Modern statistical parsers are trained on large annotated corpora (treebanks). These treebanks usually consist of sentences addressing different subdomains (e.g., sports, politics, music), which implies that the statistics gathered by current statistical parsers are mixtures of large syntactically annotated corpora (treebanks) and their
Current state-of-the-art statistical parsers are trained on bank the statistics gathered are averages over different sub-
domains. By definition, averages smooth-out the statistical differences between the individual subdomains and weaken the biases in the model.
The present paper describes a method for incorporating subdomain sensitivity into an existing state-of-the-art parser (Collins, 1997; Bikel, 2002) in order to improve its predictions. The main idea is to create and combine an ensemble of subdomain sensitive parsers (each for a different subdomain) without the need for further manual annotation of subdomain data. We exploit unannotated, subdomain specific corpora gathered from the web, for re-weighting the original treebank trees to reflect subdomain statistics, and employ that for training individual subdomain sensitive parsers. In what follows we describe first our re-weighting method for tree-weighting followed by methods for parser combination.

2. Subdomain Sensitive Parsing
To exploit domain-information and create an ensemble of subdomain sensitive parsers, one approach, used in (Bod, 1999), is to partition the given training treebank into disjoint subtreebanks, each addressing one subdomain. We think that it is often not that straightforward (sometimes even impossible) to partition the training data along clear-cut subdomain borderlines. Furthermore, this approach leads to sparse data problems.
In this paper we suggest a more subtle method than partitioning the treebank. For every subdomain, we create a “subdomain specific version” of the original treebank as follows. For every tree in the original treebank we give it a new count/weight that expresses the likelihood it will appear in the given subdomain. This method can be seen as sampling tree-types according to the specific subdomain distribution. We obtain subdomain specific parsers by training a statistical parser on each of the resulting subdomain specific versions separately. In the sequel, we present the method for tree-weighting followed by methods for parser combination.

2.1. Tree Weighting using LMs
Our Tree Weighting approach exploits Statistical Language Models induced from raw subdomain-dependent corpora. In more detail, given a set of subdomains and their corresponding raw corpora, we define a weighting function over the set of training instances in the original treebank $TB$:

1. For each subdomain-specific corpus, induce a Language Model (LM) $\theta$ from the raw subdomain corpus.
2. For every tree $\pi_i$ in the treebank $TB$, define its count under LM $\theta$ to be equal to the average per-word count of its yield $y_{[\pi_i]}$ under LM $\theta$:

$$f_\theta(\pi_i) = f_\theta(y_{[\pi_i]}) = -\log P_\theta(y_{[\pi_i]})/n$$

where $n$ is the length of yield $y_{[\pi_i]}$ and $P_\theta$ is the probability under LM $\theta$. Thus, the count is given by the length-normalized probability of the yield.
3. Let $f_{\theta}^{\max}$ be the maximum count of a tree in $TB$ according to $\theta$. The weight $w_i$ assigned to $\pi_i$ is defined as:

$$w_i = \text{round}\left\{ \left( \frac{f_{\theta}^{\max}}{f_\theta(\pi_i)} \right)^a \right\} \quad (1)$$

where $a \geq 1$ is a scaling constant. In the default setting $a = 1$. 
The Tree Weighting function thus defined for each subdomain effectively results in a subdomain-weighted treebank $T_{B_d}$. We employ each of these treebanks $T_{B_d}$ to retrain the original parser, thereby obtaining a subdomain sensitive parser. The relevant details regarding the subdomain corpora, the treebank and parser used in this study are given in the experimental section of this paper. In the next section, we propose techniques on how to combine the domain-dependent parsers.

2.2. Parser Combination Techniques

Given an ensemble of parsers, the question which naturally follows and is addressed in this section is to devise methods for combining them in order to get a single, final result. There are various ways to combine parsers, including methods for combining the outputs of classifiers by voting schemes or even parse-reranking schemes based on log-linear models (Charniak and Johnson, 2005). For our preliminary exploration in parser combination, we choose here two simple methods:

- parser pre-selection (choose a parser for a given input sentence) and
- post-selection (choose any of the output trees of the parsers given the input).

Probabilistically speaking, the two methods can be combined in a Bayesian decision rule which combines $P(\text{dom}|s)$, i.e. the (pre-selection) probability of the parser $\text{dom}$ given input sentence $s$, with $P(t|\text{dom}, s)$, the (post-selection) probability of the output tree $t$ given the parser $\text{dom}$ and input sentence $s$:

$$\arg \max_t \sum_{\text{dom}} P(\text{dom}|s) P(t|\text{dom}, s)$$

However, it is hard to obtain good estimates of $P(\text{dom}|s)$. Furthermore, the different subdomains we work with cannot comply with the theoretical requirement of being disjoint, thereby conflicting with the Bayesian formula. In this initial work we concentrate on exploring approaches to pre- and post-selection separately using statistical measures. Combining the two in the Bayesian framework is a much harder task and is left for future research.

