

# CIS 530 Fall 2009 HW 3

Instructor: Mitch Marcus

TA: Constantine Lignos

Revised: November 10th, 2009

Due: November 12th, 2009 at 4PM

## Overview of this assignment

In statistical natural language processing, one can use learning algorithms to achieve reasonably good results in most applications. But in order to improve performance, one should take a look at what errors the algorithm is making. Error analysis serves as a starting point to deciding what to do next, whether that means using more features, modifying algorithm parameters, or even switching to a different learning algorithm because one has decided that the current model is not adequate. This assignment is in exercise in digging into data and getting a feel for how a technique is performing.

## Submitting your work

### Non-code Organization

Unlike previous assignments, **do not submit any code for this assignment**. Submit your answers to all questions in a single PDF file.

You may create your solution in any program you choose (Word, OpenOffice, L<sup>A</sup>T<sub>E</sub>X, etc.) but you must export a PDF from that program to turn in. Do not turn in a .doc, .docx, .txt, .tex, or .dvi file, or any file format other than .pdf. If you do, you will be penalized 5 points and may be required to convert your file.

Please submit all your answers for the assignment in a single file called “hw3-yourpennkey.pdf” where “yourpennkey” is your PennKey.

Once again, do not submit any code for this assignment, and do not submit any file format other than PDF.

### Submitting files

To electronically submit homework, if you're not already working on Eniac you need to place the file containing your solution on your SEAS account storage. One way to do this is using an SFTP client such as FileZilla.

Then connect via ssh to seas.upenn.edu and use the turnin command to submit your file for grading:

```
% turnin -c cis530 -p hw3 <filename>
```

This should print out a confirmation message. You can run turnin multiple times before the deadline, but each time you run turnin, it overwrites your previous submission for that assignment. You can check that the homework submitted successfully:

```
% turnin -c cis530 -v
```

This will show a list of the file(s) you have submitted.

## Introduction

In this assignment, you will perform an error analysis on three versions of Brill’s transformation-based part-of-speech tagger on text from both the Brown and the Wall Street Journal (WSJ) corpora. Both corpora are available in annotated format in the Penn Treebank. Some programming will be necessary, but the main part of the assignment will be your analysis, and you should accordingly concentrate on the written part of the assignment, not on the coding.

Training a tagger on almost the complete Penn Treebank requires lots of time or memory. For this reason, the training and applying has been done for you beforehand. Should you ever want to run the tagger yourself (e.g. for a term project), please feel free to contact us. In case you’re curious, the software package we used, `fntbl1`, is an extremely fast reimplementation of Brill’s original tagger.

## Motivating the problem

The Penn Treebank contains 50,000 sentences (1,000,000 words) of WSJ text from 1989. Its current version also contains about the same amount of words from the Brown corpus. The Brown corpus is a balanced corpus—it contains texts from a wide variety of written sources. The WSJ corpus, not surprisingly, consists exclusively of financial news items. You might expect that the Brown corpus would be a better choice for training taggers and parsers. But the first and second versions of the Penn Treebank only contained the WSJ corpus, and for better or for worse it has established itself as the standard training and test data in the field. But is it really beneficial to train and test on the “same” data? Even though the actual training and test data have of course always been kept disjoint, they are still taken from the same genre of text. We might observe results that are skewed upwards and fool ourselves into thinking that our tools perform better than they would ever do in “real life” applications. Plus, training on homogeneous data that’s taken from only one genre might make us find fake generalizations that are not true of the English language in general. In other words, we might overfit.

So what happens to performance when we test on a different corpus (Brown) as opposed to another section of the same corpus (WSJ) that we’ve trained on? And what happens when we make our training data more heterogeneous by switching to the Brown corpus, or training on parts of both corpora? In this assignment, you will answer these questions for one specific case: POS tagging. You will be asked to compare the performance of various instances of the Brill tagger. You will evaluate both the overall accuracy of the taggers and their performance on the level of individual tags. Error analysis is a tedious task, but the only way to figure out what’s wrong with your system is to sit down and look at the data it produces.

