Automatic Domain Adaptation for Parsing

David McClosky[a,b]
[a]Stanford University
Stanford, CA, USA
mcclosky@stanford.edu

Eugene Charniak[b]
[b]Brown University
Providence, RI, USA
ec@cs.brown.edu

Mark Johnson[c,b]
[c]Macquarie University
Sydney, NSW, Australia
mjohnson@science.mq.edu.au

Abstract

Current statistical parsers tend to perform well only on their training domain and nearby genres. While strong performance on a few related domains is sufficient for many situations, it is advantageous for parsers to be able to generalize to a wide variety of domains. When parsing document collections involving heterogeneous domains (e.g. the web), the optimal parsing model for each document is typically not obvious. We study this problem as a new task — *multiple source parser adaptation*. Our system trains on corpora from many different domains. It learns not only statistics of those domains but quantitative measures of domain differences and how those differences affect parsing accuracy. Given a specific target text, the resulting system proposes linear combinations of parsing models trained on the source corpora. Tested across six domains, our system outperforms all non-oracle baselines including the best domain-independent parsing model. Thus, we are able to demonstrate the value of customizing parsing models to specific domains.

1 Introduction

In statistical parsing literature, it is common to see parsers trained and tested on the same textual domain (Charniak and Johnson, 2005; McClosky et al., 2006a; Petrov and Klein, 2007; Carreras et al., 2008; Suzuki et al., 2009, among others). Unfortunately, the performance of these systems degrades on sentences drawn from a different domain. This issue can be seen across different parsing models (Sekine, 1997; Gildea, 2001; Bacchiani et al., 2006; McClosky et al., 2006b). Given that some aspects of syntax are domain dependent (typically at the lexical level), single parsing models tend to not perform well across all domains (see Table 1). Thus, statistical parsers inevitably learn some domain-specific properties in addition to the more general properties of a language's syntax. Recently, Daumé III (2007) and Finkel and Manning (2009) showed techniques for training models that attempt to separate domain-specific and general properties. However, even when given models for multiple training domains, it is not straightforward to determine which model performs best on an arbitrary piece of novel text.

This problem comes to the fore when one wants to parse document collections where each document is potentially its own domain. This shows up particularly when parsing the web. Recently, there has been much interest in applying parsers to the web for the purposes of information extraction and other forms of analysis (c.f. the CLSP 2009 summer workshop "Parsing the Web: Large-Scale Syntactic Processing"). The scale of the web demands an automatic solution to the domain detection and adaptation problems. Furthermore, it is not obvious that human annotators can determine the optimal parsing models for each web page.

Our goal is to study this exact problem. We create a new parsing task, *multiple source parser adaptation*, designed to capture cross-domain performance along with evaluation metrics and baselines. Our new task involves training parsing models on labeled and unlabeled corpora from a variety of domains (*source domains*). This is in contrast to standard domain adaptation tasks where there is a single source domain. For evaluation, one is given a text (*target text*) but not the identity of its domain. The challenge is determining how to best use the available

	Test						
Train	BNC	GENIA	BROWN	SWBD	ETT	WSJ	Average
GENIA	66.3	**83.6**	64.6	51.6	69.0	66.6	67.0
BROWN	81.0	71.5	**86.3**	79.0	80.9	80.6	79.9
SWBD	70.8	62.9	75.5	**89.0**	75.9	69.1	73.9
ETT	72.7	65.3	75.4	75.2	81.9	73.2	73.9
WSJ	**82.5**	74.9	83.8	78.5	**83.4**	**89.0**	**82.0**

Table 1: Cross-domain *f*-score performance of the Charniak (2000) parser. Averages are macro-averages. Performance drops as training and test domains diverge. On average, the WSJ model is the most accurate.

resources from training to maximize accuracy across multiple target texts.

