An Investigation into Improving Part-of-Speech Tagging

Markus Dickinson
Department of Linguistics
Georgetown University
`mad87@georgetown.edu`

Abstract

We develop a method to improve POS tagging, which attempts to account for problematic ambiguities by redefining the tagset. Hand evaluating the tagger-benchmark disagreements shows us the profound effect errors have on reported accuracies, and we also explore the effect of correcting training data errors. Our results emphasize the need to focus on particular tagging problems in evaluation.

1 Introduction

It seems that part-of-speech (POS) tagging has hit a bit of an upper limit in accuracy; if we compare the state-of-the-art in 1996, the 96.63% of Ratnaparkhi (1996), with the current 97.24% (Toutanova et al., 2003), we find a gain of only 0.61% over ten years. This seems to support the claim in Church (1992) that there is an approximate 97% upper bound to tagging. However, as Marquez et al. (2000) point out, 3% error is still one error in roughly every 30 words, and if the POS processing is input into syntactic parsing, these errors can propagate to higher levels; as Dalrymple (to appear) points out, correct tagging can reduce parsing ambiguity by 45-50%. Furthermore, as shown by Padro and Marquez (1998), we do not know a tagger's true accuracy, due to errors in corpus evaluation data, and so it is not clear whether we have indeed witnessed a true improvement or not. To address these issues, we can perform more manual evaluation, in order to determine the true increase or decrease in accuracy, and we can attempt to develop models which specifically deal with the difficult cases that make up the lingering 3% of tagger-benchmark disagreements.

One way to identify the difficult cases is to examine the effect of the tagset on the tagger. The success of a tagger depends on which distinctions it learns (cf. Déjean, 2000), and if there is a way to learn better distinctions on the fly, then tagging algorithms can be improved by learning those distinctions (MacKinlay, 2005), in a way which is independent of the learning algorithm involved.

Thus, we here investigate one particular method for altering a tagset to better reflect problematic distinctions in the data, and we perform manual, as well as automatic, evaluation. We base our work on a method of POS annotation error correction designed to handle problematic distinctions, which we describe in the section 2 before turning to our POS tagging model in section 3. Examining the results as compared to the benchmark, we see overall improvement and discuss the improvement of individual tags. In section 4, however, we investigate how the interpretation of the results is affected by errors in the evaluation data and also what impact training data errors have on the accuracy of particular tags. Throughout, we emphasize the need for more qualitative evaluation to determine how to improve tagging.

2 POS Error Correction

In Dickinson (2006) we develop a tagging method to correct POS annotation errors, using a modified tagset instead of a modified tagging algorithm. The method is based on the idea that knowing the problematic distinction for a given corpus position can assist in tagging it.

To describe this method, we need to discuss the nature of tagging guidelines. Like other part-of-speech annotation guidelines (e.g., Wynne, 1996), Santorini (1990) provides a list of "confusing parts of speech," or "difficult tagging distinctions" for the Penn Treebank. For example, the manual discusses the distinction between preposition (IN) and particle (RP) and tells annotators what to do in different situations. In Dickinson (2006) we note that these difficult tagging distinctions are often the same ones that are found in detecting errors arising from inconsistencies (Dickinson and Meurers, 2003).

The tagging manual gives different diagnostics to tell the confusing parts of speech apart, such as, "A word is a particle (RP) rather than a preposition (IN) ... if it can either precede or follow a noun phrase object" (Santorini, 1990, p. 10). The crucial insight is that the diagnostics used to tell, for example, RP from IN are not the same as the ones used to tell RP from RB (adverb). In other words, these RP uses have distributional differences based on which distinction is involved.

To implement this idea, the tagset is altered while training, replacing each relevant tag with a *complex ambiguity tag*, indicating that word's ambiguity class and the tag at that corpus position. It is essentially a tag-splitting method (cf. Brants, 1996; Ule, 2003), and it results in examples like (1a) becoming (1b) in the Wall Street Journal (WSJ) part of the Penn Treebank 3 (Marcus et al., 1993). This bears much similarity to certain of the "syntactically-conditioned modifications of closed classes" in MacKinlay (2005), such as dividing IN into IN-RP ambiguous words and unambiguous IN words, but instead of using only one or two manually-derived splits, we automatically derive a range of distinctions.

