# A Multi-Domain Web-Based Algorithm for POS Tagging of Unknown Words

**Shulamit Umansky-Pesin**
Institute of computer science
The Hebrew University
pesin@cs.huji.ac.il

**Roi Reichart**
ICNC
The Hebrew University
roiri@cs.huji.ac.il

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

## Abstract

We present a web-based algorithm for the task of POS tagging of *unknown words* (words appearing only a small number of times in the training data of a supervised POS tagger). When a sentence $s$ containing an unknown word $u$ is to be tagged by a trained POS tagger, our algorithm collects from the web contexts that are partially similar to the context of $u$ in $s$, which are then used to compute new tag assignment probabilities for $u$. Our algorithm enables fast multi-domain unknown word tagging, since, unlike previous work, it does not require a corpus from the new domain. We integrate our algorithm into the MXPOST POS tagger (Ratnaparkhi, 1996) and experiment with three languages (English, German and Chinese) in seven in-domain and domain adaptation scenarios. Our algorithm provides an error reduction of up to 15.63% (English), 18.09% (German) and 13.57% (Chinese) over the original tagger.

## 1 Introduction

Part-of-speech (POS) tagging is a fundamental NLP task that has attracted much research in the last decades. While supervised POS taggers have achieved high accuracy (e.g., (Toutanova et al., 2003) report a 97.24% accuracy in the WSJ Penn Treebank), tagger performance on words appearing a small number of times in their training corpus (*unknown words*) is substantially lower. This effect is especially pronounced in the *domain adaptation* scenario, where the training and test corpora are from different domains. For example, when training the MXPOST POS tagger (Ratnaparkhi, 1996) on sections 2-21 of the WSJ Penn Treebank it achieves 97.04% overall accuracy when tested on WSJ section 24, and 88.81% overall accuracy when tested on the BNC corpus, which contains texts from various genres. For unknown words (test corpus words appearing 8 times or less in the training corpus), accuracy drops to 89.45% and 70.25% respectively.

In this paper we propose an unknown word POS tagging algorithm based on web queries. When a new sentence $s$ containing an unknown word $u$ is to be tagged by a trained POS tagger, our algorithm collects from the web contexts that are partially similar to the context of $u$ in $s$. The collected contexts are used to compute new tag assignment probabilities for $u$.

Our algorithm is particularly suitable for *multi-domain* tagging, since it requires no information about the domain from which the sentence to be tagged is drawn. It does not need domain specific corpora or external dictionaries, and it requires no preprocessing step. The information required for tagging an unknown word is very quickly collected from the web.

This behavior is unlike previous works for the task (e.g (Blitzer et al., 2006)), which require a time consuming preprocessing step and a corpus collected from the target domain. When the target domain is heterogeneous (as is the web itself), a corpus representing it is very hard to assemble. To the best of our knowledge, ours is the first paper to provide such an *on-the-fly* unknown word tagging algorithm.

To demonstrate the power of our algorithm as a

*Coling 2008: Poster Volume*, pages 1274–1282,
Beijing, August 2010

fast multi-domain learner, we experiment in three languages (English, German and Chinese) and several domains. We implemented the MXPOST tagger and integrated it with our algorithm. We show error reduction in unknown word tagging of up to 15.63% (English), 18.09% (German) and 13.57% (Chinese) over MXPOST. The run time overhead is less than 0.5 seconds per an unknown word in the English and German experiments, and less than a second per unknown word in the Chinese experiments.

Section 2 reviews previous work on unknown word Tagging. Section 3 describes our web-query based algorithm. Section 4 and Section 5 describe experimental setup and results.

## 2 Previous Work

Most supervised POS tagging works address the issue of unknown words. While the general methods of POS tagging vary from study to study – Maximum Entropy (Ratnaparkhi, 1996), conditional random fields (Lafferty et al., 2001), perceptron (Collins, 2002), Bidirectional Dependency Network (Toutanova et al., 2003) – the treatment of unknown words is more homogeneous and is generally based on additional features used in the tagging of the unknown word.

Brants (2000) used only suffix features. Ratnaparkhi (1996) used orthographical data such as suffixes, prefixes, capital first letters and hyphens, combined with a local context of the word. In this paper we show that we improve upon this method. Toutanova and Manning (2000), Toutanova et al. (2003), Lafferty et al. (2001) and Vadas and Curran (2005) used additional language-specific morphological or syntactic features. Huihsin et al. (2005) combined orthographical and morphological features with external dictionaries. Nakagawa and Matsumoto (2006) used global and local information by considering interactions between POS tags of unknown words with the same lexical form.