Broadly put, we model how domain differences influence parsing accuracy. This is done by taking several computational measures of domain differences between the target text and each source domain. We use these features in a simple linear regression model which is trained to predict the accuracy of a parsing model (or, more generally, a mixture of parsing models) on a target text. To parse the target text, one simply uses the mixture of parsing models with the highest predicted accuracy. We show that our method is able to predict these accuracies quite well and thus effectively rank parsing models formed from mixtures of labeled and automatically labeled corpora.

In Section 2, we detail recent work on similar tasks. Our regression-based approach is covered in Section 3. We describe an evaluation strategy in Section 4. Section 5 presents new baselines which are intended to give a sense of current approaches and their limitations. The results of our experiments are detailed in Section 6 where we show that our system outperforms all non-oracle baselines. We conclude with a discussion and future work (Section 7).

2 Related work

The closest work to ours is Plank and Sima'an (2008), where unlabeled text is used to group sentences from WSJ into subdomains. The authors create a model for each subdomain which weights trees from its subdomain more highly than others. Given the domain specific models, they consider different parse combination strategies. Unfortunately, these methods do not yield a statistically significant improvement.

Multiple source domain adaptation has been done for other tasks (e.g. classification in (Blitzer et al., 2007; Daumé III, 2007; Dredze and Crammer, 2008)) and is related to multitask learning. Daumé III (2007) shows that an extremely simple method delivers solid performance on a number of domain adaptation classification tasks. This is achieved by making a copy of each feature for each source domain plus the “general” pseudodomain (for capturing domain independent features). This allows the classifier to directly model which features are domain-specific. Finkel and Manning (2009) demonstrate the hierarchical Bayesian extension of this where domain-specific models draw from a general base distribution. This is applied to classification (named entity recognition) as well as dependency parsing. These works describe how to train models in many different domains but sidestep the problem of domain detection. Thus, our work is orthogonal to theirs.

Our domain detection strategy draws on work in parser accuracy prediction (Ravi et al., 2008; Kawahara and Uchimoto, 2008). These works aim to predict the parser performance on a given target sentence. Ravi et al. (2008) frame this as a regression problem. Kawahara and Uchimoto (2008) treat it as a binary classification task and predict whether a specific parse is at a certain level of accuracy or higher. Ravi et al. (2008) show that their system can be used to return a ranking over different parsing models which we extend to the multiple domain setting. They also demonstrate that training their model on WSJ allows them to accurately predict parsing accuracy on the BROWN corpus. In contrast, our models are trained over multiple domains to model which factors influence cross-domain performance.

3 Approach

We start with the assumption that all target domains are mixtures of our source domains.[1] Intuitively, these mixtures should give higher probability mass to more similar source domains. This raises the question of how to measure the similarity between domains. Our method uses multiple complementary similarity measures between the target and each source. We feed these similarity measures into a regression model which learns how domain dissimilarities hurt parse accuracy. Thus, to parse a target domain, we need only find the input that maximizes the regression function — that is, the highest scoring mixture of source domains. Our system is similar to Ravi et al. (2008) in that both use regression to predict *f*-scores and some of the features are related.

3.1 Features

Our features are designed to help the regression model determine if a particular source domain mixture is well suited for a target domain as well as the quality of a source domain mixture. While we explored a large number of features, we present here only the three that were chosen by our feature selection method (Section 6.2).

Two of our features, COSINETOP50 and UNKWORDS, are designed to approximate how similar the target domain is to a specific source domain. Only the surface form of the target text and automatic analyses are available (e.g. we can tag or parse the target text, but cannot use gold tags or trees).

Relative word frequencies are an important indicator of domain. Cosine similarity uses a spatial representation to summarize the word frequencies in a corpus as a single vector. A common method is to represent each corpus as a vector of frequencies of the k most frequent words (Schütze, 1995). This method assigns high similarity to domains with a large amount of overlap in the high-frequency vocabulary items. We experimented with several orders of magnitude for k (our feature selection method later chose $k = 50$ — see Section 6.2).