(1) a. ago/RB
    b. ago/<IN/RB,RB>

A key element of the work in Dickinson (2006) is in how to assign complex ambiguity tags without greatly increasing the number of parameters. In other words, what ambiguity class should be assigned for each corpus position? Two constraints are used to create ambiguity classes. First, low-frequency tags are filtered from consideration for an ambiguity class. For example, *an* is assigned the simple tag DT instead of the complex tag <COMMA/DT,DT> because the comma tag only occurs once out of 4211 times. Secondly, only the ambiguity classes for the positions flagged by an error detection phase (Dickinson and Meurers, 2003) are considered. Thus, a variation between JJ (adjective) and PRP (personal pronoun) for *ours* is not put into the model because such a variation never occurs for errors.

3 POS Tagging

We would like to adapt the automatic correction model for POS tagging since it intentionally focuses on difficult tagset distinctions and is independent of the learning algorithm. We have reason to believe that extending the correction methodology to POS tagging could result in a better tagging model. The modified version of the HMM tagger TnT (Brants, 2000) is a better fit to the original corpus (98.49% similar) than its unmodified counterpart (97.37%).

Furthermore, this model should provide better distributional statistics. Consider examples (2) and (3). We see in (2) that *away* varies between RB (adverb) and RP (particle), and in (3), *aboard* varies between RB and IN (preposition). We find that IN/RB words like *aboard* cannot take *from* as a complement in their adverbial (RB) uses (**away from*), and RB/RP words form a natural class: *apart*, *aside*, and *away*. So, at least for some classes, tag splitting—e.g., splitting RB into <IN/RB,RB> and <RB/RP,RB>—will provide better distributional data.

(2) a. the Cray-3 machine is at least another year away/RB from a ... prototype
    b. ... I will give away/RP the store

(3) a. Saturday 's crash ... jetliner that killed 132 of the 146 people aboard/RB
    b. These are used aboard/IN military helicopters

Following Toutanova et al. (2003), we use the WSJ corpus merged data, sections 00-18 for training and sections 19-21 for development. Sections 22-24 are reserved for testing, but we have thus far only experimented on the developmental data. All tagset modification is done to training data only, and tags are mapped back to Penn Treebank tags for evaluation. As a baseline, we find that the default version of the HMM tagger TnT (Brants, 2000) obtains a precision of 96.48% on the developmental data.

3.1 Applying the correction model

As a first pass, we can simply apply the error correction method to POS tagging. Namely, in assigning ambiguity classes, we first filter out tags occurring less than 0.01 of the time for a word and less than 10 times overall. And then ambiguity classes are used which are derived from words varying in context, that is, words that have different annotations with identical surrounding context (the so-called variation nuclei of Dickinson and Meurers (2003)). These ambiguity classes are used in the complex ambiguity tags in the training data, and if no ambiguity class is relevant, the word is given a simple tag, i.e., its original unaltered tag.

Training on this model and running it on the developmental data, we find a precision of 96.53%, an increase of only 0.05%. Simply porting the model to POS tagging, then, is not very effective.

3.2 Altering the model for tagging

POS tagging differs in crucial ways from error correction, requiring the algorithm to be adapted. First, training data and testing data are disjoint for tagging, whereas they are identical for error correction (i.e., the entire corpus is used for both). Secondly, POS tagging is for an entire text, whereas automatic correction focuses only on positions flagged by an error detection method. As a consequence, the error correction tagging model only uses ambiguity classes from the flagged positions. In assigning ambiguity classes for POS tagging, however, we need to emphasize large-impact generalizations.

