We investigate the problem of parsing the noisy language of social media. We evaluate four Wall-Street-Journal-trained statistical parsers (Berkeley, Brown, Malt and MST) on a new dataset containing 1,000 phrase structure trees for sentences from microblogs (tweets) and discussion forum posts. We compare the four parsers on their ability to produce Stanford dependencies for these Web 2.0 sentences. We find that the parsers have a particular problem with tweets and that a substantial part of this problem is related to POS tagging accuracy. We attempt three retraining experiments involving Malt, Brown and an in-house Berkeley-style parser and obtain a statistically significant improvement for all three parsers.

1 Introduction

With the explosive growth in social media, natural language processing technologies, including parsers, need to adapt to reflect the linguistic changes brought about by new forms of online communication. The availability of the Penn Treebank has encouraged much research in supervised parsing for English and facilitated comparison between parsers. This has led to impressive performance for in-domain parsing. Some progress has also been achieved in adapting parsers to new domains using semi-supervised and unsupervised approaches involving some labelled source domain training data, little, if any, labelled target domain data and large quantities of unlabelled target domain data. Much of the work on parser adaptation has focused on biomedical text and questions - very little has focused on the informal language prevalent in much of the user-generated content of Web 2.0. Domain adaptation to the language of social media is particularly challenging since Web 2.0 is not really a domain, consisting, as it does, of utterances from a wide variety of speakers from different geographical and social backgrounds.

Foster (2010) carried out a pilot study on this topic by investigating the performance of the Berkeley parser (Petrov et al., 2006) on sentences taken from a sports discussion forum. Each mis-parsed sentence was examined manually and a list of problematic phenomena identified. We extend this work by looking at a larger dataset consisting not only of discussion forum posts but also microblogs or tweets. We extend the parser evaluation to the Brown reranking parser (Charniak and Johnson, 2005), MaltParser (Nivre et al., 2006) and MSTParser (McDonald et al., 2005), and we examine the ability of all four parsers to recover typed Stanford dependencies (de Marneffe et al., 2006). The relative ranking of the four parsers confirms the results of previous Stanford-dependency-based parser evaluations on other datasets (Cer et al., 2010; Petrov et al., 2010). Furthermore, our study shows that the sentences in tweets are harder to parse than the sentences from the discussion forum, despite their shorter length and that a large contributing factor is the high part-of-speech tagging error rate.

Foster’s work also included a targeted approach to improving parser performance by modifying the Penn Treebank trees to reflect observed differences between Wall Street Journal (WSJ) sentences and discussion forum sentences (subject ellipsis, non-standard capitalisation, etc.). We approach the problem from a different perspective, by seeing how far we can get by exploiting unlabelled target domain data. We employ three types of parser retraining, namely, 1) the McCloskey et al. (2006) self-training protocol, 2) uptraining of Malt using dependency trees produced by a slightly more accurate phrase structure parser (Petrov et al., 2010), and 3) PCFG-LA self-training (Huang et al., 2006).
and Harper, 2009). We combine the benefits of the dependency parsing uptraining work of Petrov et al. and the self-training protocol of McClosky et al. by retraining Malt on trees produced by a self-trained version of the Brown parser.

We find that considerable improvements can be obtained when discussion forum data is used as the source of additional training material, and more modest improvements when Twitter data is used. Grammars trained on the discussion forum data perform well on Twitter data, but the reverse is not the case. For Brown, we obtain an absolute LAS increase of 8.8% on the discussion forum data and an improvement of 5.6% on the Twitter data. For Brown, we obtain an absolute f-score improvement of 2.4% on the discussion forum data and an increase of 1.7% on Twitter. For the Berkeley-style parser that we use in the PCFG-LA self-training experiment, the f-score improvements are 4.7% and 1.2% respectively.

The novel contributions of the paper are:

1. A new dataset consisting of 1,000 hand-corrected phrase structure parse trees for sentences from two types of social media (discussion forums and tweets).
2. A detailed evaluation of four popular WSJ-trained parsers on this new dataset.
3. An investigation of how well the most successful unsupervised parser adaptation methods perform on this new dataset. Since Web 2.0 is not really a domain, it is important not to assume that the methods that have been developed for more clearly defined domains will work without carrying out the experiments.
4. A discussion of the main issues involved in parsing Web 2.0 text.