2.3. Pre-selection: Divergence Model (DVM)

In parser pre-selection, the goal is to select one of the subdomain parsers to parse an input sentence. The probability obtained from a subdomain language model might not be sensitive enough to discriminate between the different subdomains. A suitable measure must weight words that are more typical for a certain subdomain higher than words shared across subdomains. This is the idea behind our proposed Divergence Model (DVM).

Given $k$ raw subdomain corpora, and $k$ unsmoothed unigram Language Models $\theta_1, \ldots, \theta_k$ induced from the corpora, the divergence of some subdomain $i \in [1, \ldots, k]$ on word type $w$ from the $k-1$ other subdomains ($j \in [1, \ldots, k] : j \neq i$) is defined as:

$$\text{divergence}_i(w) = 1 + \frac{\sum_{j \in [1, \ldots, k] : j \neq i} \log \frac{p_{\theta_j}(w)}{p_{\theta_i}(w)}}{(k-1)}$$

where $|X|$ denotes the absolute value of $X$. The border conditions for this function are given by

- if $p_{\theta_i}(w) = 0$ then $\text{divergence}_i(w) = 1$, and
- if $p_{\theta_i}(w) = 0$, then redefine $p_{\theta_i}(w) = \text{const}$ where $\text{const}$ is a number between 0 and the smallest unigram Language Model probability. In our case, we set $\text{const} = 10^{-15}$.

We define the divergence of a sentence $w^n = w_1, \ldots, w_n$ to be equal to the average per word divergence:

$$s_{\text{divergence}} \left( i \in [1, \ldots, k], w^n \right) = \frac{\sum_{i=1}^{n} \text{divergence}_i(w_i)}{n}$$

(3)

This divergence score is used to select among the available $k$ subdomain parsers.

2.4. Post-selection: Node Weighting including DVM

In parser post-selection, the input sentence is given to all subdomain parsers and the output parse is selected from the set of output parse trees. The technique we introduce combines a constituent-based score through linear interpolation with the sentence-level score as given by the DVM. We will refer to this method as Node Weighting including the Divergence Model (NW-DVM).

Given $k$ candidate parse trees $\pi_i$, for $1 \leq i \leq k$, output by $k$ subdomain parsers. Let $s(c, \pi_i)$ be a boolean function, which is equal to 1 if tree $\pi_i$ contains constituent $c$, and 0 otherwise. The score for a constituent $c \in \pi_i$ is defined as its average occurrence in all $k$ output trees:

$$\text{score}(c) = \left[ \frac{\sum_{i=1}^{k} s(c, \pi_i)}{k} \right]$$

(4)

Given the function $\text{score}(c)$, defined in the range $[0,\ldots,1]$, and the divergence score of a sentence as defined in equation 3, a score for a tree $\pi_i$ is calculated as:

$$\text{score}(\pi_i) = (1-\lambda) \left[ \frac{\sum_{c \in \pi_i} \text{score}(c)}{|\pi_i|} \right] + \lambda \cdot s_{\text{divergence}} \left( i \in [1, \ldots, k], w^n \right)$$

(5)

where $|\pi_i|$ denotes the number of constituents in $\pi_i$, and $\lambda \in [0, 1]$ is an interpolation parameter set experimentally to balance the contribution of the DVM and the constituent-weighting subterms.

3. Experiments

All experiments were performed using Bikel’s parser (Bikel, 2002), an emulation of Collins’ Head-lexicalized Probabilistic Context Free Grammar (PCFG) model (Collins, 1997; Collins, 2003). We use the Penn Treebank (PT) Wall Street Journal (WSJ) (Marcus et al., 1993) version 2, with the now ’standard division’ (Collins, 1997; Collins, 2003) into training (sections 02-21), test (section 23) and development/dev (section 00) sets.

As concepts constituting possible subdomains within the PT WSJ we assume: FINANCIAL, POLITICS and SPORTS. For the POLITICS subdomain we use the English part of the Europarl Parallel Corpus (Koehn, 2005). For the
FINANCIAL and SPORTS subdomains, to the best of our knowledge there were no ready-to-use corpora available. Hence, we used Wikipedia (Wikipedia Foundation Inc., 2007) to create domain-specific corpora ourselves. We used Wikipedia’s category system to extract relevant articles from the English Wikipedia’s dump file, cleaned the articles from Wiki-syntax and segmented them into a one sentence-per-line corpus. The size of the resulting raw domain-specific corpora ranges from 6 to 11 million tokens.  