Unknown word tagging has also been explored in the context of domain adaptation of POS taggers. In this context two directions were explored: a supervised method that requires a manually annotated corpus from the target domain (Daume III, 2007), and a semi-supervised method that uses an unlabeled corpus from the target domain (Blitzer et al., 2006).

Both methods require the preparation of a corpus of target domain sentences and re-training the learning algorithm. Blitzer et al. (2006) used 100K unlabeled sentences from the WSJ (source) domain as well as 200K unlabeled sentences from the biological (target) domain. Daume III (2007) used an 11K words labeled corpus from the target domain.

There are two serious problems with these approaches. First, it is not always realistically possible to prepare a corpus representing the target domain, for example when that domain is the web (e.g., when the POS tagger serves an application working on web text). Second, preparing a corpus is time consuming, especially when it needs to be manually annotated. Our algorithm requires no corpus from the target data domain, no preprocessing step, and it doesn't even need to know the identity of the target domain. Consequently, the problem we address here is more difficult (and arguably more useful) than that addressed in previous work[1].

The domain adaptation techniques above have not been applied to languages other than English, while our algorithm is shown to perform well in seven scenarios in three languages.

Qiu et al. (2008) explored Chinese unknown word POS tagging using internal component and contextual features. Their work is not directly comparable to ours since they did not test a domain adaptation scenario, and used substantially different corpora and evaluation measures in their experiments.

Numerous works utilized web resources for NLP tasks. Most of them collected corpora using data mining techniques and used them offline. For example, Keller et al., (2002) and Keller and Lapata (2003) described a method to obtain frequencies for unseen adjective-noun, noun-noun and verb-object bigrams from the web by query-

[1] We did follow their experimental procedure as much as we could. Like (Blitzer et al., 2006), we compare our algorithm to the performance of the MXPOST tagger trained on sections 2-21 of WSJ. Like both papers, we experimented in domain adaptation from WSJ to a biological domain. We used the freely available Genia corpus, while they used data from the Penn BioIE project (PennBioIE, 2005).

ing a Web engine.

On-line usage of web queries is less frequent and was used mainly in semantic acquisition applications: the discovery of semantic verb relations (Chklovski and Pantel, 2004), the acquisition of entailment relations (Szpektor et al., 2004), and the discovery of concept-specific relationships (Davidov et al., 2007). Chen et al. (2007) used web queries to suggest spelling corrections.

Our work is related to self-training (McClosky et al., 2006a; Reichart and Rappoport, 2007) as the algorithm used its own tagging of the sentences collected from the web in order to produce a better final tagging. Unlike most self-training works, our algorithm is not re-trained using the collected data but utilizes it at test time. Moreover, unlike in these works, in this work the data is collected from the web and is used only during unknown words tagging. Interestingly, previous works did not succeed in improving POS tagging performance using self-training (Clark et al., 2003).

## 3 The Algorithm

Our algorithm utilizes the correlation between the POS of a word and the contexts in which the word appears. When tackling an unknown word, the algorithm searches the web to find contexts similar to the one in which the word appears in the sentence. A new tag assignment is then computed for the unknown word based on the extracted contexts as well as the original ones.

We start with a description of the web-based context searching algorithm. We then describe how we combine the context information collected by our algorithm with the statistics of the MXPOST tagger. While in this paper we implemented this tagger and used it in our experiments, the context information collected by our web-query based algorithm can be integrated into any POS tagger.

### 3.1 Web-Query Based Context Collection

An unknown word usually appears in a given sentence with other words on its left and on its right. We use three types of contexts. The first includes all of these neighboring words, the second includes the words on the left, and the third includes the words on the right.

For each context type we define a web query using two common features supported by the major search engines: wild-card search, expressed using the '*' character, and exact sentence search, expressed by quoted characters. The retrieved sentences contain the parts enclosed in quotes in the exact same place they appear in the query, while an asterisk can be replaced by any single word.

For a word $u$ we execute the following three queries for each of its test contexts:

1. **Replacement**: "$u_{-2} u_{-1} * u_{+1} u_{+2}$". This retrieves words that appear in the same context as $u$.