The new dataset is discussed in §2 and the baseline parser evaluation is detailed in §3. The retraining experiments are described in §4. §5 contains a discussion of how this work could be extended.

2 Web 2.0 Data

Our Web 2.0 dataset, summarised in Table 1, consists of a small treebank of 1,000 hand-corrected phrase structure parse trees and two larger corpora of unannotated sentences. The sentences in the treebank originate from discussion forum comments and microblogs (tweets). The sentences in the larger corpora are taken from the same sources as the treebank sentences.

2.1 Tweets

Hand-Corrected Parse Trees 60 million tweets on 50 topics encompassing politics, business, sport and entertainment, were collected using the public Twitter API between February and May 2009 (Bermingham and Smeaton, 2010). The microblog section of the Web 2.0 treebank contains 519 sentences taken from this corpus. The development set contains 269 sentences and the test set contains 250. Hyperlinks and usernames were replaced by the generic names Urlname and Username respectively, and the tweets were split by hand into sentences. Tweets containing just a hyperlink were not included in the treebank. For the rest of this paper, we refer to the development set as TwitterDev and the test set as TwitterTest.

Unannotated Sentences From the full Twitter corpus, we constructed a sub-corpus of approximately 1 million tweets. As with the treebank tweets, hyperlinks were replaced by the term Urlname and usernames by Username. Tweets with more than one non-ASCII character were removed, and the remaining tweets were passed through our in-house sentence splitter and tokenizer, resulting in a corpus of 1,401,533 sentences. We refer to this as the TwitterTrain corpus.

2.2 Discussion Forum Comments

Hand-Corrected Parse Trees The discussion forum section of the Web 2.0 treebank is an extension of that described in Foster (2010). It contains 481 sentences taken from two threads on the BBC Sport 606 discussion forum in November 2009.1 As with the tweets, the discussion forum posts were split into sentences by hand. The development set contains 258 sentences and the test set 223. For the remainder of the paper, we use the term FootballDev to refer to this development set and the term FootballTest to refer to the test set.

Unannotated Sentences The same discussion forum that was used to create FootballDev and FootballTest was scraped during the final quarter of 2010. The content was stripped of HTML markup and passed through an in-house sentence splitter and tokenizer, resulting in a corpus of 1,009,646 sentences. We call this the FootballTrain corpus.

1http://www.bbc.co.uk/dna/606/F15264075/thread=7065503&show=50 and http://www.bbc.co.uk/dna/606/F15265997/thread=7066136&show=50
Table 1: Basic Statistics on the Web 2.0 datasets: number of sentences, average sentence length, median sentence length and standard deviation

<table>
<thead>
<tr>
<th>Corpus Name</th>
<th>#Sen</th>
<th>SL Mean</th>
<th>SL Med.</th>
<th>σ</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwitterDev</td>
<td>269</td>
<td>11.1</td>
<td>10</td>
<td>6.4</td>
</tr>
<tr>
<td>TwitterTest</td>
<td>250</td>
<td>11.4</td>
<td>10</td>
<td>6.8</td>
</tr>
<tr>
<td>FootballTrain</td>
<td>1,401,533</td>
<td>8.6</td>
<td>7</td>
<td>6.1</td>
</tr>
<tr>
<td>FootballDev</td>
<td>258</td>
<td>17.7</td>
<td>14</td>
<td>13.9</td>
</tr>
<tr>
<td>FootballTest</td>
<td>223</td>
<td>16.1</td>
<td>14</td>
<td>9.7</td>
</tr>
<tr>
<td>FootballTrain</td>
<td>1,009,646</td>
<td>15.4</td>
<td>12</td>
<td>13.3</td>
</tr>
</tbody>
</table>

2.3 Annotation

The sentences in the Web 2.0 treebank (TwitterDev/Test and FootballDev/Test) were first parsed automatically using an implementation of the Collins Model 2 generative statistical parser (Bikel, 2004). They were then corrected by hand by one annotator, using as a reference the Penn Treebank (PTB) bracketing guidelines (Bies et al., 1995) and the PTB trees themselves. For structures which do not appear in the PTB, new annotation decisions needed to be made. An example is the annotation of hyperlinks in tweets. These were annotated as proper nouns in a single word noun phrase, and, if occurring at the end of a tweet, were attached in the same way as a nominal adverbial.