For each of the possible subdomains Statistical Language Models (LMs) were estimated and smoothed (using Chen and Goodman’s modified Kneser-Ney smoothing) by using the SRI Language Modeling Toolkit\(^1\) (SRILM). Then, we applied our Tree Weighting function (section 2.1.) to create the subdomain specific training data\(^2\) for the subdomain-sensitive parsers. The size of the resulting treebanks is between 127k and 160k training instances.

Figure 1 summarizes our experimental setting. The subdomain-weighted versions of the training treebank \(TB_\theta\) are created to train the respective parsing model. Our ensemble of parsers consists of a total of four parsers: three domain-dependent parsers, and the baseline parser. The domain-dependent parsers represent the FINANCIAL, POLITICS and SPORTS domains, respectively. The baseline parser is trained on the original treebank, the usual Penn Treebank WSJ sections 02 to 21, and is included in the ensemble to represent the "general" domain REST.

To evaluate parsing performance we use EVALB and the standard PARSEVAL evaluation metrics. Results of parsing sentences of length up to 40 words are reported for both the development (section 00) and test set (section 23).

Before assessing the ability of the parser combination techniques to select better parses, we first gauge in how far the Tree Weighting method (section 2.1.) results in a set of subdomain parsers with complementary capabilities.

### 3.1. Effectiveness of Subdomain Tree Weighting

To gauge whether subdomain Tree Weighting is giving more varied and useful subdomain specific parsers, we combine the output of the parsers using an optimal decision procedure, i.e. an oracle. Given \(i\) candidate parse trees, the oracle is a decision procedure that selects the best tree by measuring the accuracy in F-score\(^3\) against the gold standard tree.

| Parser / Combination Technique | Section 00 (devset) | Section 23 (testset) |
|-------------------------------|---------------------|---------------------|
| Baseline                      | 89.44 89.63 89.53   | 88.77 88.87 88.82   |
| Oracle combination            | 90.59 90.66 90.62   | 90.11 90.11 90.11   |
| Improvement over baseline     | +1.15 +1.03 +1.09   | +1.34 +1.24 +1.29   |

Table 1: Results of the pilot experiment

---

\(^1\)http://www.speech.sri.com/projects/srilm/

\(^2\)Note that we allowed duplicates in the original treebank before weighting.

\(^3\)The harmonic mean of labeled precision (LP) and labeled recall (LR): F-score = \(\frac{2LP \cdot LR}{LP + LR}\).
Table 2: Individual subdomain parser results

| Individual Subdomain Parser | LR       | LP       | F-score |
|-----------------------------|----------|----------|---------|
| Sports                      | 88.95    | 88.83    | 88.89   |
| Financial                   | 89.01    | 88.84    | 88.92   |
| Politics                    | 88.86    | 88.70    | 88.78   |

Table 2: Individual subdomain parser results

| Parser / Combination Technique | LR       | LP       | F-score |
|-------------------------------|----------|----------|---------|
| Baseline                      | 89.44    | 89.63    | 89.53   |
| Divergence Model (DVM)        | 89.50    | 89.68    | 89.59   |
| Node Weight. incl. DVM, \( \lambda = 0.6 \) | 89.53    | 89.71    | 89.62   |
| Baseline                      | 88.77    | 88.87    | 88.82   |
| Divergence Model (DVM)        | 88.80    | 88.91    | 88.85   |
| Node Weight. incl. DVM, \( \lambda = 0.6 \) | 88.84    | 88.96    | 88.90   |

Table 3: Results of Parser Combination Techniques

Both the parser pre-selection and the parser post-selection technique can improve the baseline system slightly. The best technique is Node Weighting including the Divergence Model (NW-DVM), yielding an improvement of +0.09% and +0.08%. These figures are only about one tenth of the +1.29% that an oracle obtains (see table 1).

In figure 3 we depict the performance of NW-DVM when instantiated with various values for \( \lambda \). We can see that for \( \lambda \geq 0.5 \) the technique slightly but constantly outperforms the baseline system, reaching a peak at \( \lambda = 0.6 \).

Figure 3: Result of NW-DVM for sentences up to 40 (WSJ-40) and 100 (WSJ-100) words, respectively.

4. Related Work

In (Bod, 1999), dialogue context/state is taken into account by splitting up the training treebank into four disjoint context-dependent subcorpora (place, time, date, yes/no). The “subdomain-dependent” parsers yield promising results. For a tiny domain as the one explored by Bod (train information system) this simplistic approach could work given that the treebank is large enough. However, in the general case, splitting the training data into disjoint subdomains implies sparse data problems for each of the subdomain parsers.

