these  $N$  sentences using the fully-trained model.

In the second, we compute a parse confidence score for each of the  $N$  sentences, based on the  $N$  parses produced in the first stage. We divide the training sentences to two subsets: a high quality subset  $H$  consisting of the top scored  $N_H$  sentences, and a lower quality subset  $L$  consisting of the other  $N_L = N - N_H$  sentences.

As is common practice for this problem (Klein and Manning, 2004; Seginer, 2007), the test set is contained in the training set. This methodology is a valid one because the training set is unannotated. Our test set is thus naturally divided into two subsets, a high quality subset  $HT$  consisting of the test set sentences contained in  $H$  and a lower quality subset  $LT$  consisting of the test set sentences contained in  $L$ .

In the third stage, each of the test subsets is parsed by a model trained only on its corresponding training subset. This stage is motivated by our assumption that the high and low quality subsets manifest dissimilar syntactic patterns, and consequently the statistics of the parser’s parameters suitable for one subset differ from those suitable for another.

We compute the confidence score in the second stage using the unsupervised PUPA algorithm (Reichart and Rappoport, 2009b). POS tags for it are induced using the fully unsupervised algorithm of Clark (2003). The parser we experiment with is the incremental parser of Seginer (2007), whose input consists of raw sentences and does not include any kind of supervised POS tags (created either manually or by a supervised algorithm). Consequently, our algorithm is fully unsupervised. The only parameter it has is  $N_H$  but ZL improves parser performance for most  $N_H$  values.

BZL is related to boosting. In boosting after training one member of the ensemble, examples are re-weighted such that examples that are classified correctly are down-weighted. BZL does something sim-

ilar: it uses PUPA to estimate which sentences are given high quality parse trees, and down-weights examples with high (low) PUPA score to 0 when training the  $L$ -trained ( $H$ -trained) model. However, in boosting the entire test set is annotated by the same learning model, while ZL parses each test subset with a model trained on its corresponding training subset.

### **The Extended Quality-Based Algorithm (EZL).**

The basic algorithm produces an ensemble of two parsing experts: the one trained on  $H$  and the one trained on  $L$ . It uses the ensemble to parse the test set by applying the  $H$ -trained expert to  $HT$  and the  $L$ -trained expert to  $LT$ . Naturally, there are other ways to utilize the ensemble to parse the test set. In addition, even if parse trees generated by the experts are better with high probability than those of the fully trained parser, they are not guaranteed to be so. The fully trained parser is therefore also a valuable member in the ensemble. Consequently, we introduce an extended zoomed learning algorithm (EZL).

The extended version is implemented as a final fourth stage of the previously described basic algorithm. In this stage, the two test subsets are parsed by the fully trained parsing model, in addition to being parsed by the zooming parsing models. We now have two parses for each test sentence  $s$ :  $P_Z(s)$ , the parse created by a parser trained with the sentences contained in its corresponding training subset, and  $P_F(s)$ , created by the fully trained parser.

For each of the two parses of each test sentence, a confidence score is computed by PUPA. As will be reviewed below, PUPA uses a *set* of parsed sentences to compute the statistics on which its scores are based. Therefore, there are two sources for a difference between the scores of the two parse trees of a given test sentence: the difference between the trees themselves, and the difference between the parses of the other sentences in the set.

The PUPA score for  $P_Z(s)$  is computed using the parses created for the sentences contained in the test subset of  $s$  by a parser trained with the corresponding training subset. The PUPA score for  $P_F(s)$  is computed using the parses created for the entire test set by the fully trained parser.

The algorithm now outputs a final parse by selecting for each sentence the parse tree having the higher PUPA score.

**The PUPA Confidence Score.** In the second and fourth stages of the confidence-based algorithms, an unsupervised confidence score is computed for each of the induced parse trees. The confidence score algorithm we use is the POS-based Unsupervised Parse Assessment (PUPA) algorithm (Reichart and Rappoport, 2009b). We provide here a brief description of this algorithm.