The annotator went through the dataset twice, and a second annotator then annotated 10% of the sentences (divided equally between discussion forum posts and tweets). Agreement between the two annotators on labelled bracketing is 94.2%. The sources of the disagreements involved 1) the PTB bracketing guidelines leaving open more than one annotation option (usually placement of adverbs), 2) (almost) agrammatical fragments (e.g. USA, USA, USA or Wes Brown > Drogba) and 3) multiword expressions (e.g. in fairness).

3 Baseline Evaluation

We first evaluate four widely used WSJ-trained statistical parsers on our new Web 2.0 datasets:

Berkeley (Petrov et al., 2006) We train a PCFG-LA using 6 iterations and we run the parser in accurate mode.

Brown (Charniak and Johnson, 2005) We employ this parser in its out-of-the-box settings.

Malt (Nivre et al., 2006) We use the stacklazy algorithm described in Nivre et al. (2009). We train a linear classifier where the feature interactions are modelled explicitly.

MST (McDonald et al., 2005) We use the settings described in Nivre et al. (2010).

Our training data consists of §02-21 of the WSJ section of the PTB (Marcus et al., 1994). Although our main aim in this experiment is to establish how well WSJ-trained parsers perform on our new Web 2.0 dataset, we also report performance on §22 as a reference. We use Parseval labelled f-score to compare the two phrase structure parsers. We then compare all four parsers by training the dependency parsers on WSJ phrase structure trees converted to labelled dependency trees and by converting the output of the two phrase structure parsers to labelled dependency trees. For the dependency evaluation, we use the CoNLL evaluation metrics of labelled attachment score (LAS) and unlabelled attachment score (UAS).

The labelled dependency scheme that we use is the Stanford basic dependency scheme (de Marneffe et al., 2006). We experiment with the use of gold POS tags, POS tags obtained using a POS tagger (Giménez and Márquez, 2004) and, for the phrase structure parsers, POS tags produced by the parsers themselves. The Brown parser always performs its own POS tagging. The Berkeley parser can be supplied with POS tags but it is not guaranteed to use them – trees containing the supplied POS tag for a given word may be removed from the chart during coarse-to-fine pruning.²

3.1 Results

Table 2 shows the Parseval f-score and part-of-speech (POS) tagging accuracy for the Berkeley

²In the interest of replicability, detailed information on experimental settings is available at http://nclt.computing.dcu.ie/publications/foster_ijcnlp11.html.
and Brown parsers on the three development sets.\(^3\) We observe the following: Twitter data is harder to parse than the discussion forum data; parsing accuracy is slightly higher when the Berkeley parser does its own POS tagging than when a pipeline model is employed; POS errors are a bigger problem for the Web 2.0 datasets than for the in-domain test set, particularly for TwitterDev.

The LAS and UAS scores for all four parsers are presented in Table 3. The relative ranking of the four parsers is the same as that reported in Cer et al. (2010). One striking aspect of the results is the bigger performance discrepancy between the phrase structure and dependency parsers for the discussion forum data than for the Twitter data. There is also a bigger performance discrepancy between LAS and UAS for the Web 2.0 data than for the WSJ data — this could be related to the fact that the Stanford converter has been developed using Penn Treebank trees, and it is certainly related to POS tagging accuracy since the difference is less pronounced when the input to the parsers is gold POS tags. In gold tag mode, the dependency parsers achieve slightly higher performance than the Berkeley parser for the Twitter data. This might have something to do with the 97% POS tagging accuracy for Berkeley gold tagging mode (see Table 2) but this cannot be the whole story since we do not see the same trend for the discussion forum data even though POS tagging accuracy is not 100% here either.

### 3.2 Error Analysis

In order to better understand the results in Tables 2 and 3, we examine POS confusions for the three datasets and we provide a breakdown of parsing performance by dependency type.