Our second feature for comparing domains, UNKWORDS, returns the percentage of words in one domain which never appear in the other domain. This can be done on the word type or token level. We opt for tokens since unknown words pose problems for parsing each time they occur. UNKWORDS provides the percentage of words in the source domain that are never seen in the target domain. Whereas COSINETOP50 examines how similar the high frequency words are from one domain, UNKWORDS tends to focus on the overlap of low frequency words.

As described, COSINETOP50 and UNKWORDS are functions only of two source domains and do not take the mixing weights of source domains into account. We experimented with several methods of incorporating mixing weights into the feature value. In practice, the one which worked best for us is to divide the mixture weight of the source domain by the raw feature value. This has the nice property that when a source is not used, the adjusted feature value is zero regardless of the raw feature value.

From pilot studies, we learned that a uniform mixture of available source domains gave strong results (further details on this in Section 5). Our last feature, ENTROPY, is intended to let the regression system leverage this and measures the entropy of the distribution over source domains. This provides a sense of uniformity.

3.2 Predicting cross-domain accuracy

For a given source domain mixture, we can create a parsing model by linearly interpolating the parsing model statistics from each source domain. The key component of our approach is a domain-aware linear regression model which predicts how well a specific parsing model will do on a given target text. The linear regressor is given values from the three features from the previous section (COSINETOP50, UNKWORDS, and ENTROPY) and returns an estimate of the *f*-score the parsing model would achieve the target text.

Training data for the regressor consists of examples of source domain mixtures and their actual *f*-scores on target texts. To produce this, we randomly sampled source domain mixtures, created parsing models for those mixtures, and then evaluated the parsing models on all of our target texts.

We used a simple technique for randomly sam-

[1] This may seem like a major limitation, but as we will show later, our method works quite well at incorporating self-trained (automatically parsed) corpora which can typically be obtained for any domain.

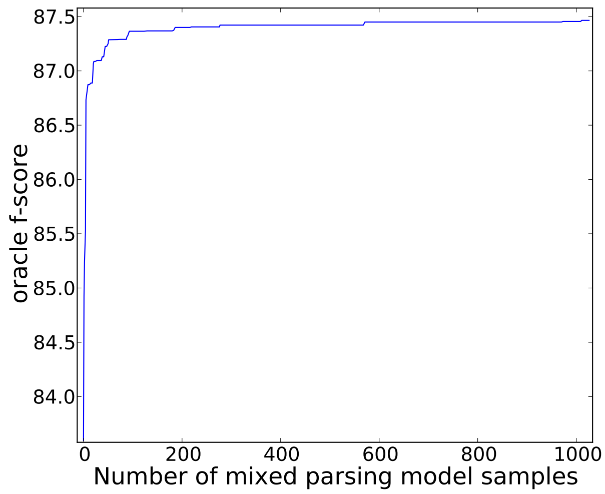

Figure 1: Cumulative oracle f-score (averaged over all target domains) as more models are randomly sampled. Most of the improvement comes the first 200 samples indicating that our samples seem to be sufficient to cover the space of good source domain mixtures.

pling source domain mixtures. First, we sample the number of source domains to use. We draw values from an exponential distribution and take their integer value until we obtain a number between two and the number of source domains. This is parametrized so that we typically only use a few corpora but still have some chance of using all of them. Once we know the number of source domains, we sample their identities uniformly at random without replacement from the list of all source domains. Finally, we sample the weights for the source domains uniformly from a simplex. The dimension of the simplex is the same as the number of source domains so we end up with a probability distribution over the sampled source domains.

In total, we sampled 1,040 source domain mixtures. We evaluated each of these source domain mixtures on the six target domains giving us 6,240 data points in total. One may be concerned that this is insufficient to cover the large space of source domain mixtures. However, we show in Figure 1 that only about 200 samples are sufficient to achieve good oracle performance[2] in practice.