The input to PUPA is a set  $I$  of parsed sentences, and its output consists of a confidence score in  $[0, 1]$  assigned to each sentence in  $I$ .

The PUPA algorithm collects statistics of the syntactic structures (parse tree constituents) contained in the set  $I$  of parsed sentences. The constituent representation is based on the POS tags of the words in the yield of the constituent and of the words in the yields of neighboring constituents. We follow Reichart and Rappoport (2009b) and induce the POS tags using the fully unsupervised POS induction algorithm of Clark (2003).

The algorithm then goes over each individual tree in the set  $I$  and scores it according to the collected statistics. The PUPA algorithm is guided by the idea that syntactic structures that are frequently created by the parser are more likely to be correct than structures the parser produces less frequently. Therefore, constituents that are more frequent in the set  $I$  receive higher scores after proper regularization is applied to prevent potential biases. The tree score is a combination of the scores of its constituents.

Full details of the PUPA algorithm are given in (Reichart and Rappoport, 2009b). The resulting score was shown to be strongly correlated with the extrinsic quality of the parse tree, defined to be its F-score similarity to the manually created (gold standard) parse tree of the sentence.

## 4 Experimental Setup

We experimented with three English corpora: the WSJ Penn Treebank (Marcus et al., 1993) consisting of economic newspaper texts, the BROWN corpus (Francis and Kucera, 1979) consisting of texts of various English genres (e.g. fiction, humor, romance, mystery and adventure) and the GENIA corpus (Kim et al., 2003) consisting of abstracts of scientific articles from the biological domain. All corpora were stripped of all annotation (bracketing and

POS tags).

For all corpora we report the parser performance on the entire corpus (WSJ: 49206 sentences, BROWN: 24243 sentences, GENIA: 4661 sentences). For WSJ we also provide an analysis of the performance of the parser when applied to sentences of bounded length. These sub-corpora are WSJ10 (7422 sentences), WSJ20 (25522 sentences) and WSJ40 (47513 sentences) where WSJ $Y$  denotes the subset of WSJ containing sentences of length at most  $Y$  (excluding punctuation).

Seginer’s parser achieves its best reported results when trained on the full WSJ corpus. Consequently, for all corpora, we compare the performance of the parser when trained with the ZL algorithms to its performance when trained with the full corpus.

The POS tags required as input by the PUPA algorithm are induced by the fully unsupervised POS induction algorithm of Clark (2003)<sup>2</sup>. Reichart and Rappoport (2009b) demonstrated an unsupervised technique for the estimation of the number of induced POS tags with which the correlation between PUPA’s score and the parse F-score is maximized. When exploring an experimental setup identical to our WSJ setup, they set the number of induced tags to be 5. We therefore induced 5 POS tags for each corpus, using all its sentences as input for Clark’s algorithm. Our implementation of the PUPA algorithm will be made available on line.

For each corpus we performed  $K$  experiments with each of the three ZL algorithms, where  $K$  equals to the number of sentences in the corpus divided by 1000 (rounded upwards). In each experiment the size of the high quality  $H$  and lower quality  $L$  training subsets is different.  $H$  consists of the  $N_H$  top ranked sentences according to PUPA (or  $N_H$  randomly selected sentences for RZL), with  $N_H$  changing from 1000 upwards in steps of 1000.  $L$  consists of the rest of the sentences in the training corpus (WSJ). The results reported for RZL are averaged over 10 runs.