2. **Left-side**: "$* \ * \ u \ u_{+1} \ u_{+2}$". This retrieves alternative left-side contexts for the word $u$ and its original right-side context.

3. **Right-side**: query "$u_{-2} \ u_{-1} \ u \ * \ *$". This retrieves alternative right-side contexts for $u$ and its original left-side context.

| Query Type | Query | Matches (Counts) |
|---|---|---|
| Replacement | "irradiation and * treatment of" | heat (15)<br>chemical (7)<br>the (6)<br>radiation (1)<br>pressure (1) |
| Left-side | "* * H2O2 treatment of" | by an (9)<br>indicated that (5)<br>enhanced by (4)<br>familiar with (3)<br>observed after (3) |
| Right-side | "irradiation and H2O2 * *" | in comparison (3)<br>on Fe (1)<br>treatment by (1)<br>cause an (1)<br>does not (1) |

Table 1: Top 5 matches of each query type for the word 'H2O2' in the GENIA sentence: "UV irradiation and H2O2 treatment of T lymphocytes induce protein tyrosine phosphorylation and Ca2+ signals similar to those observed following biological stimulation.". For each query the matched words (matches) are ranked by the number of times they occur in the query results (counts).

An example is given in Table 1, presenting the top 5 matches of every query type for the word 'H2O2', which does not appear in the English WSJ corpus, in a sentence taken from the English Genia corpus. Since matching words can appear multiple times in the results, the algorithm maintains for each match a counter denoting the number of times it appeared in the results, and sorts the results according to this number.

Seeing the table, readers might think of the following algorithm: take the leading match in the Replacement query, and tag the unknown word using its most frequent tag (assuming it is a known word). We have experimented with this method, and it turned out that its results are worse than those given by MXPOST, which we use as a baseline.

The web queries are executed by Yahoo! BOSS[2], and the resulting XML containing up to a 1000 results (a limit set by BOSS) is processed for matches. A list of matches is extracted from the *abstract* and *title* nodes of the web results along with counts of the number of times they appear. The matches are filtered to include only known words (words that appear in the training data of the POS tagger more than a threshold) and to exclude the original word or context.

Our algorithm uses a positive integer parameter $N_{web}$: only the $N_{web}$ top-scoring unique results of each query type are used for tagging. If a left-side or right-side query returns less than $N_{web}$ results, the algorithm performs a 'reduced' query: "$* \; * \; u \; u_{+1}$" for left-side and "$u_{-1} \; u \; * \; *$" for the right side. These queries should produce more results than the original ones due to the reduced context. If these reduced queries do not produce $N_{web}$ results, the web query algorithm is not used to assist the tagger for the unknown word $u$ at hand. If a replacement query does not produce at least $N_{web}$ unique results, only the left-side and right-side queries are used.

For Chinese queries, search engines do their own word segmentation so the semantics of the '*' operator is supposedly the same as for English and German. However, the answer returned by the search engine does not provide this segmentation. To obtain the words filling the '*' slots in our queries, we take all possible segmentations in which the two words appears in our training data.

The queries we use in our algorithm are not the only possible ones. For example, a possible query we do not use for the word $u$ is "$**u_{-1}uu_{+1}u_{+2}$". The aforementioned set of queries gave the best results in our English, German and Chinese development data and is therefore the one we used.

### 3.2 Final Tagging

**The MXPOST Tagger.** We integrated our algorithm into the maximum entropy tagger of (Ratnaparkhi, 1996). The tagger uses a set $h$ of contexts ('history') for each word $w_i$ (the index $i$ is used to allow an easy notation of the previous and next words, whose lexemes and POS tags are used as features). For each such word, the tagger computes the following conditional probability for the tag $t_r$:

$$p(t_r|h) = \frac{p(h,t_r)}{\sum_{t'_r \in T} p(h,t'_r)} \qquad (1)$$

where $T$ is the tag set, and the denominator is simply $p(h)$. The joint probability of a history $h$ and a tag $t$ is defined by:

$$p(h,t) = Z \prod_{j=1}^{k} \alpha_j^{f_j(h,t)} \qquad (2)$$

where $\alpha_1, \ldots, \alpha_k$ are the model parameters, $f_1, \ldots, f_k$ are the model's binary features (indicator functions), and $Z$ is a normalization term for ensuring that $p(h,t)$ is a probability.