Train		Test	
Source	Target	Source	Target
$\mathbf{C} \setminus \{t\}$	$\mathbf{C} \setminus \{t\}$	$\mathbf{C} \setminus \{t\}$	$\{t\}$

(a) Out-of-domain evaluation

Train		Test	
Source	Target	Source	Target
$\mathbf{C}$	$\mathbf{C} \setminus \{t\}$	$\mathbf{C}$	$\{t\}$

(b) In-domain evaluation

Table 2: List of domains allowed in single round of evaluation. In each round, the evaluation corpus is t. $\mathbf{C}$ is the set of all target domains.

4 Evaluation

Multiple-source domain adaptation is a new task for parsing and thus some thought must be given to evaluation methodology. We describe two evaluation scenarios which differ in how foreign the target text is from our source domains. Schemas for these evaluation scenarios are shown in Table 2. Note that training and testing here refer to training and testing of our regression model, **not** the parsing models.

In the first scenario, *out-of-domain evaluation*, one target domain is completely removed from consideration and only used to evaluate proposed models at test time. The regressor is trained on training points that use any of the remaining corpora, $\mathbf{C} \setminus \{t\}$, as sources or targets. For example, if $t =$ WSJ, we can train the regressor on all data points which don't use WSJ (or any self-trained corpora derived from WSJ) as a source or target domain. At test time, we are given the text of WSJ's test set. From this, our system creates a parsing model using the remaining available corpora for parsing the raw WSJ text.

This evaluation scenario is intended to evaluate how well our system can adapt to an entirely new domain with only raw text from the new domain (for example, parsing biomedical text when none is available in our list of source domains). Ideally, we would have a large number of web pages or other documents from other domains which we could use solely for evaluation. Unfortunately, at this time, only a handful of domains have been annotated with constituency structures under the same

[2] We calculate this by picking the best available model for each target domain and taking the average of their f-scores. This can pick different models for each target domain.

annotation guidelines. Instead, we hold out each hand-annotated domain, t, (including any automatically parsed corpora derived from that source domain) as a test set in a round-robin fashion.[3] For each round of the round robin we obtain an f-score and we report the mean and variance of the f-scores for each model.

The second scenario, *in-domain evaluation*, allows the target domain, t, to be used as a source domain in training but not as a target domain. This is intended to evaluate the situation where the target domain is not actually that different from our source domains. The in-domain evaluation can approximate how our system would perform when, for example, we have WSJ as a source domain and the target text is news from a source other than WSJ. Thus, our model still has to learn that WSJ and the North American News Text corpus (NANC) are good for parsing news text like WSJ without seeing any direct evaluations of the sort (WSJ and NANC can be used in models which are evaluated on all *other* corpora, though).

5 Baselines

Given that this is a new task for parsing, we needed to create baselines which demonstrate the current approaches to multiple-source domain adaptation. One approach is to take all available corpora and mix them together uniformly.[4] The UNIFORM baseline does exactly this using the available hand-built training corpora. SELF-TRAINED UNIFORM uses self-trained corpora as well. In the out-of-domain scenario, these exclude the held out domain, but in the in-domain setting, the held out domain is included. These baselines are similar to the ALL and WEIGHTED baselines in Daumé III (2007).

Another simple baseline is to use the same parsing model regardless of target domain. This is how large heterogeneous document collections are typically parsed currently. We use the WSJ corpus since it is the best single corpus for parsing all six target domains (see Table 1). We refer to this baseline as FIXED SET: WSJ. In the out-of-domain scenario, we fall back to SELF-TRAINED UNIFORM when the target domain is WSJ while the in-domain scenario uses the WSJ model throughout.

There are several interesting oracle baselines as well which serve to measure the limits of our approach. These baselines examine the resulting f-scores of models and pick the best model according to some criteria. The first oracle baseline is BEST SINGLE CORPUS which parses each corpus with the source domain that maximizes performance on the target domain. In almost all cases, this baseline selects each corpus to parse itself.