We report the parser performance on the test corpus for each training protocol. Following the unsupervised parsing literature multiple brackets and brackets covering a single word are not counted, but the sentence level bracket is. We exclude punctua-

---

<sup>2</sup>[www.cs.rhul.ac.uk/home/alexc/RHUL/Downloads.html](http://www.cs.rhul.ac.uk/home/alexc/RHUL/Downloads.html)

	WSJ10, F(Full) = 76			WSJ20, F(Full) = 64.82			WSJ40, F(Full) = 57.54			WSJ, F(Full) = 56.7		
$N_H$	25%	50%	75%	25%	50%	75%	25%	50%	75%	25%	50%	75%
EZL	<b>76.38</b> <b>+0.38</b>	<b>76.80</b> <b>+0.80</b>	<b>76.14</b> <b>+0.14</b>	<b>65.75</b> <b>+0.93</b>	<b>66.14</b> <b>+1.30</b>	<b>65.66</b> <b>+0.82</b>	<b>58.32</b> <b>+0.78</b>	<b>58.75</b> <b>+1.21</b>	<b>58.56</b> <b>+1.02</b>	<b>57.47</b> <b>+0.77</b>	<b>57.90</b> <b>+1.20</b>	<b>57.73</b> <b>+1.13</b>
BZL	75.07 -0.93	75.78 -0.22	75.02 -0.98	65.08 +0.26	65.74 +0.92	64.79 -0.03	58.13 +0.59	58.70 +1.16	58.21 +0.67	57.30 +0.60	57.88 +1.18	57.66 +1.06
RZL	75.41 -0.59	75.00 -1.00	75.32 -0.68	64.43 -0.39	64.66 -0.16	65.32 +0.50	57.27 -0.27	57.63 +0.09	58.39 +0.85	56.44 -0.26	56.84 +0.14	57.59 +0.89

	WSJ10			WSJ20			WSJ40			WSJ		
$ LT $	10%	20%	30%	10%	20%	30%	10%	20%	30%	10%	20%	30%
EZL	<b>1.32</b>	<b>0.95</b>	<b>0.61</b>	<b>2.98</b>	<b>3.13</b>	<b>1.76</b>	<b>2.60</b>	<b>2.80</b>	2.62	<b>2.44</b>	2.40	2.50
BZL	0.37	0.80	0.53	2.38	3.12	1.23	2.34	3.20	<b>3.35</b>	2.28	<b>2.50</b>	<b>3.23</b>
RZL	-2.10	-1.88	-1.20	-0.91	-0.50	0.72	0.30	0.35	1.50	0.34	0.50	1.60

Table 1: Performance of the EZL, BZL and RZL algorithms in the WSJ experiments (results for BROWN and GENIA are shown in Table 2). Results are presented for four test corpora WSJ10, WSJ20, WSJ40 and the entire WSJ. **Top table:** Results for various values of  $N_H$  (the number of sentences in the high quality training subset). Evaluation is performed for all sentences in the test corpora. For each algorithm, the top line is its F-score performance and the bottom line is the difference from the F-score of the fully-trained Seginer parser (denoted by F(Full)). The EZL algorithm is superior. **Bottom table:** Results for various lower quality test subsets. Presented are the differences from the F-score of the fully-trained Seginer parser. The test subsets selected by different algorithms for a specific  $N_H$  value are not necessarily identical and for the sub-corpora they are not necessarily of identical size. Reported are the improvements for the  $LT$ 's of smallest size which is over 10%, 20%, and 30% of the test corpus (the top table reports results for the entire test set, which is why we can report F-scores there). The  $LT$  set size is denoted with  $|LT|$ .

tion and null elements as in (Klein, 2005). To evaluate the quality of a parse tree with respect to its gold standard, the unlabeled parsing F-score is used.

## 5 Results

**Entire Corpus Results.** We start by discussing the effect of ZL on the performance of the Seginer parser when no length restriction is imposed on the test corpus sentences (WSJ, BROWN and GENIA).

Table 1 (top, right section, for WSJ), Figure 1 (top line, right graph, for WSJ), and Table 2 (the left section of each table, top table for BROWN and bottom table for GENIA) present the difference between the F-score performance of the Seginer parser when trained with the ZL algorithms and the parser's performance when trained with the entire corpus.