## Description of the files

Download and unpack the file [http://www.cis.upenn.edu/~cis530/hw\\_2009/hw3data.zip](http://www.cis.upenn.edu/~cis530/hw_2009/hw3data.zip). It contains six files, corresponding to the application of three versions of the Brill tagger to two different texts each. The three versions of the tagger are:

1. **brillwsj**- Brill’s TBL Tagger, trained on the Wall Street Journal
2. **brillbrown**- Brill’s TBL Tagger, trained on the Brown corpus
3. **brillboth**- Brill’s TBL Tagger, trained on both Brown and WSJ

The names are given to provide you with an easy way of referring to each tagger in the subsequent problems. The files are named according to the following convention: `taggername.testfile.txt`. For example, `brillbrown.wsj.txt` is the result of training Brill’s tagger on the Brown corpus and applying it to the WSJ. For the WSJ corpus, the following tradition has evolved in NLP: sections 2-21 are used for training, section 22 or 24 is held out for development, and section 23 is used as the final test data. Sections 0 and 1 are perceived to be less reliable than the others, and thus are generally not used. The above taggers are trained on sections 2 to 21 and applied to section 23. For the Brown corpus, there is no convention. So each section

of the corpus was split in approximately two halves to create two balanced subcorpora, one for training and one for test.

In each file you downloaded, each line contains a token followed by the tag assigned to it by the tagger, followed by the correct (gold standard) tag. Sentences are followed by a blank line. For example, the line `bidding VBG NN` indicates that the word `bidding` has been incorrectly tagged as a present participle (VBG), but is really a common noun (NN) (or at least that's what the human annotators considered it). The Penn Treebank tagset is included at the end of this assignment.

## 1 Evaluate the taggers (20 points)

In Python or in your favorite language, write a program that evaluates the performance of each tagger on each corpus. Your program should calculate and print the accuracy in percent for the given tagger output, compared to the given gold standard. The accuracy is defined as the number of correctly tagged tokens divided by the total number of tokens in the corpus. Remember that some lines in the files are blank and do not contain any tokens. You do not need to turn in the source code of your program, however, you must write it on your own.

You'll report accuracies for two conditions:

1. All tokens in the corpus
2. All non-punctuation tokens in the corpus- To compute this, calculate the accuracy excluding all tokens tagged as punctuation in the gold standard. These tokens are tagged with tags 37-48 as listed at the end of this document.

In some cases, the gold standard or the tagger assigns two tags to a token, such as `hunting NN|VBG`. Ignore these tokens in computing accuracy for both conditions. That is, do not count them toward the total number of tokens in the corpus or the number of correctly tagged tokens.

To report the accuracy of each of the taggers on each corpus under each condition, arrange your results in a 3-by-2 table, with one row for each test corpus and one column for each trained version of the tagger, like so:

	WSJ	Brown	Both
WSJ	x%	y%	z%
Brown	a%	b%	c%

You'll produce a table like this for each condition. Write one paragraph answering the following questions: Can you make any generalizations from these accuracy rates? Does it seem more important for the Brill tagger to have more training data or a better match between test and training data?

## 2 Error analysis

### 2.1 Data preparation (10 points)

In this exercise, you are asked for each tagger/corpus combination to examine the errors the tagger makes. Write a program that takes a file containing the tagger output and gold standard (as in the previous problem) and computes a confusion matrix. A confusion matrix is a table that records for each combination of two non-identical tags A and B the percentage of times that a token was mistakenly tagged as A when its gold standard tag was in fact B. Again, you may ignore any token tagged with multiple tags in the gold standard.

You do not need to hand in your code, nor the confusion matrices. Instead, for the tagger trained on both the Brown and WSJ corpus, hand in a list or table showing the top ten entries of its confusion matrix with

their respective percentages for each testing corpus. Thus, hand in results for the files `brillboth_wsj.txt`, and `brillboth_brown.txt`.

For example, let's pretend there are only three tags, NN, JJ, and VBN, in the WSJ. We see 100 tokens whose gold standard tags are NN. Of those 100 tokens, 90 are tagged NN, 7 are tagged JJ, and 3 are tagged VBN. We also see 100 tokens whose gold standard tags are JJ, and of those 70 are tagged JJ, 29 are tagged VBN, and 1 is tagged NN. Your summarization of the confusion matrix would be as follows:

```
Trained on: Both
Tested on: WSJ
Tagger output-Gold standard, Percent Occurrences
VBN-JJ 29%
JJ-NN 7%
VBN-NN 3%
NN-JJ 1%
```

Of course, these are not the actual entries you will get for this file. Remember not to include correct tagging (i.e. entries of the form A-A) in your results summary.