<table>
<thead>
<tr>
<th>Parser</th>
<th>LAS</th>
<th>UAS</th>
<th>WSJ22</th>
<th>FootballDev</th>
<th>TwitterDev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Berk O</td>
<td>90.5</td>
<td>93.2</td>
<td>79.8</td>
<td>84.8</td>
<td>68.9</td>
</tr>
<tr>
<td>Berk P</td>
<td>89.9</td>
<td>92.5</td>
<td>80.1</td>
<td>84.9</td>
<td>68.2</td>
</tr>
<tr>
<td>Brown</td>
<td>91.5</td>
<td>94.2</td>
<td>82.0</td>
<td>86.3</td>
<td>71.4</td>
</tr>
<tr>
<td>Malt P</td>
<td>88.0</td>
<td>90.6</td>
<td>76.1</td>
<td>81.5</td>
<td>67.3</td>
</tr>
<tr>
<td>MST P</td>
<td>88.8</td>
<td>91.3</td>
<td>76.4</td>
<td>81.1</td>
<td>68.1</td>
</tr>
<tr>
<td>Berk G</td>
<td>91.6</td>
<td>93.4</td>
<td>83.1</td>
<td>86.4</td>
<td>76.8</td>
</tr>
<tr>
<td>Malt G</td>
<td>90.0</td>
<td>91.6</td>
<td>80.4</td>
<td>83.7</td>
<td>78.3</td>
</tr>
<tr>
<td>MST G</td>
<td>90.7</td>
<td>92.3</td>
<td>80.8</td>
<td>83.4</td>
<td>78.4</td>
</tr>
</tbody>
</table>

### 3.2.1 POS Tagging

Something we notice in Tables 2 and 3 is the difference in parsing accuracy between the scenario in which the parser is supplied with the correct POS tag for each word in the input string and the realistic scenarios in which it is supplied with POS tags produced by a POS tagger or in which it produces the POS tags as part of the parsing process. It is clear from this difference that a proportion of the parsing errors can be attributed to POS tagging errors, and it is also clear that this proportion is greater for the out-of-domain Web 2.0 text than it is for the in-domain WSJ text. The proportion of unknown words in the development sets already tells us something: 2.8% of the tokens in WSJ22 do not occur in WSJ2-21 compared to 6.8% for FootballDev and 16.6% for TwitterDev.\(^4\)

We look in more detail at the POS tagging errors produced by the Berkeley parser in own tagging mode, the Brown parser and SVMTool (the POS tagger used in the Malt and Berkeley pipelines). Instead of looking at the most common POS tagging errors, we attempt to locate the tagging errors that are associated with inaccurate phrase structure trees. For each POS confusion that occurs more than 5 times in the particular development set, we find the relative frequency of this confusion in sentences receiving a Parseval f-score under 70.0. We then order the POS confusions by these relative frequencies. The top-ranking confusions (gold/system) common to all three systems are as follows:

1. **WSJ22**: NNS/VBZ, VBZ/NNS
2. **FootballDev**: RB/JJ, RB/RP, VB/VBP
3. **TwitterDev**: JJ/NNP, NN/VB, NNP/VB, NNP/JJ, VBZ/NNS

The tendency for the noun/verb confusion that we see in WSJ22 and TwitterDev to affect parser accuracy has been documented before (Dalrymple, 2006). The following is a TwitterDev example:

\[
\text{(FRAG)}
\begin{align*}
\text{(NP (JJ Username))} \\
\text{(S (NP (JJ fantastic)) (VP (VB win)))} \\
\text{(. !) (. !))}
\end{align*}
\]

The **RB/JJ** confusion in FootballDev can be explained by the tendency of some posters to drop the **-ly** suffix on adverbs (e.g. **played bad**). The

\(^3\)The Brown parser makes use of non-PTB tags to mark auxiliary verbs (AUX and AUXG). We take this difference into account when calculating POS tagging accuracy.

\(^4\)The tokens **Username** and **Urlname** are unknown and occur repeatedly. But even discounting these, the ratio is 14.0%.
prominence of NNP in TwitterDev is interesting and suggests, for one thing, that less emphasis should be placed on capitalisation in the tagging of unknown words:

\[
\begin{align*}
(S & \quad \text{(NP (NNP grrr))} \\
& \quad \text{(VP (VB spotify)} \\
& \quad \text{(PP (IN in} \\
& \quad \text{(NP some kind of infinite update loop))})
\end{align*}
\]

3.2.2 Stanford Dependencies

We analyse the deprel-attachment f-scores of the best non-gold-tagged configuration of each parser. This means that for the Berkeley parser we use the version that performs its own tagging.