As a first pass, we can assign ambiguity classes based on all possible tags for a word, obtained from the training data (cf. Cutting et al., 1992). With this training model, we see very little increase, obtaining only 96.54% precision. The problem is that this method results in far too many specific tags. With 887 total tags and 280 distinct ambiguity classes, we find unique classes like <JJ/JJR/RB/RBR/VB,VB> for the word *further*.

Thus, we need to limit what ambiguity classes are possible for a word, and to that end, we first filter low-frequency tags from consideration and then further restrict our attention to only those classes which have a large impact. At a given corpus position, any tag for which an ambiguity class is not applicable is then given a simple tag.

Determining ambiguity classes In order to limit the number of tags for an ambiguity class, we filter out tags occurring less than 10% of the time for a word. This filtering removes erroneous tags and low-frequency tags that are not indicative of a word. For example, instead of <CD/DT/JJ/NN/NNP/VBP,DT> for *the*—which has five erroneous tags—we have DT. As an example of a non-indicative tag, the word *all* varies between DT, PDT, and RB, but RB accounts for 3.7% of the cases (38/1017); after filtering, we obtain DT/PDT and broaden coverage of our ambiguity classes. It is not that RB is wrong; it is just that by restricting our focus to only DT and PDT, we are able to group *all* with a word like *nary* in its DT and PDT uses. To be more explicit, *all* now has three possible tags: <DT/PDT,DT>, <DT/PDT,PDT>, and RB. This tagging model uses 155 ambiguity classes and gives us 96.63% precision on the developmental data.

Filtering improves POS tagging, but we still have some very specific classes and ones which do not fall under the category of "confusing parts of speech." For instance, *Put* (which only appears 17 times) has the ambiguity class JJ/NN/VB/VBD/VBN, which no other word has, and ambiguity classes like $/NNP (for the token *C*) are not problematic variations for annotators (Santorini, 1990). Thus, after filtering tags below the 10% threshold, we then only use the the classes with the broadest impact. Concretely, we keep the ambiguity classes with more than n tokens, where n is empirically determined. So, a class like JJ/RB, which has 59 word types realized with 5054 tokens, is ranked above NN/NNP, with 378 word types and 4217 tokens. Such a token-based measure reflects decisions the tagger sees repeatedly and thus guarantees wider coverage. The best result, for n = 400 (hereafter *TnT400*), is 96.66%, with 38 ambiguity classes, as can be seen in figure 1.

A side effect of this method is that we obtain some

n	Precision	Classes
100	96.64%	56
200	96.64%	47
300	**96.66%**	**42**
400	**96.66%**	**38**
500	96.65%	34
1000	96.62%	20

Figure 1: Results on developmental data

classes that are essentially lexicalized, a technique which others have shown to be useful in tagging (e.g., Pla and Molina, 2004). For example, *that* is the only DT/IN/WDT word, with 7699 tokens, and thus tagging an item as <DT/IN/WDT,DT> is the same as tagging it <*that*,DT>. With a value of 400 for n, seven of the 38 ambiguity classes are lexicalized cases, as shown in figure 2.

Word	Class
there	DT/IN/WDT
over	IN/RP
'	''/POS
there	EX/RB
no	DT/RB
like	IN/VB
her	PRP/PRP$

Figure 2: Lexicalized classes in TnT400

Because we also filter out non-indicative tags, there is a slight difference between our "lexicalized" classes and the notion of lexicalization used by others. Consider the word *like*; in our model, it has four possible tags: <IN/VB,IN>, <IN/VB,VB>, JJ, and VBP. So, the JJ and VBP cases will get grouped with other JJ and VBP cases, whereas in a lexicalized model, they would receive the tags <*like*,JJ> and <*like*,VBP>. Our hope is to simply let non-indicative tags be handled by overall corpus information. The general advantage of our method is that we usually have information about broader classes of words for tagging. For example, when using lexicalized features in Ratnaparkhi (1996), the performance for *about* goes down, because of "tagging errors due to inconsistent data" (Ratnaparkhi, 1996, p. 138). In our model, *about* has 16 other IN/RB words from which to gather information, and we thus hope to overcome this problem.