Our second oracle baseline, BEST SEEN, chooses the best parsing model from all those explored for each test set. Recall that while training the regression model in Section 3.2, we needed to explore many possible source domain mixtures to approximate the complete space of mixed parsing models. To the extent that we can fully explore the space of mixed parsing models, this baseline represents an upper bound for model mixing approaches. Since fully exploring the space of possible weightings is intractable, it is not a true upper bound. While it is theoretically possible to beat this pseudo-upper bound, (indeed, this is the mark of a good domain detection system) it is far from easy. We provide BEST SINGLE CORPUS and BEST SEEN for both in-domain and out-of-domain scenarios. The out-of-domain scenario restricts the set of possible models to those not including the target domain.

Finally, we searched for the BEST OVERALL MODEL. This is the model with the highest average f-score across all six target domains. This baseline can be thought of as an oracle version of FIXED SET: WSJ and demonstrates the limit of using a single parsing model regardless of target domain. Naturally, the very nature of this baseline places it only in the in-domain evaluation scenario. Since it was able to select the model according to f-scores on our six target domains, its performance on domains outside that set is not guaranteed.

To provide a better sense of the space of mixed parsing models, we also provide the WORST SEEN baseline which picks the worst model available for a specific target corpus.[5]

[3] Thus, the schemas in Table 2 are schemas for each round.

[4] Accounting for size so that the larger corpora don't overwhelm the smaller ones.

[5] This turns out to be GENIA for all corpora other than GENIA and SWBD when the target domain is GENIA.

6 Experiments

Our experiments use the Charniak (2000) generative parser. We describe the corpora used in our nine source and six target domains in Section 6.1. In Section 6.2, we provide a greedy strategy for picking features to include in our regression model. The results of our experiments are in Section 6.3.

6.1 Corpora

We aimed to include as many different domains as possible annotated under compatible schemes. We also tried to include human-annotated corpora and automatically labeled corpora (self-trained corpora as in McClosky et al. (2006a) which have been shown to work well across domains). Our final set includes text from news (WSJ, NANC), broadcast news (ETT), literature (BROWN, GUTENBERG), biomedical (GENIA, MEDLINE), spontaneous speech (SWBD), and the British National Corpus (BNC). In our experiments, self-trained corpora cannot be used as target domains since we lack gold annotations and BNC is not used as a source domain due to its size. An overview of our corpora is shown in Table 3.

We use news articles portion of the Wall Street Journal corpus (WSJ) from the Penn Treebank (Marcus et al., 1993) in conjunction with the self-trained North American News Text Corpus (NANC, Graff (1995)). The English Translation Treebank, ETT (Bies, 2007), is the translation[6] of broadcast news in Arabic. For literature, we use the BROWN corpus (Francis and Kučera, 1979) and the same division as (Gildea, 2001; Bacchiani et al., 2006; McClosky et al., 2006b). We also use raw sentences which we downloaded from Project Gutenberg[7] as a self-trained corpus. The Switchboard corpus (SWBD) consists of transcribed telephone conversations. While the original trees include disfluency information, we assume our speech corpora have had speech repairs excised (e.g. using a system such as Johnson et al. (2004)). Our biomedical data comes from the GENIA treebank[8] (Tateisi et al., 2005), a corpus of abstracts from the Medline database.[9] We downloaded additional sentences from Medline for our self-trained MEDLINE corpus. Unlike the other two self-trained corpora, we include two versions of MEDLINE. These differ on whether they were parsed using GENIA or WSJ as a base model to study the effect on cross-domain performance. Finally, we use a small number of sentences from the British National Corpus (BNC) (Foster and van Genabith, 2008).[10] The sentences were chosen randomly, so each one is potentially from a different domain. On the other hand, BNC can be thought of as its own domain in that it contains significant lexical differences from the American English used in our other corpora.

We preprocessed the corpora to standardize many of the annotation differences. Thus, our results on them may be slightly different than other works on these corpora. Nevertheless, these changes should not significantly impact overall the performance.