A further related study is the work by (Sekine, 1997). He analyzes the “domain dependence of parsing”. In his experiments a domain is characterized by the natural domains defined in the Brown corpus, for example ‘Press Reportage’, ‘General Fiction’ or ‘Romance and Love Story’. Sekine observes that in parsing, the data from the same domain is the most advantageous, followed by data from the same class, while training on data from another domain generally performs worst. Sekine claims that when trying “to parse a text in a particular domain, we should prepare a grammar which suits this domain” (Sekine, 1997), thus suggesting a “domain-dependent parser”.

Although different, work on domain adaptation is distantly related to our work. Recent research on adaptation is too numerous to discuss in a short paper. In particular, (Jiang and Zhai, 2007) suggest “instance weighting” as a method for adaptation. Our approach, subdomain instance weighting using raw data, can be seen as a novel version thereof. Theoretically speaking, successful domain adaptation hinges on some sense of “overlap” between the source and target domains, e.g., (Ando and Zhang, 2005). The overlap between source and target domains can be seen as a (mix of) subdomain(s) of both. Naturally, instance weighting and its subdomain instantiation can be seen as a weighted version of self-training, e.g., (McClosky et al., 2006), which is again related to co-training (Blum and Mitchell, 1998).

5. Conclusions and Outlook

This paper explores a particular instantiation for subdomain sensitive parsing. We exploit unlabeled subdomain corpora gathered from the web in order to create a set of subdomain
sensitive parsers, where each parser is assumed to specialize in its domain.

The empirical results demonstrate that our subdomain Tree Weighting method is promising as it yields an improvement of up to 1.29% F-score on the Penn Treebank Wall Street Journal. However, the experiments also show that parser combination is not a straightforward task: amalgamating parser predictions gives only a modest improvement (+0.09%) over the (very tough) baseline system.

Given the simplicity of the Treebank Weighting scheme and the examined parse(r) selection methods, the experimental results are promising and warrant further exploration. Therefore, future work will be manifold and consists in, amongst others: defining other ways of instantiating subdomain sensitive parsers by using a more sophisticated notion of subdomain (e.g. integrating relations between entities), extending the current approach to a Bayesian approach, exploring more sophisticated parser combination techniques, and examining the approach on corpora other than the WSJ, e.g., Brown corpus (see (Gildea, 2001)), thus looking at domain adaptation. Furthermore, we would like examine our approach with the current best-performing parsing systems (Charniak and Johnson, 2005). It consists of a generative n-best parsing system that employs a discriminative reranking scheme. We would like to gauge to what extent n-best parsing might benefit from subdomain information.

6. References

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. Journal of Machine Learning Research, 6:1817–1853.

Bikel, D. (2002). Design of a Multi-lingual, Parallel-processing Statistical Parsing Engine. Proceedings of HLT 2002.

Avrim Blum and Tom M. Mitchell. 1998. Combining labeled and unlabeled data with co-training. In COLT, pages 92–100.

Bod, R. (1999). Context-sensitive spoken dialogue processing with the DOP model. In Natural Language Engineering, Cambridge University Press.

Charniak, E. and Johnson, M. (2005). Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking. Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005).

Collins, M. (1997). Three Generative, Lexicalised Models for Statistical Parsing. Madison. Proceedings of the 35th Annual Meeting of the ACL-EACL.

Collins, M. (2003). Head-Driven Statistical Models for Natural Language Parsing. Association of Computational Linguistics, Gildea, D. (2001). Corpus variation and parser performance. In Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing (EMNLP).

Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in nlp. In Proceedings of ACL 2007, pages 264–271, Prague, Czech Republic, June. Association for Computational Linguistics.

Kneser, R. and Peters, J. (1997). Semantic clustering for adaptive language modeling. volume 02, page 779, Los Alamitos, CA, USA. IEEE Computer Society.

Koehn, P. (2005). Europarl: A Multilingual Corpus for Evaluation of Machine Translation. MT Summit.

Marcus, M., Santorini, B., and Marcinkiewicz, M. (1993). Building a Large Annotated Corpus of English: the Penn Treebank. Computational Linguistics, 13.

D. McClosky, E. Charniak, and M. Johnson. 2006. Reranking and self-training for parser adaptation. In Proceedings of the COLING-ACL 2006. The Association for Computer Linguistics.

Wikimedia Foundation Inc. (2007). Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/.

Sekine, S. (1997). The Domain Dependence of Parsing. Washington, DC, USA. The Fifth Conference on Applied Natural Language Processing.