In the training phase the algorithm performs maximum likelihood estimation for the $\alpha$ parameters. These parameters are then used when the model tags a new sentence (the test phase). For words that appear 5 times or less in the training data, the tagger extracts special features based on the morphological properties of the word.

**Combining Models.** In general, we use the same equation as MXPOST to compute joint probabilities, and our training phase is identical to its training phase. What we change are two things. First, we add new contexts to the 'history' of a word when it is considered as unknown (so Equation (2) is computed using different histories). Second, we use a different equation for computing the conditional probability (below).

When the algorithm encounters an unknown word $w_i$ in the context $h$ during tagging, it performs the web queries defined in Section 3.1. For

[2] http://developer.yahoo.com/search/boss/

each of the $N_{web}$ top resulting matches for each query, $\{h'_n | n \in [1, N_{web}]\}$, the algorithm creates its corresponding history representation $h_n$. Converting $h'_n$ to $h_n$ is required since in MXPOST a history consists of an ordered set of words together with their POS tags, while $h'_n$ is an ordered set of words without POS tags. Consequently, we define $h_n$ to consist of the same ordered set of words as $h'_n$, and we tag each word using its most frequent POS tag in the training corpus. If $w_{i-1}$ or $w_{i-2}$ are unknown words, we do not tag them, letting MXPOST use its back-off technique for such a case (which is simply to compute the features that it can and ignore those it cannot).

For each possible tag $t \in T$, its final assignment probability to $w_i$ is computed as an average between its probability given the various contexts:

$$\tilde{p}(t_r|h) = \frac{p_{org}(t_r|h) + \sum_{n=1}^{QN_{web}} p_n(t_r|h_n)}{QN_{web} + 1} \quad (3)$$

where $Q$ is the number of query types used (1, 2 or 3, see Section 3.1).

During inference, we use the two search space constraints applied by the original MXPOST. First, we apply a beam search procedure that considers the 10 most probable different tag sequences of the tagged sentence at any point in the tagging process. Second, known words are constrained to be annotated only by tags with which they appear in the training corpus.

## 4 Experimental Setup

**Languages and Datasets.** We experimented with three languages, English, German and Chinese, in various combinations of training and testing domains (see Table 2). For English we used the Penn Treebank WSJ corpus (WSJ) (Marcus et al., 1993) from the economics newspapers domain, the GENIA corpus version 3.02p (GENIA) (Kim et al., 2003) from the biological domain and the British National Corpus version 3 (BNC) (Burnard, 2000) consisting of various genres. For German we used two different corpora from the newspapers domain: NEGRA (Brants, 1997) and TIGER (Brants et al., 2002). For Chinese we used the Penn Chinese Treebank corpus version 5.0 (CTB) (Xue et al., 2002).

All corpora except of WSJ were split using random sampling. For the NEGRA and TIGER corpora we used the Stuttgart-Tuebingen Tagset (STTS).

According to the annotation policy of the GENIA corpus, only the names of journals, authors, research institutes, and initials of patients are annotated by the 'NNP' (Proper Name) tag. Other proper names such as general people names, technical terms (e.g. 'Epstein-Barr virus') genes, proteins, etc. are tagged by other noun tags ('NN' or 'NNS'). This is in contrast to the WSJ corpus, in which every proper name is tagged by the 'NNP' tag. We therefore omitted cases where 'NNP' is replaced by another noun tag from the accuracy computation of the GENIA domain adaptation scenario (see analysis in (Lease and Charniak, 2005)).

In all experimental setups except of WSJ-BNC the training and test corpora are tagged with the same POS tag set. In order to evaluate the WSJ-BNC setup, we converted the BNC tagset to the Penn Treebank tagset using the comparison table provided in (Manning and Schuetze, 1999) (pages 141–142).

**Baseline.** As a baseline we implemented the MXPOST tagger. An executable code for MXPOST written by its author is available on the internet, but we needed to re-implement it in order to integrate our technique. We made sure that our implementation does not degrade results by running it on our WSJ scenario (see Table 2), which is very close to the scenario reported in (Ratnaparkhi, 1996). The accuracy of our implementation is 97.04%, a bit better than the numbers reported in (Ratnaparkhi, 1996) for a WSJ scenario using different sections.