Discussion The increase in tagging precision, from 96.48% to 96.66%, is only an increase of 0.18% and as we will see in section 4, we have reason to question even this improvement. Furthermore, it is still well below the state-of-the-art precision of 97.24% by Toutanova et al. (2003) on the same data. Why, then, should this tagging method be considered?

There are a few reasons to pursue this line of research. First, the tagset alteration can be applied to any POS tagging algorithm, and this is a step in improving pre-existing tagging methods by changing the tagset (cf. MacKinlay, 2005). Given that we kept the corpus constant, this empirically demonstrates that the choice of tagset distinctions affects the tagging performance. Even if we have certain external (linguistic) criteria that need to be maintained in the corpus, we can realign the tagset (with an unambiguous mapping to the original tagset) to better meet the internal criterion of effective tagging (cf. Elworthy, 1995; Déjean, 2000). Future work will go into applying such methodology to a variety of different kinds of taggers.

Secondly, this method can target words or ambiguities that are of interest and are of particular difficulty. After all, this framework works well when narrowing in only on inconsistent errors for error correction. Future work will have to go into investigating exactly which (kinds of) classes are useful for POS tagging and why. For example, non-local distinctions such as that between past tense verb (VBD) and past participle (VBN) may not benefit from the complex ambiguity tag methodology.

As a starting point, we can investigate the differences between the confusions for these two methods. The the top ten most common types of confusions (i.e., TnT-benchmark disagreements) for the default TnT are given in figure 3. In figure 4, the confusion matrix for TnT400 is given, along with the difference between this model and the TnT model; cases where tagging performance improved are in bold (i.e., the number of errors went down, resulting in a negative difference).

We can see from figures 3 and 4 that TnT400 has different problems and different biases than does TnT. Looking at the columns where JJ is the origi-

Original	TnT	Count
NN	JJ	436
NN	NNP	264
VBD	VBN	220
JJ	NN	204
VBN	VBD	156
IN	RB	149
VBN	JJ	140
JJ	VBN	129
RB	RP	127
NNP	NNPS	122

Figure 3: Confusions for TnT

Original	TnT400	Count	Difference
NN	**JJ**	**396**	**-40**
JJ	NN	246	42
NN	**NNP**	**218**	**-46**
VBD	**VBN**	**195**	**-25**
VBN	VBD	173	17
JJ	VBN	163	34
IN	RB	149	0
VBP	VB	127	16
NNP	NNPS	123	1
VBN	**JJ**	**111**	**-29**
...	...	...	
RB	**RP**	**54**	**-73**

Figure 4: Confusions for TnT400

nal tag, for instance, we find that TnT400 has more erroneous cases than TnT, but less cases when JJ is the tag it (mis)guessed. In other words, performance when guessing JJ improved, while tagging NN got worse. Following Toutanova and Manning (2002) and MacKinlay (2005), we need to further evaluate performance on individual classes to see where improvements are and adjust the model accordingly. In Dalrymple (to appear), adjective-noun disagreements accounted for 29.63% of ambiguities between parses, showing that narrowing in on this tag distinction, for example, will have an impact on parsing performance.

Exploring these strengths and weaknesses of the TnT400 model and why they occur could lead us to further improve the model. A next step is to train TnT using only one class at a time to see which classes are the most effective.

4 The Effect of Errors

It has been mentioned in several places (Dickinson and Meurers, 2005; Květŏn and Oliva, 2002; van Halteren, 2000; Padro and Marquez, 1998) that annotation errors are problematic for natural language processing, and so to fully evaluate the improvements of our tagger, we need to investigate the effect of errors in the data.

4.1 Evaluation data errors

Since we have kept the corpus data constant for different tagging methods, we can use our results to illustrate the effect of errors in the evaluation data. For the output of TnT on the developmental data, we sampled 100 differences between the benchmark and the tagged corpus, removed the original tags, and marked the correct tag, based only on the tagging manual (Santorini, 1990). We then performed the same procedure for TnT400.