For most of the dependency types there is a general trend as exemplified in Figure 1: for each of the three datasets, the relation Brown > Berk > MSTParser > MaltParser holds. The WSJ22 results are around 5-10% absolute better than the FootballDev results while the drop for TwitterDev is in the 15-20% range on average. Frequent dependencies like nominal subjects (nsubj), direct objects (dobj), adverbial clauses (advcl), copulars, open complements (xcomp), prepositional modifiers follow this trend. The relations det, root, aux, pobj, poss, possessive, neg are easy to recover in all datasets, with no big drops observed for FootballDev or TwitterDev. For adjectival modifiers (amod), adverbial modifiers (advmod), and complements (complm), the decreasing pattern over the three datasets holds but the dependency parsers outperform the constituency parsers.

Coordination is one of the harder relations to recover. According to the Stanford scheme, the first conjunct is the head of the coordination. The conjunction is attached to the head via the cc relation and the other conjuncts are attached via the conj relation. For WSJ22, the phrase structure parser scores are around 85% for cc and slightly lower for conj, and the dependency parsers scores are around 80% for cc and 71% for conj. For the Web 2.0 data the scores decrease from WSJ22 to FootballDev but increase again for TwitterDev. The drop in performance for FootballDev is in line with Foster’s (2010) observation that the discussion forum data contain difficult coordination cases involving coordination of unlike constituents. It is possible that the length of the Twitter sentences acts as a natural inhibitor to such cases. We also note that Brown clearly outperforms the other parsers on TwitterDev.

4 Making Use of Unlabelled Data

One approach to the parser domain adaptation problem is to train a new system using large quantities of automatically parsed target domain text. We experiment with two retraining methods: self-training in which the training data of the parser we are attempting to adapt is augmented by adding trees produced by the same parser for the sentences in our unannotated target domain corpus, and uptraining, in which the training set of a less accurate parser is augmented with trees for the unannotated corpus sentences produced by a more accurate parser.

4.1 Charniak and Johnson Self-Training

McClosky et al. (2006) demonstrate that a WSJ-trained parser can be adapted to the fiction domains of the Brown corpus by performing a type of self-training that involves the use of the Brown parser. Their training protocol is as follows: sentences from the LA Times are parsed using the first-stage parser and reranked in the second stage. These parse trees are added to the original WSJ training set and the first-stage parser is retrained. The sentences from the target domain, in this case, Brown corpus sentences are then parsed using the newly trained first-stage parser and reranked using the original reranker, resulting in a performance jump from 85.2% to 87.8%. One of the factors that make this training protocol effective are the non-generative features in the discriminative reranker (McClosky et al., 2008), and the use of the reranker means that this method is not a pure
self-training one, but rather a type of uptraining.

We apply the procedure McClosky at el. to determine whether the performance of Brown can be improved on Web 2.0 data. The top graph in Figure 2 shows the results obtained for FootballDev when the first-stage parser is retrained on various combinations of WSJ02-21 and parse trees produced by the reranking parser for sentences in FootballTrain. The bottom graph represents the f-scores for TwitterDev using TwitterTrain instead of FootballTrain. It is clear from the top graph that adding material from FootballTrain results in a significant improvement over the baseline f-scores of 79.7. The highest f-score is 83.8, obtained using two copies of WSJ2-21 and 500,000 FootballTrain trees. The improvements achieved using TwitterTrain are less pronounced, with an absolute improvement of 2.4% obtained using 600,000 TwitterTrain trees and two copies of WSJ02-21.

4.2 Malt Uptraining

Petrov et al. (2010) perform a Stanford-dependency-based parser evaluation, with sentences from QuestionBank (Judge et al., 2006) as their test data. They find that deterministic dependency parsers such as MaltParser suffer more from the domain differences between QuestionBank and WSJ than phrase structure parsers such as the Berkeley parser. They then attempt to improve the accuracy of MaltParser on questions by training it on questions parsed by the Berkeley parser, arguing that the linear time complexity of a parser such as Malt is needed for real-time processing of web data. They demonstrate that the same improvement in accuracy can be obtained by using 100,000 automatically parsed questions as can be obtained using 2,000 manually parsed QuestionBank trees.