**Parameter Tuning.** We ran experiments with three values of the unknown word threshold $T$: 0 (only words that do not appear in the training data are considered unknown), 5 and 8. That is, the algorithm performs the web context queries and utilizes the tag probabilities of equation 3 for words that appear up to 0 ,5 or 8 times in the training data.

Our algorithm has one free parameter $N_{web}$, the number of query results for each context type used

| Language | Expe. name | Training | Development | Test |
|---|---|---|---|---|
| English | WSJ | sections 2-21 (WSJ) | section 22 (WSJ) | section 23 (WSJ)<br>(2.4%,6.7%,8.4%) |
| English | WSJ-BNC | sections 2-21 (WSJ) | 2000 BNC sentence | 2000 BNC sentences<br>(8.4%,14.9%,17%) |
| English | WSJ-GENIA | WSJ sections 2-21 | 2000 GENIA sentences | 2000 GENIA sentences<br>(22.7%,30.65%,32.9%) |
| German | NEGRA | 15689 NEGRA sentences | 1746 NEGRA sentences | 2096 NEGRA sentences<br>(11.1%,24.7%,28.7%) |
| German | NEGRA-TIGER | 15689 NEGRA sentences | 2000 TIGER sentences | 2000 TIGER sentences<br>(16%,27.3%,30.6%) |
| German | TIGER-NEGRA | 15689 TIGER sentences | 1746 NEGRA sentences | 2096 NEGRA sentence<br>(16.2%,27.9%,31.6%) |
| Chinese | CTB | 14903 CTB sentences | 1924 CTB sentences | 1945 CTB senteces<br>(7.4%,15.7%,18.1%) |

Table 2: Details of the experimental setups. In the 'Test' column the numbers in parentheses are the fraction of the test corpus words that are considered unknown, when the unknown word threshold is set to 0, 5 and 8 respectively.

|  | $T = 0$ |  |  | $T = 5$ |  |  | $T = 8$ |  |  |
|---|---|---|---|---|---|---|---|---|---|
|  | WSJ | WSJ-BNC | WSJ-GENIA | WSJ | WSJ-BNC | WSJ-GENIA | WSJ | WSJ-BNC | WSJ-GENIA |
| Baseline | 83.56 | 61.22 | 80.05 | 88.79 | 68.71 | 80.12 | 89.45 | 70.25 | 80.8 |
| Unlimited (-) | **84.85** | 63.51 | 82.50 | **89.86** | 71.12 | 82.51 | **90.47** | 72.77 | 83.16 |
| Top 5 (-) | 84.25 | 64.24 | 82.75 | 89.73 | 71.21 | 82.78 | 90.36 | 72.74 | 83.46 |
| Top 10 (-) | 84.42 | 64.10 | **83.17** | 89.70 | 71.36 | **83.00** | 90.29 | 72.87 | **83.70** |
| Top 10 (+) | 84.67 | **64.47** | 82.60 | 89.83 | **72.12** | 82.54 | 90.29 | **73.53** | 83.22 |
| best imp. | 1.19<br>7.23% | 3.25<br>8.38% | 3.12<br>15.63% | 1.07<br>9.54% | 3.41<br>10.89% | 2.88<br>14.48% | 1.02<br>9.66% | 3.28<br>11.02% | 2.9<br>15.1% |

|  | $T = 0$ |  |  | $T = 5$ |  |  | $T = 8$ |  |  |
|---|---|---|---|---|---|---|---|---|---|
|  | NEGRA | NEGRA-TIGER | TIGER-NEGRA | NEGRA | NEGRA-TIGER | TIGER-NEGRA | NEGRA | NEGRA-TIGER | TIGER-NEGRA |
| Baseline | 90.26 | 85.71 | 87.18 | 91.06 | 87.88 | 87.86 | 91.45 | 88.22 | 88.18 |
| Unlimited (-) | 91.22 | 86.60 | 89.49 | 91.66 | 88.22 | **89.84** | 92.25 | 89.08 | 90.23 |
| Top 5 (-) | **91.41** | 86.68 | 89.32 | **91.95** | **89.01** | 89.72 | **92.38** | 89.33 | 90.26 |
| Top 10 (-) | 91.06 | 86.83 | **89.50** | 91.25 | 88.36 | **89.84** | 92.33 | **89.38** | **90.26** |
| Top 10 (+) | 90.58 | **86.86** | 89.43 | 91.25 | 88.36 | **89.84** | 91.53 | 88.35 | 89.71 |
| best imp. | 1.15<br>11.8% | 1.15<br>8.04% | 2.32<br>18.09% | 0.89<br>9.95% | 1.13<br>9.32% | 1.98<br>16.3% | 0.93<br>10.87% | 1.16<br>9.84% | 2.08<br>17.59% |

