A SIMPLE RULE-BASED PART OF SPEECH TAGGER

Eric Brill *
Department of Computer Science
University of Pennsylvania
Philadelphia, Pennsylvania 19104
brill@unagi.cis.upenn.edu

ABSTRACT
Automatic part of speech tagging is an area of natural language processing where statistical techniques have been more successful than rule-based methods. In this paper, we present a simple rule-based part of speech tagger which automatically acquires its rules and tags with accuracy comparable to stochastic taggers. The rule-based tagger has many advantages over these taggers, including: a vast reduction in stored information required, the perspicuity of a small set of meaningful rules, ease of finding and implementing improvements to the tagger, and better portability from one tag set, corpus genre or language to another. Perhaps the biggest contribution of this work is in demonstrating that the stochastic method is not the only viable method for part of speech tagging. The fact that a simple rule-based tagger that automatically learns its rules can perform so well should offer encouragement for researchers to further explore rule-based tagging, searching for a better and more expressive set of rule templates and other variations on the simple but effective theme described below.

1. INTRODUCTION
There has been a dramatic increase in the application of probabilistic models to natural language processing over the last few years. The appeal of stochastic techniques over traditional rule-based techniques comes from the ease with which the necessary statistics can be automatically acquired and the fact that very little handcrafted knowledge need be built into the system. In contrast, the rules in rule-based systems are usually difficult to construct and are typically not very robust.

One area in which the statistical approach has done particularly well is automatic part of speech tagging, assigning each word in an input sentence its proper part of speech [1, 2, 3, 4, 6, 9, 11, 12]. Stochastic taggers have

* A version of this paper appears in Proceedings of the Third Conference on Applied Computational Linguistics (ACL), Trento, Italy, 1992. Used by permission of the Association for Computational Linguistics; copies of the publication from which this material is derived can be obtained from Dr. Donald E. Walker (ACL), Bellcore, MRE 2A379, 445 South Street, Box 1910, Morristown, NJ 07960-1910, USA. The author would like to thank Mitch Marcus and Rich Pito for valuable input. This work was supported by DARPA and AFOSR jointly under grant No. AFOSR-90-0066, and by ARO grant No. DAAL 03-89-C0031 PRI.

obtained a high degree of accuracy without performing any syntactic analysis on the input. These stochastic part of speech taggers make use of a Markov model which captures lexical and contextual information. The parameters of the model can be estimated from tagged [1, 3, 4, 6, 12] or untagged [2, 9, 11] text. Once the parameters of the model are estimated, a sentence can then be automatically tagged by assigning it the tag sequence which is assigned the highest probability by the model. Performance is often enhanced with the aid of various higher level pre- and postprocessing procedures or by manually tuning the model.

A number of rule-based taggers have been built [10, 7, 8]. [10] and [7] both have error rates substantially higher than state of the art stochastic taggers. [8] disambiguates words within a deterministic parser. We wanted to determine whether a simple rule-based tagger without any knowledge of syntax can perform as well as a stochastic tagger, or if part of speech tagging really is a domain to which stochastic techniques are better suited.

In this paper we describe a rule-based tagger which performs as well as taggers based upon probabilistic models. The rule-based tagger overcomes the limitations common in rule-based approaches to language processing: it is robust, and the rules are automatically acquired. In addition, the tagger has many advantages over stochastic taggers, including: a vast reduction in stored information required, the perspicuity of a small set of meaningful rules as opposed to the large tables of statistics needed for stochastic taggers, ease of finding and implementing improvements to the tagger, and better portability from one tag set or corpus genre to another.

2. THE TAGGER
The tagger works by automatically recognizing and remedying its weaknesses, thereby incrementally improving its performance. The tagger initially tags by assigning each word its most likely tag, estimated by examining a large tagged corpus, without regard to context. In both sentences below, run would be tagged as a verb:

112
**Report Documentation Page**

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

| 1. REPORT DATE | 2. REPORT TYPE | 3. DATES COVERED |
|----------------|----------------|-----------------|
| 1992           |                | 00-00-1992 to 00-00-1992 |

| 4. TITLE AND SUBTITLE |
|-----------------------|
| A Simple Rule-Based Part of Speech Tagger |

| 5a. CONTRACT NUMBER |
|---------------------|
|                    |

| 5b. GRANT NUMBER |
|------------------|
|                  |

| 5c. PROGRAM ELEMENT NUMBER |
|-----------------------------|
|                             |

| 5d. PROJECT NUMBER |
|--------------------|
|                    |

| 5e. TASK NUMBER |
|-----------------|
|                 |

| 5f. WORK UNIT NUMBER |
|----------------------|
|                      |

| 6. AUTHOR(S) |
|              |

| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) |
|-----------------------------------------------------|
| Department of Computer and Information Sciences, University of Pennsylvania, Philadelphia, PA, 19104 |

