Fine vs. Coarse-grained Categories for POS Tagging
Assignment I
Due: March 5th, 2015

In this assignment you will investigate the ability of a POS tagging system to deal with fine-grained POS categories and coarse-grained POS categories. The goal of the assignment is to run conduct three experiments, analyze the results, and to draw conclusions. This is in effect a quick iteration of what you will be doing for your course project.

There are two parts to this assignment.

NOTE: This is a long document. Please read through this carefully and make sure you understand the requirements before beginning the assignment.

Part I: POS Tagging using fine-grained Penn Tree Bank (PTB) categories
[40 points]

1) Download the Natural Language Toolkit (nltk) or your favorite POS tagger.
2) Use the first 500 sentences in PTB as training sentences and the next 500 as test sentences.
    a. In NLTK you can access the PTB sentences via the Treebank module (using 'from nltk.corpus import treebank')
    b. These first 1000 sentences are available as part of the assignment in Blackboard. Each line is of the form:
        Id sentence w/actual tags
    c. These sentences contain the fine-grained PTB categories.
3) Train your POS tagger using the training sentences.
    a. In NLTK you have a bunch of taggers that you can use. I recommend that you use something simple (e.g. BigramTagger).
4) Use the trained model to predict POS tags for the test sentences.
5) Save the predictions in a tab-separated-file (part-I-predictions.tsv) in the following format:
        Id sentence w/ actual tags sentence w/ predicted tags
6) Evaluate the performance of the tagger on the test sentences. See section on Evaluation below for details on what to report.
7) Analyze the results by inspecting the cases where the tagger makes errors. See Analysis section below for details.

Part II: POS Tagging using Coarse-grained POS categories

Suppose you decided that for your application you only need coarse-grained POS categories. That is you want to know whether something is a noun, verb, adjective, or an adverb. Anything else that doesn't fit this category will be treated as "misc" category.

Tag Map: Fine-to-Coarse POS Tag Mapping

Super Noun:

SNN = {"NN", "NNS", "NNP", "NNPS", "PRP", "PRP$"}

Super Verb:

SVB = {"VB", "VBP", "VBD", "VBN", "VBZ", "VBG"}

Super Adjective:

SJJ = {"JJ", "JJR", "JJS"}

Super Adverb:

SRB = {"RB", "RBR", "RBS"}

You will investigate two ways of building a coarse-grained POS tagger.

Method A: Train on Fine-grained and post-process to coarse-grained. [30 points]

- Build a POS tagger that can predict the fine-grained categories. Follow Steps 2-4 in Part I.
- Convert the predicted (tagger assigned) and actual (original PTB tag) fine-grained tags to coarse-grained tags using the Tag Map given above.
- Save the coarse-grained actual tags and the predictions into a Method-A-predictions.tsv file:

  Id sentence w/ actual coarse tags sentence w/ predicted tags

- Evaluate the performance on the coarse-grained tags.

Method B: Train and test on coarse-grained. [30 points]

- Pre-process the 1000 sentences to convert the PTB tags to the coarse-grained tags using the Tag Map.
- Train and test the POS tagger on the coarse-grained categories directly.
- Save the coarse-grained actual tags and the predictions into a Method-B-predictions.tsv file:

  Id sentence w/ actual coarse tags sentence w/ predicted tags

- Evaluate the performance on the coarse grained tags.

Analyze the performance and provide plausible hypotheses that explain the difference in performance for Methods A and B. Also indicate what tests you can do to verify your hypotheses.

Evaluation

For each round of evaluation you must provide the following measures:

1) Overall Accuracy – The percentage of tokens for which the predicted tag was same as the actual tag. Use fine-grained PTB tags for Part I and coarse-grained tags for Part II.
2) Per Tag Accuracy – Report accuracy on each category in descending order of accuracy.

You can report 1 and 2 in a single table that looks like this:

Original POS Category	Accuracy
NN	92
NNS	89.1
PRP	76
VBZ	58.4
…	
Overall Accuracy	78.9

3) Confusion Matrix – Create a confusion matrix C which records for each actual POS category the types of errors that the POS tagger makes. Entry C_{ij} contains the number of times the POS tagger predicted category j for tokens whose original POS tag is i. If the POS tagger is perfect then only the diagonal will have non-zero entries.

Predicted Tag

Actual Tag	NN	NNS	PRP	VBZ	…
NN	129	74			
NNS	12	154			
PRP					
VBZ					
…					

Analysis

Part I

- Give plausible explanations for the two categories on which the POS tagger performs the worst and the top two categories on which it performs the best. For example, the tagger may get PRP with a high accuracy and may make most mistakes on RBP. Think what factors could affect the performance on a category.
- Briefly (in a couple of sentences) mention what tests you can do to verify if your explanations are indeed true.

Part II

- Did you a-priori (before experimentation) expect Method A or B to perform better? Why? There is no correct answer here. This exercise is to test your ability to articulate your intuitions based on what you've learnt in class.

- Give plausible explanations for the observed differences in overall accuracy between Method A and B. Again there is no correct answer here. The purpose of this exercise is for you to connect concepts we've learnt in class to what you observe in practice.

Items to turn in:

1. **Code** – Any code you had to write for the assignment. This includes:
   a. Code you wrote to invoke the training and test methods in the POS tagger implementation.
   b. Code to convert the fine-grained to coarse-grained tags.
   c. Code to evaluate the tagger performance.

   Please do NOT include the POS tagger code. Document the major parts of your code to clearly identify the sections that correspond to the training, test, and evaluation steps.

   Please include a README that gives an overview of the files in your directory.
2. **Report** – Your report should include evaluation and analysis for Part I and Part II. Use captions for your tables and figures to explain what it contains. The report should not be more than 5 pages long.

3. **Experimental Results** – Turn in the saved predictions files. There should be three files one each for Part I, Method A and B. Your evaluation statistics should line up with the predictions. For example, if I calculated the overall accuracy from the prediction files they should match the accuracy in your report.