For all test corpora and sizes of the high quality training subset ( $N_H$ ), zoomed learning improves the parser performance. ZL improves the parser performance by 1.13% (WSJ), 1.46% (BROWN, the number does not appear in the table) and 4.47% (GENIA).

For WSJ, the most substantial improvement is pro-

vided by EZL, while for BROWN and GENIA the best results for some  $N_H$  values are achieved by BZL and for others by EZL (and for GENIA with small  $N_H$  values even by RZL).

Note, that for all three corpora zoomed learning with random selection (RZL) improves the parser performance on the entire test corpus, although to a lesser extent than confidence-based ZL. This is true for almost all  $N_H$  values, including those that do not appear in the tables. See Figure 1 (top line, right-most graph) for WSJ.

We follow the unsupervised parsing literature and provide performance analysis for WSJ sentences of bounded length (WSJ10, WSJ20 and WSJ40). To prevent clutter, for BROWN and GENIA we report only entire corpus results.

Table 1 (top, left three sections) and Figure 1 (top line, three leftmost graphs) present results for WSJ10, WSJ20 and WSJ40.

The result patterns for the sub-corpora are similar to those reported for the entire WSJ corpus. EZL and BZL both improve over the fully-trained parser, and

BROWN	ENTIRE CORPUS (F = 57.19)				LT				HT			
$N_H$	10%	30%	50%	70%	10%	30%	50%	70%	10%	30%	50%	70%
EZL	0.55	0.69	<b>0.64</b>	<b>0.66</b>	0.65	0.82	<b>1.15</b>	<b>1.31</b>	-1.44	-0.03	0.04	<b>0.30</b>
BZL	<b>1.11</b>	<b>0.80</b>	0.02	-0.10	<b>1.42</b>	<b>1.20</b>	0.76	0.51	-4.80	-1.30	-0.79	-0.37
RZL	0.257	0.755	0.49	0.24	0.23	0.75	0.60	0.53	<b>0.44</b>	<b>0.76</b>	<b>0.42</b>	0.12

GENIA	ENTIRE CORPUS (F = 42.71)				LT				HT			
$N_H$	10%	30%	50%	70%	10%	30%	50%	70%	10%	30%	50%	70%
EZL	0.01	0.83	1.10	1.66	-0.01	0.76	0.80	3.37	0.34	1.00	1.40	1.55
BZL	-0.46	1.40	<b>2.74</b>	<b>4.47</b>	-0.54	0.40	0.96	<b>4.09</b>	0.42	<b>4.29</b>	<b>4.60</b>	<b>5.49</b>
RZL	<b>0.61</b>	<b>1.70</b>	2.09	1.99	<b>0.28</b>	<b>2.08</b>	<b>3.30</b>	3.86	<b>3.04</b>	3.22	2.80	1.85

Table 2: Results for the BROWN (top table) and GENIA (bottom table) corpora. Results are presented for the entire corpus (left column section), the low quality test subset (middle column section, *LT*) and the high quality test subset (right column section, *HT*) of each corpus, as a function of the high quality training set size ( $N_H$ ). Since the tables present entire corpus results, the training and test subsets are identical.

the improvement of the former is more substantial.

**Baselines.** A key principle of ZL is the selection of subsets that are better parsed by a parser trained only with the sentences they contain than with a parser trained with the entire training corpus. To verify the importance of this principle we considered two alternative training protocols.

In the first, the entire test corpus is parsed with a parser that was trained with a subset of randomly selected sentences from the training set. We run this protocol for all three corpora (and for the WSJ sub-corpora) with various training set sizes and obtained substantial degradation in the parser performance. The performance monotonically increases with the training set size and reached its maximum when the entire corpus is used. We conclude that using less training material harms the parser performance if a test subset is not carefully selected.