| 8. PERFORMING ORGANIZATION REPORT NUMBER |
|------------------------------------------|
|                                         |

| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) |
|--------------------------------------------------------|

| 10. SPONSOR/MONITOR’S ACRONYM(S) |
|----------------------------------|
|                                  |

| 11. SPONSOR/MONITOR’S REPORT NUMBER(S) |
|----------------------------------------|
|                                       |

| 12. DISTRIBUTION/AVAILABILITY STATEMENT |
|-----------------------------------------|
| Approved for public release; distribution unlimited |

| 13. SUPPLEMENTARY NOTES |
|-------------------------|
|                         |

| 14. ABSTRACT |
|--------------|
|              |

| 15. SUBJECT TERMS |
|-------------------|
|                   |

| 16. SECURITY CLASSIFICATION OF: |
|---------------------------------|
| a. REPORT unclassified          |
| b. ABSTRACT unclassified        |
| c. THIS PAGE unclassified        |

| 17. LIMITATION OF ABSTRACT |
|-----------------------------|
|                            |

| 18. NUMBER OF PAGES |
|---------------------|
| 5                   |

| 19a. NAME OF RESPONSIBLE PERSON |
|---------------------------------|

Standard Form 298 (Rev. 8-98)  
Prescribed by ANSI Std Z39-18
The run lasted thirty minutes. We run three miles every day.

The initial tagger has two procedures built in to improve performance; both make use of no contextual information. One procedure is provided with information that words that were not in the training corpus and are capitalized tend to be proper nouns, and attempts to fix tagging mistakes accordingly. This information could be acquired automatically (see below), but is prespecified in the current implementation. In addition, there is a procedure which attempts to tag words not seen in the training corpus by assigning such words the tag most common for words ending in the same three letters. For example, *blahblahous* would be tagged as an adjective, because this is the most common tag for words ending in *ous*. This information is derived automatically from the training corpus.

This very simple algorithm has an error rate of about 7.9% when trained on 90% of the tagged Brown Corpus\(^1\), and tested on a separate 5% of the corpus.\(^2\) Training consists of compiling a list of the most common tag for each word in the training corpus.

The tagger then acquires patches to improve its performance. Patch templates are of the form:

- If a word is tagged \(a\) and it is in context \(C\), then change that tag to \(b\), or
- If a word is tagged \(a\) and it has lexical property \(P\), then change that tag to \(b\), or
- If a word is tagged \(a\) and a word in region \(R\) has lexical property \(P\), then change that tag to \(b\).

The initial tagger was trained on 90% of the corpus (the training corpus). 5% was held back to be used for the patch acquisition procedure (the patch corpus) and 5% for testing. Once the initial tagger is trained, it is used to tag the patch corpus. A list of tagging errors is compiled by comparing the output of the tagger to the correct tagging of the patch corpus. This list consists of triples \(<tag_a, tag_b, number>\), indicating the number of times the tagger mistagged a word with \(tag_a\) when it should have been tagged with \(tag_b\) in the patch corpus. Next, for each error triple, it is determined which instantiation of a template from the prespecified set of patch templates results in the greatest error reduction. Currently, the patch templates are:

Change tag \(a\) to tag \(b\) when:

1. The preceding (following) word is tagged \(z\).
2. The word two before (after) is tagged \(z\).
3. One of the two preceding (following) words is tagged \(z\).
4. One of the three preceding (following) words is tagged \(z\).
5. The preceding word is tagged \(z\) and the following word is tagged \(w\).
6. The preceding (following) word is tagged \(z\) and the word two before (after) is tagged \(w\).
7. The current word is (is not) capitalized.
8. The previous word is (is not) capitalized.

For each error triple \(<tag_a, tag_b, number>\) and patch, we compute the reduction in error which results from applying the patch to remedy the mistagging of a word as \(tag_a\) when it should have been tagged \(tag_b\). We then compute the number of new errors caused by applying the patch; that is, the number of times the patch results in a word being tagged as \(tag_b\) when it should be tagged \(tag_a\). The net improvement is calculated by subtracting the latter value from the former.

For example, when the initial tagger tags the patch corpus, it mistags 159 words as verbs when they should be nouns. If the patch *change the tag from verb to noun if one of the two preceding words is tagged as a determiner* is applied, it corrects 98 of the 159 errors. However, it results in an additional 18 errors from changing tags which really should have been *verb to noun*. This patch results in a net decrease of 80 errors on the patch corpus.

The patch which results in the greatest improvement to the patch corpus is added to the list of patches. The patch is then applied in order to improve the tagging of the patch corpus, and the patch acquisition procedure continues.
The first ten patches found by the system are listed below.