|  | CTB |  |  |
|---|---|---|---|
|  | $T = 0$ | $T = 5$ | $T = 8$ |
| Baseline | 74.99 | 78.03 | 79.81 |
| Unlimited (-) | 77.01 | 80.46 | 81.94 |
| Top 5 (-) | **77.58** | **80.75** | 82.19 |
| Top 10 (-) | 77.43 | 80.68 | 82.45 |
| Top 10 (+) | 77.43 | 80.68 | **82.35** |
| best imp. | 2.59<br>10.35% | 2.72<br>12.28% | 2.74<br>13.57% |

Table 3: Accuracy of unknown word tagging in the English (top table), German (middle table) and Chinese (bottom table) experiments. Results are presented for three values of the unknown word threshold parameter $T$: 0, 5 and 8. For all setups our models improves over the MXPOST baseline of (Ratnaparkhi, 1996). The bottom line of each table ('best imp.') presents the improvement (top number) and error reduction (bottom number) of the best performing model over the baseline. The best improvement is in domain adaptation scenarios.

in the probability computation of equation 3. For each setup (Table 2) we ran several combinations of query types and values of $N_{web}$. We report results for the four leading combinations:

- $N_{web} = 5$, left-side and right-side queries (Top 5 (-)).
- $N_{web} = 10$, left-side and right-side queries (Top 10 (-)).
- $N_{web} = 10$, replacement, left-side and right-side queries (Top 10 (+)).
- $N_{web} =$ Unlimited (in practice, this means 1000, the maximum number of results provided by Yahoo! Boss), left-side and right-side queries (Unlimited (-) ).

The order of the models with respect to their performance was identical for the development and test data. That is, the best parameter/queries combination for each scenario can be selected using the development data. We experimented with other parameter/queries combinations and additional query types but got worse results.

## 5 Results

The results of the experiments are shown in Table 3. Our algorithm improves the accuracy of the MXPOST tagger for all three languages and for all values of the unknown word parameter.

Our experimental scenarios consist of three in-domain setups in which the model is trained and tested on the same corpus (the WSJ, NEGRA and CTB experiments), and four domain adaptation setups: WSJ-GENIA, WSJ-BNC, TIGER-NEGRA and NEGRA-TIGER.

Table 3 shows that our model is relatively more effective in the domain adaptation scenarios. While in the in-domain setups the error reduction values are 7.23% – 9.66% (English), 9.95% – 11.8% (German) and 10.35% – 13.57% (Chinese), in the domain adaptation scenarios they are 8.38% – 11.02% (WSJ-BNC), 14.48% – 15.63% (WSJ-GENIA), 8.04% – 9.84% (NEGRA-TIGER) and 16.3% – 18.09% (TIGER-NEGRA).

**Run Time.** As opposed to previous approaches to unknown word tagging (Blitzer et al., 2006; Daume III, 2007), our algorithm does not contain a step in which the base tagger is re-trained with a corpus collected from the target domain. Instead, when an unknown word is tackled at test time, a set of web queries is run. This is an advantage for flexible multi-domain POS tagging because preprocessing times are minimized, but might cause an issue of overhead per test word.

To show that the run time overhead created by our algorithm is small, we measured its time performance (using an Intel Xeon 3.06GHz, 3GB RAM computer). The average time it took the best configuration of our algorithm to process an unknown word and the resulting total addition to the run time of the base tagger are given in Table 4. The average time added to an unknown word tagging is less than half a second for English, even less for German, and less than a second for Chinese. This is acceptable for interactive applications that need to examine a given sentence without being provided with any knowledge about its domain.

**Error Analysis.** In what follows we try to analyze the cases in which our algorithm is most effective and the cases where further work is still required. Due to space limitations we focus only on the (Top 10 (+), $T = 5$) parameters setting, and report the patterns for one English setup. The corresponding patterns of the other parameter settings, languages and setups are similar.