The second protocol is the ‘less is more’ protocol of Spitkovsky et al., (2010) in which we parsed each test corpus using a parser that was trained with all training sentences of a bounded length. Unlike in their paper, in which this protocol improves the performance of the DMV unsupervised dependency parser (Klein and Manning, 2004), for the Seginer parser the protocol harms the results. When parsing the entire WSJ with a WSJ10-trained parser or with a WSJ20-trained parser, the F-score results are 59.99% and 72.22% compared to 76.00% of the fully-trained parser. For GENIA the numbers are 15.61 and 35.87 compared to 42.71 and for BROWN

they are 36.05 and 50.02 compared to 57.19<sup>3</sup>.

It is also interesting that sentence length is generally not a good subset selection criterion for ZL. When parsing WSJ10 with a WSJ10-trained parser, F-score results are 59.29 while the F-score of the fully-trained parser on this corpus is 76.00. The same phenomenon is observed with WSJ20 (F-score of 61.90 with WSJ20 training and of 64.82 with the entire WSJ training), and for the BROWN corpus (65.01 vs. 69.43 for BROWN10 and 61.90 vs 62.92 for BROWN20). For GENIA, however, while parsing GENIA10 with a GENIA10-trained parser harms the performance (45.28 vs. 60.23), parsing GENIA20 with a GENIA20-trained parser enhances the performance (53.23 vs. 50.00).

These results emphasize the power of random selection for ZL as random selection does provide a good selection criterion.

**LT vs. HT.** In what follows we analyze the ZL algorithms aiming to characterize their strengths and weaknesses.

Table 1 (bottom), the middle and right sections of Table 2 (both tables) and Figure 1 (second and third lines) present the performance of the ZL algorithms on the lower quality and higher quality test subsets (*LT* and *HT*). The results patterns for WSJ and BROWN are different than those of GENIA.

For WSJ (and its sub-corpora) and BROWN,

<sup>3</sup>We repeated this protocol multiple times for each corpus, training the parser with sentences of length 5 to 45 in steps of 5. In all cases we observed performance degradation compared to the fully-trained parser.