(1) TO IN NEXT-TAG AT
(2) VBN VBD PREV-WORD-IS-CAP YES
(3) VBD VBN PREV-1-OR-2-OR-3-TAG HVD
(4) VB NN PREV-1-OR-2-TAG AT
(5) NN VB PREV-TAG TO
(6) TO IN NEXT-WORD-IS-CAP YES
(7) NN VB PREV-TAG MD
(8) PPS PPO NEXT-TAG .
(9) VBN VBD PREV-TAG PPS
(10) NP NN CURRENT-WORD-IS-CAP NO

The first patch states that if a word is tagged TO and the following word is tagged AT, then switch the tag from TO to IN. This is because a noun phrase is much more likely to immediately follow a preposition than to immediately follow infinitive TO. The second patch states that a tag should be switched from VBN to VBD if the preceding word is capitalized. This patch arises from two facts: the past verb tag is more likely than the past participle verb tag after a proper noun, and is also the more likely tag for the second word of the sentence. The third patch states that VBD should be changed to VBN if any of the preceding three words are tagged HVD.

Once the list of patches has been acquired, new text can be tagged as follows. First, tag the text using the basic lexical tagger. Next, apply each patch in turn to the corpus to decrease the error rate. A patch which changes the tagging of a word from a to b only applies if the word has been tagged b somewhere in the training corpus.

Note that one need not be too careful when constructing the list of patch templates. Adding a bad template to the list will not worsen performance. If a template is bad, then no rules which are instantiations of that template will appear in the final list of patches learned by the tagger. This makes it easy to experiment with extensions to the tagger.

3 AT = article, HVD = had, IN = preposition, MD = modal, NN = sing. noun, NP = proper noun, PPS = 3rd sing. nom. pronoun, PPO = obj. personal pronoun, TO = infinitive to, VB = verb, VBN = past part. verb, VBD = past verb.

4 Both the first word of a sentence and proper nouns are capitalized.

3. RESULTS
The tagger was tested on 5% of the Brown Corpus including sections from every genre. First, the test corpus was tagged by the simple lexical tagger. Next, each of the patches was in turn applied to the corpus. Below is a graph showing the improvement in accuracy from applying patches. It is significant that with only 71 patches, an error rate of 5.1% was obtained. Of the 71 patches, 66 resulted in a reduction in the number of errors in the test corpus, 3 resulted in no net change, and 2 resulted in a higher number of errors. Almost all patches which were effective on the training corpus were also effective on the test corpus.

Unfortunately, it is difficult to compare our results with other published results. In [12], an error rate of 3-4% on one domain, Wall Street Journal articles and 5.6% on another domain, texts on terrorism in Latin American countries, is quoted. However, both the domains and the tag set are different from what we use. [1] reports an accuracy of “95-99% correct, depending on the definition of correct”. We implemented a version of the

5 We ran the experiment three times. Each time we divided the corpus into training, patch and test sets in a different way. All three runs gave an error rate of 5%.
algorithm described in [1] which did not make use of a
dictionary to extend its lexical knowledge. When trained
and tested on the same samples used in our experiment,
we found the error rate to be about 4.5%. [3] quotes
a 4% error rate when testing and training on the same
text. [6] reports an accuracy of 96-97%. Their proba-
bilistic tagger has been augmented with a handcrafted
procedure to pretag problematic "idioms". This proce-
dure, which requires that a list of idioms be laboriously
created by hand, contributes 3% toward the accuracy of
their tagger, according to [3]. The idiom list would have
to be rewritten if one wished to use this tagger for a
different tag set or a different corpus. It is interesting
to note that the information contained in the idiom list
can be automatically acquired by the rule-based tagger.
For example, their tagger had difficulty tagging as old as
An explicit rule was written to pretag as old as with
the proper tags. According to the tagging scheme of the
Brown Corpus, the first as should be tagged as a quali-
fier, and the second as a subordinating conjunction. In
the rule-based tagger, the most common tag for as is
subordinating conjunction. So initially, the second as is
tagged correctly and the first as is tagged incorrectly.
To remedy this, the system acquires the patch: if the cur-
rent word is tagged as a subordinating conjunction, and
so is the word two positions ahead, then change the tag of
the current word to qualifier. The rule-based tagger has
automatically learned how to properly tag this "idiom."

Regardless of the precise rankings of the various taggers,
we have demonstrated that a simple rule-based tagger
with very few rules performs on par with stochastic tag-
gers. It should be mentioned that our results were ob-
tained without the use of a dictionary. Incorporating a
large dictionary into the system would improve per-
formance in two ways. First, it would increase the accuracy
in tagging words not seen in the training corpus, since
part of speech information for some words not appearing
in the training corpus can be obtained from the dic-
tionary. Second, it would increase the error reduction
resulting from applying patches. When a patch indicates
that a word should be tagged with tagb instead of taga,
the tag is only switched if the word was tagged with taga
somewhere in the training corpus. Using a dictionary
would provide more accurate knowledge about the set
of permissible part of speech tags for a particular word.
We plan to incorporate a dictionary into the tagger in
the future.