6.2 Feature selection

While our final model uses only three features, we considered many other possible features (not described due to space constraints). In order to explore these without hill climbing on our test data, we created a round-robin tuning scenario. Since the out-of-domain evaluation scenario holds out one target domain, this gives us six test evaluation rounds. For each of these six rounds, we hold out one of the remaining five target domains for tuning. This gives us 30 tuning evaluation rounds and we pick our features to optimize our aggregate performance over all of them. A model that performs well in this situation has proven that it has useful features which transfer to unknown target domains.

The next step is to determine the loss function to minimize. Our primary guide is *oracle f-score loss* which we determine as follows. We take all test data points (i.e. mixed parsing models evaluated on the target domain) and predict their *f*-scores with our model. In particular for this measure, we are interested in the point with the highest predicted *f*-score. We take its actual *f*-score which we call the *candidate f-score*. When tuning, we know the true *f*-scores of all test points. The difference between the highest *f*-score (the oracle *f*-score for

[6] The transcription and translation were done by humans.

[7] http://gutenberg.org/

[8] http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/

[9] http://www.ncbi.nlm.nih.gov/PubMed/

[10] http://nclt.computing.dcu.ie/~jfoster/resources/, downloaded January 8th, 2009.

Corpus	Source?	Target?	Average length	Train	Tune	Test
BNC		●	28.3	—	—	1,000
BROWN	●	●	20.0	19,786	2,082	2,439
ETT	●	●	25.6	2,639	1,029	1,166
GENIA	●	●	27.5	14,326	1,361	1,360
MEDLINE	●		27.2	278,192	—	—
SWBD	●	●	9.2	92,536	5,895	6,051
WSJ	●	●	25.5	39,832	1,346	2,416
NANC	●		23.2	915,794	—	—
GUTENBERG	●		26.2	689,782	—	—
MEDLINE	●		27.2	278,192	—	—

Table 3: List of source and target domains, sizes of each division in trees, and average sentence length. Indented rows indicate self-trained corpora parsed using the non-indented row as a base parser.

this dataset) and the candidate *f*-score is the oracle *f*-score loss. Ties need to be handled correctly to avoid degenerate models.[11] If there is a tie for highest predicted *f*-score, the candidate *f*-score is the one with the *lowest* actual *f*-score. This approach is conservative but ensures that regression models which give everything the same predicted *f*-score do not receive zero oracle *f*-score loss.

Armed with a tuning regime and a loss function, we created a procedure to pick the combination of features to use. We used a parallelized best-first search procedure. At each round, it expanded the current best set of features by adding or removing each feature where 'best' was determined by the loss function. We explored over 6,000 settings, though the best setting of (UNKWORDS, COSINETOP50, ENTROPY) was found within the first 200 settings explored. The best setting obtains an oracle *f*-score loss of 0.37 and a root mean squared error of 0.48 — these numbers are quite low and show the high accuracy of our regression model (similar to those in Ravi et al. (2008)). Additionally, the features are complementary in that UNKWORDS focuses on low frequency words whereas COSINETOP50 looks only at high frequency words and ENTROPY functions as a regularizer.

6.3 Results

We present an overview of our final results for out-of-domain and in-domain evaluation in Table 4. The results include the *f*-score macro-averaged over the six target domains and their standard deviation.

In both situations, the FIXED SET: WSJ baseline performs fairly poorly. Not surprisingly, assuming all of our target domains are close enough to WSJ works badly for our set of target domains and it does particularly poorly on SWBD and GENIA. On average, the UNIFORM baseline does slightly better for out-of-domain and over 3% better for in-domain. UNIFORM actually does fairly well on out-of-domain except on GENIA. In general, using more source domains is better which partially explains the success of UNIFORM. This seems to be the case since even if a source domain is terribly mismatched with the target domain, it may still be able to fill in some holes left by the other source domains. Of course, if it overpowers more relevant domains, performance may suffer. The SELF-TRAINED UNIFORM baseline uses even more source domains as well as the largest ones. In both scenarios, this dramatically improves performance and is the second best non-oracle system. This baseline provides more evidence as to the power of self-training for improving parser adaptation. If we excluded all self-trained corpora, our performance on this task would be substantially worse. We believe the self-trained corpora are beneficial in this task since they help reduce data sparsity of smaller corpora. The BEST SINGLE CORPUS baseline is poor in the out-of-domain scenario primarily because the actual best single corpus is excluded by the task specification in most cases. When we move to in-domain, this baseline improves

[11] For example, regression models which assign every parsing model the same *f*-score.