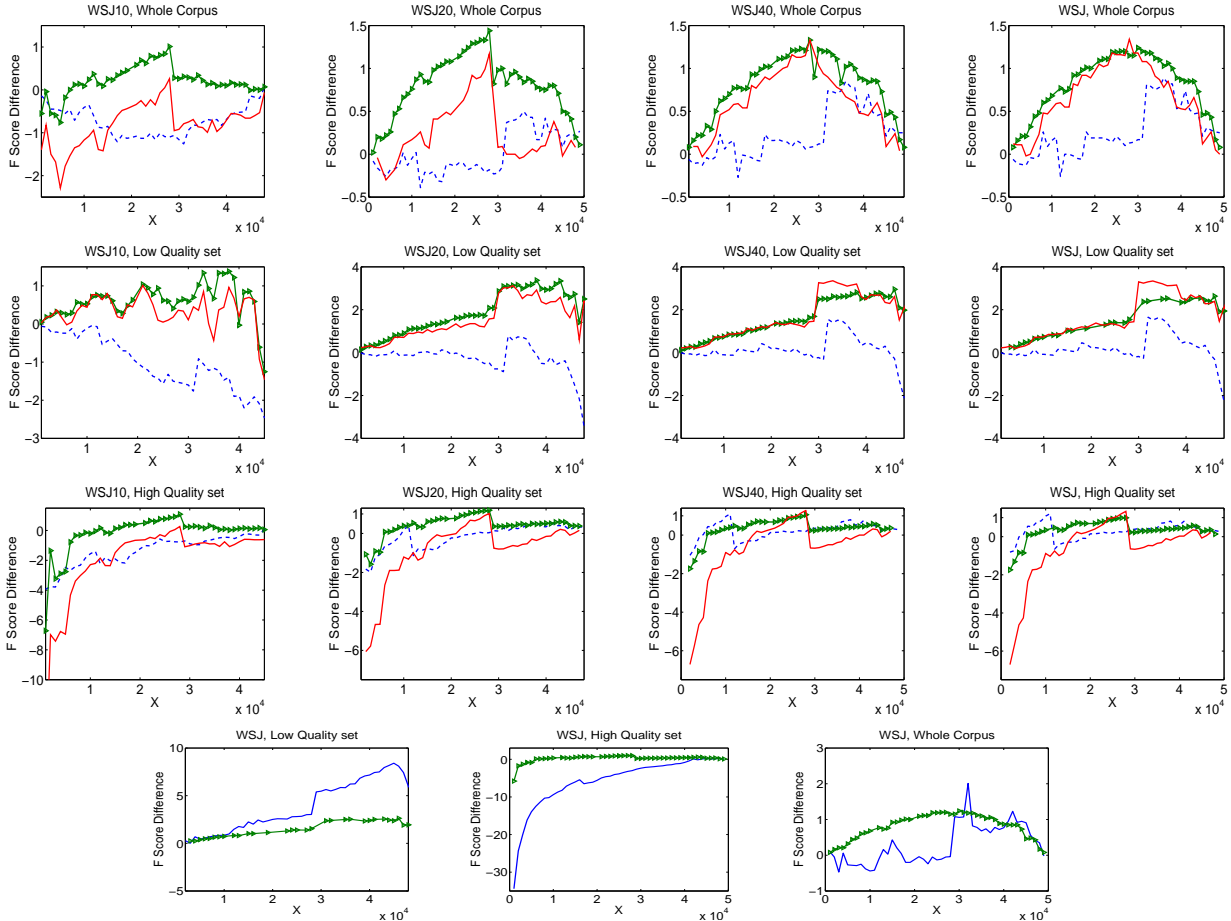


Figure 1: WSJ results. **Top Three Lines:** Difference in F-score performance of the Seginer parser between training with ZL and training with the entire WSJ corpus. Results are presented for the entire corpus (top line), the lower quality test subset (*LT*, middle line) and the higher quality test subset (*HT*, bottom line) as a function of the size of the high quality training subset  $X = N_H$ , measured in sentences. The curve with triangles is for the extended zoomed learning algorithm (EZL), the solid curve is for the basic zoomed learning algorithm (BZL) and the dashed curve is for zoomed learning with random selection (RZL). **Bottom line:** Comparison between the performance of the Seginer parser with the EZL algorithm (curves with triangles) and when subset selection is performed using the oracle F-score of the trees (solid curves). F-score differences from the performance of the fully trained parser are presented for the WSJ test corpus as a function of  $N_H$ , the high quality training subset size. Oracle selection is superior for the lower quality subset but inferior for the high quality subset.

confidence-based ZL (BZL and EZL) provides a substantial improvement for *LT*. For WSJ, F-score improvement is up to 1.32% (WSJ10), 3.13% (WSJ20), 3.35% (WSJ40) and 3.23% (the entire WSJ). For BROWN the improvement is up to 1.42%.

For *HT*, confidence-based ZL is less effective when these corpora are considered. As indicated in the third line of Figure 1, for WSJ and its subcorpora, EZL leads to a small improvement on *HT*, while BZL generally leads to a performance degradation on this test subset. For BROWN (the right section of Table 2 (top)), confidence-based ZL gener-

ally leads to a performance degradation on *HT*.

For GENIA, EZL and BZL improve the parser performance on both *LT* and *HT* for most  $N_H$  values. Understanding this difference is a subject for future research. Our initial hypothesis is that due to the relative small size of the GENIA corpus (4661 sentences compared 24243 and 49206 sentences of WSJ and BROWN respectively), there is more room for improvement in the parser performance on this corpus, and consequently ZL improves on both sets.