We report the errors of the base tagger that our algorithm most usually fixes and the errors that our algorithm fails to fix. We describe the base tagger errors of the type 'POS tag 'a' is replaced with POS tag 'b' (denoted by: a -> b)' using the following data: (1) total number of unknown words whose correct tag is 'a' that were assigned 'b' by the base tagger; (2) the percentage of unknown words whose correct tag is 'a' that were assigned 'b' by the base tagger; (3) the percentage of unknown words whose correct tag is 'a' that were assigned 'b' by our algorithm; (4) the percentage of mistakes of type (1) that were corrected by our algorithm.

In the English WSJ-BNC setup, the base tagger mistakes that our algorithm handles well (according to the percentage of corrected mistakes) are: (1) NNS -> VBZ (23, 3.73%, 0.8%, 65.2%); (2) CD -> JJ (19 ,13.2% ,9.7% ,37.5%); (3) NN ->

| | WSJ | WSJ-BNC | WSJ-GENIA | NEGRA | NEGRA-TIGER | TIGER-NEGRA | CTB |
|---|---|---|---|---|---|---|---|
| Total addition | 00:28:26 | 00:31:53 | 1:37:32 | 00:57:03 | 00:19:10 | 00:36:54 | 2:29:13 |
| Avg. time per word | 0.42 | 0.32 | 0.33 | 0.36 | 0.11 | 0.21 | 0.95 |

Table 4: The processing time added by the web based algorithm to the base tagger. For each setup results are presented for the best performing model and for the unknown word threshold of 8. Results for the other models and threshold parameters are very similar. The top line presents the total time added in the tagging of the full test data (hours:minutes:seconds). The bottom line presents the average processing time of an unknown word by the web based algorithm (in seconds).

JJ (97, 6.17%, 5.3%, 27.8%); (4) JJ -> NN (69, 9.73%, 7.76%, 33.3%). The errors that were not handled well by our algorithm are: (1) IN -> JJ (70, 46.36% , 41%, 8.57%); (2) VBP -> NN (25, 19.5%, 21.9% , 0%).

In this setup, 'CD' is a cardinal number, 'IN' is a preposition, 'JJ' is an adjective, 'NN' is a noun (singular or mass), 'NNS' is a plural noun, 'VBP' is a verb in non-third person singular present tense and 'VBZ' is a verb in third person, singular present tense.

We can see that no single factor is responsible for the improvement over the baseline. Rather, it is due to correcting many errors of different types. The same general behavior is exhibited in the other setups for all languages.

**Multiple Unknown Words.** Our method is capable of handling sentences containing several unknown words. Query results in which '*' is replaced by an unknown word are filtered. For queries in which an unknown word appears as part of the query (when it is one of the two right or left non-'*' words), we let MXPOST invoke its own unknown word heuristics if needed[3].

In fact, the relative improvement of our algorithm over the baseline is *better* for adjacent unknown words than for single words. For example, consider a sequence of consecutive unknown words as correctly tagged if all of its words are assigned their correct tag. In the WSJ-GENIA scenario (Top 10 (+), $T = 5$), the error reduction for sequences of length 1 (unknown words surrounded by known words, 8767 sequences) is 8.26%, while for 2-words (2620 sequences) and 3-words (614 sequences) it is 11.26% and 19.11% respectively. Similarly, for TIGER-NEGRA (same parameters setting) the error reduction is 6.85%, 8.07% and 18.18% for sequences of length 1 (4819) ,2 (1126) and 3 (223) respectively.

## 6 Conclusions and Future Work

We presented a web-based algorithm for POS tagging of unknown words. When an unknown word is tackled at test time, our algorithm collects web contexts of this word that are then used to improve the tag probability computations of the POS tagger.

In our experiments we used our algorithm to enhance the unknown word tagging quality of the MXPOST tagger (Ratnaparkhi, 1996), a leading state-of-the-art tagger, which we implemented for this purpose. We showed significant improvement (error reduction of up to 18.09%) for three languages (English, German and Chinese) in seven experimental setups. Our algorithm is especially effective in domain-adaptation scenarios where the training and test data are from different domains.

Our algorithm is fast (requires less than a second for processing an unknown word) and can handle test sentences coming from any desired unknown domain without the costs involved in collecting domain-specific corpora and retraining the tagger. These properties makes it particularly appropriate for applications that work on the web, which is highly heterogeneous.