Oracle	Baseline or model	Average f-score
●	Worst seen	62.0 ± 6.1
●	Best single corpus	81.0 ± 2.9
	Fixed set: WSJ	81.0 ± 3.5
	Uniform	81.4 ± 3.6
	Self-trained uniform	83.4 ± 2.5
	Our model	84.0 ± 2.5
●	Best seen	84.3 ± 2.6

(a) Out-of-domain evaluation

Oracle	Baseline or model	Average f-score
	Fixed set: WSJ	82.0 ± 4.8
	Uniform	85.4 ± 2.4
●	Best single corpus	85.6 ± 2.9
	Self-trained uniform	86.1 ± 2.0
●	Best overall model	86.2 ± 1.9
	Our model	86.9 ± 2.4
●	Best seen	87.5 ± 2.1

(b) In-domain evaluation

Table 4: Baselines and final results for the two multiple-source domain adaptation evaluation scenarios. Results include f-scores, macro-averaged over all six target domains and their standard deviations.

but is still worse than SELF-TRAINED UNIFORM on average. It beats SELF-TRAINED UNIFORM primarily on WSJ, SWBD, and GENIA indicating that these three domains are best when not diluted by others. By definition, the WORST SEEN baseline does terribly, almost 20% worse then BEST SINGLE CORPUS.

Our model is the best non-oracle system for both evaluation scenarios. For out-of-domain evaluation, our system is only 0.3% worse than the BEST SEEN models for each target domain. For the in-domain scenario, we are within 0.6% of the BEST SEEN models. For a sense of scale, our out-of-domain and in-domain f-scores on WSJ are 83.1% and 89.8% respectively. Both numbers are quite close to the BEST SEEN baseline. Additionally, our model is 0.7% better than the BEST OVERALL MODEL. Recall that the BEST OVERALL MODEL is the single model with the best performance across all six target domains.[12] By beating this baseline, we show that there is value in customizing parsing models to the target domain. It is also interesting that the BEST OVERALL MODEL is only marginally better than SELF-TRAINED UNIFORM. Without any further information about the target corpus, an uninformed prior appears best.

7 Discussion

We have shown that for both out-of-domain and in-domain evaluations, our system is well adapted to predicting the effects of domain divergence on parsing accuracy. Using the parsing model with the highest predicted f-score leads to great performance in practice. There is a substantial benefit to doing this over existing approaches (using the same model for all domains or mixing all training data together uniformly). Creating a number of domain-specific models and mixing them together as needed is a viable approach.

One can think of our system as trying to estimate document-level context. Our representation of this context is simply a distribution over our source domains, but one can imagine more complex options such as a high-dimensional vector space. Additionally, our model separates domain and syntax estimation, but a future direction is to learn these jointly. This would combine our work with (Daumé III, 2007; Finkel and Manning, 2009).

We have focused on the Charniak (2000) parser, the first stage in the two stage Charniak and Johnson (2005) reranking parser. Applying our methods to other generative parsers (such as (Collins, 1999; Petrov and Klein, 2007)) is trivial, but it is less clear how our methods can be applied to the discriminative reranker component of the two stage parser. One avenue of approach is to incorporate the domain representation into the feature space, as in Daumé III (2007) but with more complex domain information.

Acknowledgments

This work was performed while the first author was at Brown and supported by DARPA GALE contract HR0011-06-2-0001. We would like to thank the BLLIP team and our anonymous reviewers for their comments.

[12] Somewhat surprisingly, the best overall model uses almost entirely self-trained corpora consisting of 9.5% GUTENBERG, 60.3% NANC, 26.0% MEDLINE (by GENIA), and 4.2% SWBD.