**Oracle Analysis.** Confidence-based ZL is based on the idea that sentences for which the fully-trained

parser provides parses of similar quality manifest similar syntactic patterns. Consequently, the parser performance on a set of such sentences can be improved if it is trained only with the sentences contained in the set. An oracle experiment, where selection is based on the F-score computed using the gold standard tree instead of on the PUPA score, can shed light on the validity of this idea.

Figure 1 (bottom line) compares the performance of EZL with that of the oracle-based zoomed learning algorithm when the test corpus is the entire WSJ. For the low quality test subset, oracle selection is dramatically better than confidence-based selection. For the high quality test subset the opposite pattern holds, that is, EZL is superior. These differences lead to the entire corpus pattern where EZL is superior for most  $N_H$  values.

Oracle-based and confidence-based zoomed learning demonstrate the same trend: they improve over the baseline for *LT* much more than for *HT*. For *HT*, oracle-based ZL even harms results and so does BZL, which does not benefit from the averaging effect of EZL. The magnitude of the effect of oracle-based zoomed learning is much stronger. These results support our idea that training the parser on a set selected by a well-designed confidence test leads to improvement of the parser performance for the selected sentences when the fully-trained parser produces parses of mediocre quality for them.

Integration of the experimental results for zoomed learning with the three selection methods: random, confidence-based and oracle-based leads to an important conclusion that should guide future research. The more accurate the confidence score used by the zoomed learning algorithm, the more substantial is the performance improvement for the low quality test subset, at the cost of more substantial degradation in the performance on the high quality subset (but recall the different GENIA pattern which should be further explored).

**EZL Variants.** For confidence-based ZL we explored two methods for utilizing the ensemble members for generating a final parse tree for each of the test sentences. In BZL, the *L*-trained parser and the *H*-trained parser generate parse trees for *LT* and *HT* sentences respectively. In EZL, for each sentence the final parse is selected between the parse

created by a parser trained with the sentences contained in its corresponding training subset, and the parse created by the fully trained parser.

There are other ways to use the ensemble members. While for all corpora it is beneficial to use the *L*-trained parser for the low quality test subset (*LT*), the results for WSJ and BROWN imply that it might be better to use the fully-trained parser or the EZL algorithm to parse the high quality test subset (*HT*). We have experimented with these methods and got only a minor improvement over the results reported here (improvement is more substantial for BROWN than for WSJ but does not exceed 0.5% for both). This can also be inferred from the relative minor performance degradation of BZL and EZL on *HT*.

We also explored a ZL scenario in which the entire test set is parsed either by the *H*-trained parser or by the *L*-trained parser. These protocols result in substantial degradation in parser performance (compared to the fully-trained parser) since the performance of the *H*-trained parser on *LT* and the performance of the *L*-trained parser on *HT* are poor.

## 6 Conclusions

We introduced zoomed learning – a training algorithm for unsupervised parsers. We applied three variants of ZL to the best fully unsupervised parsing algorithm (Seginer, 2007) and show an improvement of up to 4.47% in three English domains: WSJ, BROWN and GENIA.

Future research should focus on the development of more accurate estimators of parser output quality, and experimentation with different corpora, languages and parsers.

Developing a quality assessment algorithm for dependency trees will allow us to apply confidence-based ZL to unsupervised dependency parsing. Particularly, it will enable us to explore the combination of the methods proposed in (Spitkovsky et al., 2010) with ZL for the DMV model and to integrate the PUPA score into their bootstrapping algorithm.

Another direction is to apply ZL to other NLP tasks and ML areas, supervised and unsupervised.