For the TnT400-benchmark differences, we find that 20 TnT400 "errors" are due to benchmark errors and 9 are likely correct for both. With the extra 29 correct, we estimate the true accuracy to be as high as 97.63%, instead of the reported 96.66%.

The TnT-benchmark differences, on the other hand, turn up 36 benchmark errors correct for TnT and 7 cases correct for either, giving an estimated true accuracy of up to 97.99%. Thus, our modifications actually make the model worse, even though they appear to make the model better. This lends empirical evidence to the proof in Padro and Marquez (1998) that a worse tagger may have a better reported accuracy.

It might seem that we have ignored cases where the tagger and the corpus are both erroneous in the same way (i.e., they agree yet are wrong). However, the corpus error rate is the same for both tagger outputs, so the erroneous agreement rate can be factored out, and the taggers can be truly compared.[1] From this, we can conclude that TnT is a more accurate tagger than TnT400 on this data. Thus, we can see that not accounting for such errors can skew our results, and this leads us to question whether improvements in precision scores are truly tagging improvements and are not just modeling noise.

Future work will have to investigate the exact nature of the differences between TnT and the benchmark as compared to the differences between TnT400 and the benchmark. We expect there to be particular tags for which our performance has improved with TnT400.

4.2 Training data errors

After noticing the effect of errors in the evaluation data, we ran one final experiment to probe the effect of errors in the training data. Instead of (or in addition to) trying to improve the tagging technology, maybe we should try improving the quality of the data to see better performance. If we can correct the training data, we can compare TnT trained on this cleaned data with TnT trained on the original data and see what impact the cleaning of errors has.

Thus, we performed the automatic error detection and correction procedure from Dickinson (2006) on the training corpus. This uses the same procedure as described in section 3.1, except that it is run on the training data itself, and only the detected positions are kept. This makes 3533 changes to the training data and ensures a consistent training corpus.

We trained on this modified version of sections 00 through 18 and again ran the tagger (*TnTMod*) on the developmental data. We find a precision of 96.46%, similar to the original TnT precision of 96.48%.

To factor in the evaluation data errors, we again performed a random selection of 100 TnTMod-benchmark disagreements. We find that 30 positions are benchmark errors which TnTMod got right, and 3 more are correct for both. The estimated top accuracy is thus 97.70%, lower than the 97.99% of the original TnT.

It seems that our result falls in line with that of Osborne (2002), wherein NLP technologies—shallow parsers, in this case—are "surprisingly robust" to noise in the training data (p. 696). However, we need to be cautious in making this claim; Květŏn and Oliva (2002), for example, show that POS taggers are affected by noise in the training data. Additionally, we have to keep in mind that the correction model is estimated to be only 73.86% accurate in its changes (Dickinson, 2006). Thus, our results are really comparing the original training data, with its inconsistencies, against training data which is consistent and contains fewer errors. Furthermore, there is a sampling issue to consider (see below).

In light of the observation in Dalrymple (to appear) that certain tags are important for reducing parsing ambiguity, it is important to analyze which tagging variations change their distribution. Ignoring the randomly-selected data for the moment, we thus compare confusion matrices between the original TnT model and TnTMod, as given in figure 5.

Original	TnTMod	Count	Differences
NN	JJ	453	17
NN	NNP	247	-17
VBD	VBN	231	11
JJ	NN	196	-8
IN	**RB**	**194**	45
VBN	VBD	160	4
VBN	JJ	141	1
JJ	VBN	127	-2
RB	RP	126	-1
NNP	NNPS	122	0

Figure 5: Confusions for TnTMod

[1]To see this, let d_i be the number of disagreements between tagger i and the benchmark which are benchmark problems ($4634 \times 0.45 = 2085.3$ for TnT; $4399 \times 0.29 = 1275.71$ for TnT400). Further, let c_i be the number of positions which are reported to be correct (127,134 for TnT; 127,369 for TnT400). If e is the number of errors in the data, then all the errors not in d_i (or $e - d_i$) must be in c_i. The true number of correct positions is thus: $c_i - (e - d_i) + d_i = c_i - e + 2d_i$. Since e is the same for both, we can simply compare $c_i + 2d_i$, where d_i is estimated from our sample. The value of $c_i + 2d_i$ for TnT is 131,305; for TnT400, it is 129,920.