## 2.2 Analysis (30 points)

Discuss your findings in 1-2 pages (and no more!), including any space needed for compact examples. We are aware that a linguistic background is not a prerequisite for the class and that students have a varying level of experience with English, and we will take this into consideration when we review your responses. We want to see that you dug into the data and found some kind of patterns and have a theory, even if it's a weak or incorrect one, to explain what's going on.

Make sure to answer at least these questions, but if you find interesting patterns not addressed here you may focus on exploring them as well and answer these only briefly. You need not address all six accuracy numbers for both conditions; feel free to pick the conditions and training/test pairings that seem most interesting to you. Unless you find it particularly interesting, you need not discuss the condition that includes punctuation tags in the accuracy level at all.

- What are the most frequent errors overall? What do their POS tags stand for? Give examples of sentences or phrases (i.e. sentence parts) that contain these errors and discuss why an automatic tagger might have trouble with them<sup>1</sup>. If there is significant variation across training sets and/or test corpora, and if you have ideas about them, you might also want to discuss for some errors why they might be more severe in some cases and less in others. Are some errors the tagger makes more likely to also be made by a human than others? You may find the Treebank POS annotation guide to be helpful: [http://repository.upenn.edu/cgi/viewcontent.cgi?article=1603&context=cis\\_reports](http://repository.upenn.edu/cgi/viewcontent.cgi?article=1603&context=cis_reports)
- How could tagger performance be increased? Would it help to write additional transformational rules and add them to the rule list generated by training the Brill Tagger? If so, give an example of such a rule. If not, explain why not. Review the slides on Brill Tagging if you're not sure. There's no clear right/wrong answer here, but you should use what you know about the Brill Tagger to create a hypothesis and give some facts that could support it.

## 3 Debriefing

Include the answers to these questions at the end of your solution.

1. Approximately how many hours did you spend on this assignment?

---

<sup>1</sup>Hint: Morphology, which in English is generally expressed as the endings of words, is definitely something you should pay attention to.

2. Would you rate it as easy, moderate, or difficult?
3. How deeply do you feel you understand the material it covers (0%-100%)?
4. Any other comments?

These questions are intended to help us calibrate the homework assignments. Your answers will not affect your grade, and unless you request it we will not contact you about them.

## Revision History

11/4/2009	Initial version
11/10/2009	Added clarification that the tagger can also output two tags and a link to the annotation guidelines

## 4 Appendix: The Penn Treebank Tagset

1. CC Coordinating conjunction
2. CD Cardinal number
3. DT Determiner
4. EX Existential there
5. FW Foreign word
6. IN Preposition or subordinating conjunction
7. JJ Adjective
8. JJR Adjective, comparative
9. JJS Adjective, superlative
10. LS List item marker
11. MD Modal
12. NN Noun, singular or mass
13. NNS Noun, plural
14. NNP Proper noun, singular
15. NNPS Proper noun, plural
16. PDT Predeterminer
17. POS Possessive ending
18. PRP Personal pronoun
19. PRP\$ Possessive pronoun
20. RB Adverb
21. RBR Adverb, comparative

22. RBS Adverb, superlative
23. RP Particle
24. SYM Symbol
25. TO to
26. UH Interjection
27. VB Verb, base form
28. VBD Verb, past tense
29. VBG Verb, gerund or present participle
30. VBN Verb, past participle
31. VBP Verb, non-3rd person singular present
32. VBZ Verb, 3rd person singular present
33. WDT Wh-determiner
34. WP Wh-pronoun
35. WP\$ Possessive wh-pronoun
36. WRB Wh-adverb
37. # Pound sign
38. \$ Dollar sign
39. . Sentence-final punctuation
40. , Comma
41. : Colon, semi-colon
42. ( Left bracket character
43. ) Right bracket character
44. " Straight double quote
45. ‘ Left open single quote
46. ‘‘ Left open double quote
47. ’ Right close single quote
48. ’’ Right close double quote