References

Michiel Bacchiani, Michael Riley, Brian Roark, and Richard Sproat. 2006. MAP adaptation of stochastic grammars. *Computer Speech and Language*, 20(1):41–68.

Ann Bies. 2007. *GALE Phase 3 Release 1 - English Translation Treebank*. Linguistic Data Consortium. LDC2007E105.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Association for Computational Linguistics*, Prague, Czech Republic.

Xavier Carreras, Michael Collins, and Terry Koo. 2008. TAG, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *Proceedings of CoNLL 2008*, pages 9–16, Manchester, England, August.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine *n*-best parsing and MaxEnt discriminative reranking. In *Proceedings of the ACL 2005*, pages 173–180.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the North American Chapter of the ACL (NAACL)*, pages 132–139.

Michael Collins. 1999. *Head-driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, The University of Pennsylvania.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of ACL 2007*, Prague, Czech Republic.

Mark Dredze and Koby Crammer. 2008. Online methods for multi-domain learning and adaptation. In *Proceedings of the EMNLP 2008*, pages 689–697, Honolulu, Hawaii, October.

Jenny Rose Finkel and Christopher D. Manning. 2009. Hierarchical bayesian domain adaptation. In *Proceedings of HLT-NAACL 2009*, pages 602–610, Boulder, Colorado, June.

Jennifer Foster and Josef van Genabith. 2008. Parser evaluation and the bnc: Evaluating 4 constituency parsers with 3 metrics. In *Proceedings LREC 2008*, Marrakech, Morocco, May.

W. Nelson Francis and Henry Kučera. 1979. *Manual of Information to accompany a Standard Corpus of Present-day Edited American English*, for use with Digital Computers. Brown University, Providence, Rhode Island.

Daniel Gildea. 2001. Corpus variation and parser performance. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 167–202.

David Graff. 1995. *North American News Text Corpus*. Linguistic Data Consortium. LDC95T21.

Mark Johnson, Eugene Charniak, and Matthew Lease. 2004. An improved model for recognizing disfluencies in conversational speech. In *Proc. of the Rich Text 2004 Fall Workshop (RT-04F)*.

Daisuke Kawahara and Kiyotaka Uchimoto. 2008. Learning reliability of parses for domain adaptation of dependency parsing. In *Third International Joint Conference on Natural Language Processing (IJCNLP '08)*.

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

David McClosky, Eugene Charniak, and Mark Johnson. 2006a. Effective self-training for parsing. In *Proceedings of HLT-NAACL 2006*, pages 152–159.

David McClosky, Eugene Charniak, and Mark Johnson. 2006b. Reranking and self-training for parser adaptation. In *Proceedings of COLING-ACL 2006*, pages 337–344, Sydney, Australia, July. Association for Computational Linguistics.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 404–411, Rochester, New York, April. Association for Computational Linguistics.

Barbara Plank and Khalil Sima'an. 2008. Subdomain sensitive statistical parsing using raw corpora. In *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, May.

Sujith Ravi, Kevin Knight, and Radu Soricut. 2008. Automatic prediction of parser accuracy. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 887–896, Honolulu, Hawaii, October. Association for Computational Linguistics.

Hinrich Schütze. 1995. Distributional part-of-speech tagging. In *Proceedings of the 7th conference of the EACL*, pages 141–148.

Satoshi Sekine. 1997. The domain dependence of parsing. In *Proc. Applied Natural Language Processing (ANLP)*, pages 96–102.

Jun Suzuki, Hideki Isozaki, Xavier Carreras, and Michael Collins. 2009. An empirical study of semi-supervised structured conditional models for dependency parsing. In *Proceedings EMNLP 2009*, pages 551–560, Singapore, August.

Yuka Tateisi, Akane Yakushiji, Tomoko Ohta, and Jun'ichi Tsujii. 2005. Syntax Annotation for the GENIA corpus. *Proceedings of IJCNLP 2005, Companion volume*, pages 222–227.