## References

- Markus Becker and Miles Osborne, 2005. A two-stage method for active learning of statistical grammars. *IJ-CAI '05*.
- Rens Bod, 2006a. An all-subtrees approach to unsupervised parsing. *ACL-COLING '06*.
- Rens Bod, 2006b. Unsupervised parsing with U-DOP. *CoNLL '06*.
- Rens Bod, 2007. Is the end of supervised parsing in sight? *ACL '07*.
- Leo Breiman, 1996. Bagging predictors. *Machine Learning*, 24(2):123–140.
- Rich Caruana and Alexandru Niculescu-Mizil, 2006. An empirical comparison of supervised learning algorithms. *ICML '06*.
- Alexander Clark, 2001. *Unsupervised language acquisition: theory and practice*. Ph.D. thesis, University of Sussex.
- Alexander Clark, 2003. Combining distributional and morphological information for part of speech induction. *EACL '03*.
- Shay Cohen, Kevin Gimpel and Noah Smith, 2008. Logistic normal priors for unsupervised probabilistic grammar induction. *NIPS '08*.
- Shay Cohen and Noah Smith, 2009. Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. *NAACL '09*.
- David Cohn, Les Atlas and Richard Ladner. 1994. Improving generalization with active learning. *Machine Learning*, 15(2):201–221.
- Simon Dennis, 2005. An exemplar-based approach to unsupervised parsing. *CogSci '05*.
- W. N. Francis and H. Kucera 1979. Manual of information to accompany a standard corpus of present-day edited American English, for use with digital computers. *Department of Linguistics, Brown University Press, Providence, RI*.
- Yoav Freund and Robert E. Schapire, 1996. Experiments with a new boosting algorithm. *ICML '96*.
- William Headden III, Mark Johnson and David McClosky, 2009. Improving unsupervised dependency parsing with richer contexts and smoothing. *NAACL '09*.
- John Henderson and Eric Brill, 2000. Bagging and boosting a treebank parser. *NAACL '00*.
- Rebecca Hwa. 2004. Sample selection for statistical parsing. *Computational Linguistics*, 30(3):253–276.
- Daisuke Kawahara and Kiyotaka Uchimoto 2008. Learning reliability of parses for domain adaptation of dependency parsing. *IJCNLP '08*.
- Dan Klein and Christopher Manning, 2002. A generative constituent-context model for improved grammar induction. *ACL '02*.
- Dan Klein and Christopher Manning, 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. *ACL '04*.
- Dan Klein, 2005. *The unsupervised learning of natural language structure*. Ph.D. thesis, Stanford University.
- Jin-Dong Kim, Tomoko Ohta, Yuka Teteisi and Jun'ichi Tsujii, 2003. GENIA corpus – a semantically annotated corpus for bio-textmining. *Bioinformatics*, (supplement: 11th ISMB) 19:i180–i182, Oxford University Press, 2003.
- 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.
- David McClosky, Eugene Charniak, and Mark Johnson, 2006. Reranking and self-training for parser adaptation. *ACL-COLING '06*.
- Sujith Ravi, Kevin Knight and Radu Soricut, 2008. Automatic prediction of parser accuracy. *EMNLP '08*.
- Roi Reichart and Ari Rappoport, 2007. An ensemble method for selection of high quality parses. *ACL '07*.
- Roi Reichart and Ari Rappoport, 2009a. Sample selection for statistical parsers: cognitively driven algorithms and evaluation measures. *CoNLL '09*.
- Roi Reichart and Ari Rappoport, 2009b. Automatic selection of high quality parses created by a fully unsupervised parser. *CoNLL '09*.
- Yoav Seginer, 2007. Fast unsupervised incremental parsing. *ACL '07*.
- Noah Smith and Jason Eisner, 2006. Annealing structural bias in multilingual weighted grammar induction. *ACL-COLING '06*.
- Valentin Spitzkovsky, Hiyan Alshawi, and Daniel Jurafsky, 2010. From baby steps to leapfrog: how “less is more” in unsupervised dependency parsing. *NAACL '10*.
- Ricardo Vilalta and Irina Rish, 2003. A decomposition of classes via clustering to explain and improve naive bayes. *ECML '03*.
- Alexander Yates, Stefan Schoenmackers and Oren Etzioni, 2006. Detecting parser errors using web-based semantic filters. *EMNLP '06*.