Although most tagging differences are similar, we see that there is a definite difference between TnT and TnTMod when it comes to IN/RB variations. Specifically, TnTMod has 45 more RB disagreements with the benchmark IN than TnT does. We do not know if this is actually an improvement in tagging or not—as we mention in Dickinson (2006), words like *about* are often incorrectly tagged IN instead of RB in the benchmark corpus. What we can say at this point is that the quality and consistency of the training data has a big impact on the accuracy of IN and RB tags. If these tags are important to a task, then the quality of data is also important.

Unfortunately, our sample of 100 TnTMod-benchmark disagreements contains no instances of IN words tagged by TnTMod as RB, so we cannot draw any conclusions about the benchmark errors in this particular distinction. This points to a problem with randomly sampling 100 positions from any of the tagger-benchmark disagreements; different tags have different accuracies, and we might randomly draw the wrong ones. For example, there are 30 cases in the TnTMod sample which originally had one of the canonical verb tags (VB, VBP, VBD, VBN),[2] and there are only 18 in the TnT sample.

Examining the change in accuracy of each tag, or tag ambiguity, is important, but, following Ratnaparkhi (1996) and others, it is also important to analyze the distribution of individual words. Namely, in the future, we will need to compare individual words (e.g., *about*, *that*) to see if they have a substantial increase or decrease in accuracy.

Additional work in the future includes training the TnT400 model on the cleaned data, to see whether we were modeling noise in the training data before and thus whether consistent training data results in better complex ambiguity tags. Qualitatively analyzing this output will show us the properties of the tagset which are more or less difficult and which can be overcome by an altered tagset and which by cleaned data.

5 Summary and Outlook

We have explored a new method for POS tagging, which alters the tagset by adding complex ambiguity tags reflecting each word's potential ambiguities. We found a slight improvement for this, but later discovered that this method was less robust to errors than the original tagger. With this result, we demonstrated that one must account for errors in the evaluation data when comparing taggers, and attention needs to be given to which individual tags are improved. We also showed that errors in the training data affect the accuracy of particular tags. We make two conclusions from this work. 1) We need to focus on gains for individual tags in POS tagging, and why they improve. A method able to pinpoint difficult tags is thus desirable. 2) The effect of benchmark errors needs continued analysis, in order to gauge true tagging improvement.

There are several directions in which to take this work. The tagging methodology worked for error correction because it focused on particularly problematic distinctions. Likewise, we can focus on problematic distinctions in the future by running the tagger on the training data and deriving classes which are the most confusing for the tagger.

Another possible approach is to train k different models, each one focusing on only one particular distinction, and then merging the results. This should provide better contextual data—e.g., if we are deciding between IN and RB, we want to know that the previous word is a noun (NN), not that it is <JJ/NN,NN>. In the process, this will tell us which distinctions are the most and least useful for tagging.

Once we are able to improve the model further, then of course, we still have to run it on the testing data. We would also like to run this on other taggers, given that the methodology is independent of the underlying tagging algorithm. Additionally, to gauge whether the method is generally applicable, other corpora and tagsets need to be used.

Acknowledgments I would like to thank two anonymous reviewers for their helpful comments. This material is based upon work supported by the National Science Foundation under Grant No. IIS-0623837.

References

Brants, Thorsten (1996). Estimating Markov Model Structures. In *Proceedings ICSLP-96*. Philadelphia, PA, pp. 893–896.

[2]VBG is most often confused with JJ, so we do not include it as a canonical verb tag.

Brants, Thorsten (2000). TnT – A Statistical Part-of-Speech Tagger. In *Proceedings of ANLP 2000*. Seattle, WA, pp. 224–231.