We perform two uptraining experiments. In the first, we retrain MaltParser using a combination of WSJ02-21 and trees produced by Brown for sentences in the FootballTrain or TwitterTrain corpora (we call this vanilla uptraining). In the second and novel approach, we use a self-trained Brown grammar to parse the trees for uptraining (we call this domain-adapted uptraining). For all configurations, the POS tagger, SVMtool, is retrained on the same data as MaltParser.

The results of the uptraining experiments show that significant improvements are obtained for both types of uptraining, but as expected, the domain-adapted uptraining is superior. The graph in Fig. 3 shows that the best FootballDev grammar is obtained using domain-adapted uptraining with 350,000 FootballTrain trees and one copy of WSJ02-21 (an improvement of 5.7% over the baseline). The corresponding Twitter graph (not shown due to lack of space) shows that an improvement of 4.6% can be obtained on TwitterDev using domain-adapted uptraining with 200,000 TwitterTrain trees and one copy of WSJ02-21.

4.3 Latent Variable Self-Training

Experiments with pure self-training, i.e. training a parser on its own output, have had mixed results over the years. Charniak (1997), Steedman et al. (2003) and Plank (2009) provide evidence that it is not effective, whereas the experiments of Reichart and Rappoport (2007), Huang and Harper (2009) and Sagae (2010) suggest that it can be useful. Huang and Harper (2009) are the first to apply self-training using a PCFG-LA — with positive results. They argue that self-training works in this scenario because the additional training data prevents the split-merge process from overfitting.

We apply the self-training method of Huang and Harper to our new datasets. Like Huang and Harper, we use our own PCFG-LA parser because the trainer is multi-threaded, allowing us to handle the computation needed to train a PCFG-LA on large corpora. We train a 6-iteration PCFG-LA using WSJ02-21 and use it to parse the FootballTrain and TwitterTrain corpora. We then add the automatically parsed material to our WSJ training set and retrain more 6-iteration PCFG-LAs. The result for FootballDev using FootballTrain is shown in Table 4. We achieve an absolute f-score improvement of 5.5% on FootballDev. The corresponding TwitterTrain graph is not shown for space reasons. The TwitterTrain improvements are more modest, with an f-score increase of approximately 2% on TwitterDev.

4.4 Test Set Results

For each of our three retraining experiments, we take the best grammar for FootballDev and the best grammar for TwitterDev and apply them to the three test sets. In the PCFG-LA self-training experiments, a FootballTrain grammar actually outperforms all TwitterTrain grammars on TwitterDev.
and so we use this for final testing. The results are provided in Table 4. Statistically significant differences between the relevant baseline are marked with an asterisk.

## 5 Discussion

**Parser retraining** The variance in the size of improvements between the development and test sets (a greater improvement for Malt uptraining and a smaller improvement for Brown self-training) and the fact that, for Brown and Malt, the best grammar on TwitterDev is outperformed on TwitterTest by the best grammar on FootballDev is most likely due to the small size of the datasets. However, the results are promising, and clearly demonstrate that unlabelled user-generated content can be used to improve parser accuracy.

The reasons for the improvements yielded by the three types of retraining need to be determined. The underperformance of the TwitterTrain material in comparison to the FootballTrain material suggests that sample selection involving language and topic identification needs to be applied before parser retraining. We also intend to test the combination of PCFG-LA self-training and product grammar parsing described in Huang et al. (2010) on our Web 2.0 dataset.

### Combination Parsing

Several successful parsing methods have employed multiple parsing models, combined using techniques such as voting, stacking and product models (Henderson and Brill, 2000; Nivre and McDonald, 2008; Petrov, 2010). An ensemble approach to parsing seems particularly appropriate for the linguistic melting pot of Web 2.0, as does the related idea of selecting a model based on characteristics of the input. For example, a preliminary error analysis of the Malt uptraining results shows that coordination cases in TwitterDev are helped more by grammars trained on FootballTrain than on TwitterTrain, suggesting that sentences containing a conjunction should be directed to a FootballTrain grammar. McClosky et al. (2010) use linear regression to determine the correct mix of training material for a particular document. We intend to experiment with this idea in the context of Web 2.0 parsing.