In future work we intend to integrate our algorithm with additional POS taggers, experiment with additional corpora and domains, and improve our context extraction mechanism so that our algorithm will be able to fix more error types.

## References

Blitzer, John, Ryan McDonald, and Fernando Pereira, 2006. Domain adaptation with structural correspon-

[3] They are needed only if the word is on the left of the word to be tagged.

dence learning. *EMNLP '06*.

Brants, Sabine, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius and George Smith, 2002. The TIGER Treebank. *Proceedings of the Workshop on Treebanks and Linguistic Theories*.

Brants, Thorsten, 1997. The NEGRA Export Format. *CLAUS Report, Saarland University*.

Brants, Thorsten, 2000. Tnt: a statistical part-of-speech tagger. In *The Sixth Conference on Applied Natural Language Processing*.

Burnard, Lou, 2000. *The British National Corpus User Reference Guide*. Technical Report, Oxford University.

Chen, Qing, Mu Li, and Ming Zhou. 2007. Improving query spelling correction using web search results. In *EMNLP-CoNLL '07*.

Chklovski, Timothy and Patrick Pantel. 2004. Verbocean: Mining the web for fine-grained semantic verb relations. *EMNLP '04*.

Clark, Stephen, James Curran and Miles Osborne. 2003. Bootstrapping POS-taggers using unlabeled data. *CoNLL '03*.

Collins, Michael. 2002. Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. *EMNLP '02*.

Daume III, Hal. 2007. Frustratingly easy domain adaptation. *ACL '07*.

Davidov, Dmitry, Ari Rappoport, and Moshe Koppel. 2007. Fully unsupervised discovery of concept-specific relationships by web mining. *ACL '07*.

Huihsin, Tseng, Daniel Jurafsky, and Christopher Manning. 2005. Morphological features help pos tagging of unknown words across language varieties. *The Fourth SIGHAN Workshop on Chinese Language Processing*.

Jin–Dong Kim, Tomoko Ohta, Yuka Teteisi and Jun'ichi Tsujii, 2003. GENIA corpus – a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19:i180–i182, Oxford University Press, 2003.

Lafferty, John D., Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *The Eighteenth International Conference on Machine Learning*.

Keller, Frank, Mirella Lapata, and Olga Ourioupina. 2002. Using the Web to Overcome Data Sparseness. *EMNLP '02*.

Keller, Frank, Mirella Lapata. 2003. . *Computational Linguistics*, 29(3):459–484.

Lease, Matthew and Eugene Charniak. 2005. Parsing Biomedical Literature. *Proceedings of the Second International Joint Conference on Natural Language Processing.*.

Manning Chris and Hinrich Schuetze. 1999. Foundations of Statistical Natural Language Processing. *MIT Press.*.

Marcus, Mitchell P., Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330.

McClosky, David, Eugene Charniak, and Mark Johnson, 2006a. Effective self-training for parsing. *HLT-NAACL '06*.

Nakagawa, Tetsuji and Yuji Matsumoto. 2006. Guessing parts-of-speech of unknown words using global information. *ACL-COLING '06* .

PennBioIE. 2005. Mining the Bibliome Project.. *http://bioie.ldc.upenn.edu*.

Qiu, Likun, Changjian Hu and Kai Zhao. 2008. A method for automatic POS guessing of Chinese unknown words. *COLING '08*.

Ratnaparkhi, Adwait. 1996. A maximum entropy model for part-of-speech tagging. *EMNLP '96*.

Reichart, Roi and Ari Rappoport. 2007. Self-Training for Enhancement and Domain Adaptation of Statistical Parsers Trained on Small Datasets. *ACL '07*.

Reynolds, Sheila M. and Jeff A. Bilmes. 2005. Part-of-speech tagging using virtual evidence and negative training. *EMNLP '06*.

Szpektor, Idan, Hristo Tanev, Ido Dagan, and Bonaventura Coppola. 2004. Scaling web-based acquisition of entailment relations. *EMNLP '04*.

Toutanova, Kristina and Christopher D. Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. *EMNLP '00*.

Toutanova, Kristina, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. *NAACL '03*.

Vadas, David and James R. Curran. 2005. Tagging unknown words with raw text features. *Australasian Language Technology Workshop 2005*.

Nianwen Xue, Fu-Dong Chiou and Martha Palmer, 2002. Building a large–scale annotated Chinese corpus. *ACL '02*.