As an estimate of the improvement possible by using
a dictionary, we ran two experiments where all words
were known by the system. First, the Brown Corpus
was divided into a training corpus of about one million
words, a patch corpus of about 65,000 words and a test
corpus of about 65,000 words. Patches were acquired
as described above. When tested on the test corpus,
with lexical information derived solely from the training
corpus, the error rate was 5%. Next, the same patches
were used, but lexical information was gathered from
the entire Brown Corpus. This reduced the error rate to
4.1%. Finally, the same experiment was run with lexical
information gathered solely from the test corpus. This
resulted in a 3.5% error rate. Note that the patches used
in the two experiments with no unknown words were
not the optimal patches for these tests, since they were
derived from a corpus that contained unknown words.

4. CONCLUSIONS

We have presented a simple rule-based part of speech
tagger which performs as well as existing stochastic tag-
gers, but has significant advantages over these taggers.

The tagger is extremely portable. Many of the higher
level procedures used to improve the performance of
stochastic taggers would not readily transfer over to a
different tag set or genre, and certainly would not trans-
fer over to a different language. Everything except for
the proper noun discovery procedure is automatically ac-
quired by the rule-based tagger, making it much more
portable than a stochastic tagger. If the tagger were
trained on a different corpus, a different set of patches
suitable for that corpus would be found automatically.

Large tables of statistics are not needed for the rule-
based tagger. In a stochastic tagger, tens of thousands
of lines of statistical information are needed to capture
contextual information. This information is usually a ta-
ble of trigram statistics, indicating for all tags taga, tagb,
and tagc the probability that tagc follows taga and tagb.
In the rule-based tagger, contextual information is cap-
tured in fewer than eighty rules. This makes for a much
more perspicuous tagger, aiding in better understanding
and simplifying further development of the tagger. Con-
textual information is expressed in a much more compact
and understandable form. As can be seen from compar-
ing error rates, this compact representation of contextual
information is just as effective as the information hidden
in the large tables of contextual probabilities.

Perhaps the biggest contribution of this work is in
demonstrating that the stochastic method is not the only
viable approach for part of speech tagging. The fact that
the simple rule-based tagger can perform so well should
offer encouragement for researchers to further explore
rule-based tagging, searching for a better and more ex-

\footnote{This was one of the 71 patches acquired by the rule-based
tagger.}

\footnote{And even this could be learned by the tagger.}
pressive set of patch templates and other variations on
this simple but effective theme.

References
1. Church, K. A Stochastic Parts Program and Noun
Phrase Parser for Unrestricted Text. In Proceedings of
the Second Conference on Applied Natural Language
Processing, ACL, 136-143, 1988.
2. Cutting, D., Kupiec, J., Pederson, J. and Sibun, P. A
Practical Part-of-Speech Tagger. In Proceedings of the
Third Conference on Applied Natural Language Process-
ing, ACL, 1992.
3. DeRose, S.J. Grammatical Category Disambiguation by
Statistical Optimization. Computational Linguistics 14:
31-39, 1988.
4. Deroualt, A. and Merialdo, B. Natural language mod-
eling for phoneme-to-text transcription. IEEE Transac-
tions on Pattern Analysis and Machine Intelligence, Vol.
PAMI-8, No. 6, 742-749, 1986.
5. Francis, W. Nelson and Kučera, Henry, Frequency anal-
ysis of English usage. Lexicon and grammar. Houghton
Mifflin, Boston, 1982.
6. Garside, R., Leech, G. & Sampson, G. The Computa-
tional Analysis of English: A Corpus-Based Approach.
Longman: London, 1987.
7. Green, B. and Rubin, G. Automated Grammatical Tag-
ing of English. Department of Linguistics, Brown Uni-
versity, 1971.
8. Hindle, D. Acquiring disambiguation rules from text.
Proceedings of the 27th Annual Meeting of the Associa-
tion for Computational Linguistics, 1989.
9. Jelinek, F. Markov source modeling of text generation.
In J. K. Skwirzinski, ed., Impact of Processing Tech-
niques on Communication, Dordrecht, 1985.
10. Klein, S. and Simmons, R.F. A Computational Ap-
proach to Grammatical Coding of English Words. JACM
10: 334-47. 1963.
11. Kupiec, J. Augmenting a hidden Markov model for
phrase-dependent word tagging. In Proceedings of the
DARPA Speech and Natural Language Workshop, Morgan
Kaufmann, 1989.
12. Meteer, M., Schwartz, R., and Weischedel, R. Empirical
Studies in Part of Speech Labelling, Proceedings of the
DARPA Speech and Natural Language Workshop, Mor-
gan Kaufmann, 1991.