### Preprocessing

Foster (2010) and Gadde et al. (2011) report improved parsing and tagging performance when the input data is normalised before processing. This work employs very little data cleaning and future work will involve exploring the interaction between preprocessing and parser retraining. Hyperlinks and usernames in tweets were replaced by the terms *Urlname* and *Username* respectively — to make life easier for parsers and POS taggers, proper nouns that are in the systems’ lexicons should be used. The automatic sentence splitter and tokeniser that was used to create the Web 2.0 training sets makes use of abbreviation statistics in order to determine sentence boundaries. We compiled an abbreviation table using football discussion forum data but made no attempt to modify it for Twitter data. What is needed is a sentence-splitter tuned to the punctuation conventions of Twitter. However, more fundamental question remain: what is the correct unit of analysis for tweets and does it even make sense to talk about sentences in the context of Twitter? Our next step in this direction is to experiment with the Twitter-specific resources (tagset, tagger, tokeniser) described in Gimpel et al. (2011).

### More Datasets

We have focused on WSJ material as the source for our labelled training data. Future work will involve the use of other syntactically annotated resources including Brown and Switchboard, as well as Ontonotes 4.0, which has recently been released and which contains syntactically annotated web text (300k words).

### More Parser Evaluation

The cross-parser evaluation we have presented in the first half of the paper is by no means exhaustive. For example, to measure the positive effect of discriminative reranking, the first-stage Brown parser should also be included in the evaluation. Other statistical parsers could be evaluated, and it would be interesting to examine the performance of sys-

---

Table 4: Test Set Results

<table>
<thead>
<tr>
<th>Grammar</th>
<th>FootballTest</th>
<th>TwitterTest</th>
<th>WSJ23</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charniak and Johnson self-training (F-Score)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baseline</td>
<td>81.2</td>
<td>73.3</td>
<td>91.4</td>
</tr>
<tr>
<td>Best FootballDev</td>
<td>83.6*</td>
<td>75.0*</td>
<td>91.1*</td>
</tr>
<tr>
<td>Best TwitterDev</td>
<td>81.0</td>
<td>74.7*</td>
<td>91.1*</td>
</tr>
<tr>
<td>Latent variable PCFG self-training (F-Score)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baseline</td>
<td>77.7</td>
<td>69.5</td>
<td>89.8</td>
</tr>
<tr>
<td>Best FootballDev</td>
<td>82.4*</td>
<td>70.6*</td>
<td>89.4*</td>
</tr>
<tr>
<td>Best TwitterDev</td>
<td>81.7*</td>
<td>70.7</td>
<td>89.6</td>
</tr>
<tr>
<td>Malt uptraining (LAS)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baseline</td>
<td>71.8</td>
<td>64.1</td>
<td>87.7</td>
</tr>
<tr>
<td>Best FootballDev</td>
<td>80.6*</td>
<td>69.7*</td>
<td>86.5*</td>
</tr>
<tr>
<td>Best TwitterDev</td>
<td>76.7*</td>
<td>68.2*</td>
<td>87.1*</td>
</tr>
</tbody>
</table>

---

See Foster et al. (2011) for a preliminary analysis of the effect of Malt uptraining on sentences from TwitterDev.
tems which employ hand-crafted grammars and treebank-trained disambiguators in order to determine whether a system less tuned to the PTB is more appropriate for this kind of heterogeneous data (Plank and van Noord, 2010). We have employed the Stanford dependencies in this work — other labelled dependency schemes are available and it might be informative to examine the relative performance of the parsers from the perspective of many such schemes rather than just one. Gold standard dependency annotations for the new sentences would also be a bonus.

Acknowledgements

This research has been supported by Enterprise Ireland (CFTD/2007/229) and by Science Foundation Ireland (Grant 07/CE/11142) as part of the CNGL (www.cngl.ie) at School of Computing, DCU, and by the French Agence Nationale pour la Recherche, through the SEQUOIA project (ANR-08-EMER-013). We thank the reviewers for their helpful comments.

References


Roi Reichart and Ari Rappaport. 2007. Self-training for enhancement and domain adaptation of statistical parsers trained on small datasets. In Proceedings of ACL.


Figure 3: Malt uptraining LAS results for *FootballDev*: V stands for Vanilla Uptraining and D for Domain-Adapted Uptraining

Figure 4: PCFG-LA self-training results