Church, Kenneth W. (1992). Current practice in part of speech tagging and suggestions for the future. In Simmons (ed.), *Sbornik praci: In Honour of Henry Kučera*, Michigan Slavic Studies, pp. 13–48.

Cutting, Doug, Julian Kupiec, Jan Pedersen and Penelope Sibun (1992). A Practical part-of-speech tagger. In *Proceedings of ANLP-92*. Trento, Italy, pp. 133–140.

Dalrymple, Mary (to appear). How much can part of speech tagging help parsing? *Natural Language Engineering* .

Déjean, Hervé (2000). How to Evaluate and Compare Tagsets? A Proposal. In *Proceedings of LREC-00*. Athens.

Dickinson, Markus (2006). From Detecting Errors to Automatically Correcting Them. In *Proceedings of EACL-06*. Trento, Italy, pp. 265–272.

Dickinson, Markus and W. Detmar Meurers (2003). Detecting Errors in Part-of-Speech Annotation. In *Proceedings of EACL-03*. Budapest, Hungary, pp. 107–114.

Dickinson, Markus and W. Detmar Meurers (2005). Prune Diseased Branches to Get Healthy Trees! How to Find Erroneous Local Trees in a Treebank and Why It Matters. In *Proceedings of TLT 2005*. Barcelona, Spain.

Elworthy, David (1995). Tagset Design and Inflected Languages. In *Proceedings of the ACL-SIGDAT Workshop*. Dublin.

Květŏn, Pavel and Karel Oliva (2002). Achieving an Almost Correct PoS-Tagged Corpus. In *Text, Speech and Dialogue (TSD)*. Heidelberg: Springer, no. 2448 in Lecture Notes in Artificial Intelligence (LNAI), pp. 19–26.

MacKinlay, Andrew (2005). The Effects of Part-of-Speech Tagsets on Tagger Performance.

Marcus, M., Beatrice Santorini and M. A. Marcinkiewicz (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics* 19(2), 313–330.

Marquez, Lluis, Lluis Padro and Horacio Rodriguez (2000). A Machine Learning Approach to POS Tagging. *Machine Learning* 39(1), 59–91.

Osborne, Miles (2002). Shallow Parsing using Noisy and Non-Stationary Training Material. In *JMLR Special Issue on Machine Learning Approaches to Shallow Parsing*, vol. 2, pp. 695–719.

Padro, Lluis and Lluis Marquez (1998). On the Evaluation and Comparison of Taggers: the Effect of Noise in Testing Corpora. In *Proceedings of COLING/ACL-98*. pp. 997–1002.

Pla, Ferran and Antonio Molina (2004). Improving part-of-speech tagging using lexicalized HMMs. *Natural Language Engineering* 10(2), 167–189.

Ratnaparkhi, Adwait (1996). A maximum entropy model part-of-speech tagger. In *Proceedings of EMNLP-96*. Philadelphia, PA, pp. 133–141.

Santorini, Beatrice (1990). *Part-Of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision, 2nd printing)*. Tech. Rep. MS-CIS-90-47, The University of Pennsylvania, Philadelphia, PA.

Toutanova, Kristina, Dan Klein, Christopher D. Manning and Yoram Singer (2003). Feature-Rich Part-of-Speech Tagging Using a Cyclic Dependency Network. In *Proceedings of HLT-NAACL 2003*. pp. 252–259.

Toutanova, Kristina and Christopher D. Manning (2002). Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. In *Proceedings of EMNLP/VLC-2000*. Hong Kong.

Ule, Tylman (2003). Directed Treebank Refinement for PCFG Parsing. In *Proceedings of TLT 2003*. Växjö, Sweden, pp. 177–188.

van Halteren, Hans (2000). The Detection of Inconsistency in Manually Tagged Text. In Anne Abeillé, Thosten Brants and Hans Uszkoreit (eds.), *Proceedings of LINC-00*. Luxembourg.

Wynne, Martin (1996). *A Post-Editor's Guide to CLAWS7 Tagging*. UCREL, Lancaster University, Lancaster.