IITP-03 Assignment 1: POS Tagging

Assigned: 14 February 2020 11:59 AM
Due: 5 March 2020 11:59 PM

1 Background

POS tagging forms an important part of NLP workflow for most modern day NLP systems. Within the NLP community, POS tagging is largely perceived as a solved problem, or at least well enough solved such that most people don't put much effort into improving POS tagging for its own sake.

But clearly you haven't solved it yet, so you can take a crack at it. It's a good problem for learning about NLP, and that's what we are aiming for in this exercise.

2 Files

Unzip the archive you downloaded from Piazza. You should have the following files:

```
handout
├── hw01.pdf
├── train_hmm.py
├── viterbi.py
├── tag_acc.py
├── ptb.2-21.tgs
├── ptb.2-21.txt
├── ptb.22.tgs
├── ptb.22.txt
└── ptb.23.txt
```

For this assignment, we've provided you the text of the first 23 sections of the Penn Tree Bank (PTB). We've also provided you with the gold standard tags of the first 22 sections as well (One seems to be missing... Hmm...). The first 21 sections are concatenated into `ptb.2-21.txt` and `ptb.2-21.tgs` for convenience. You will use section 22 and 23 for evaluation & testing purposes in later tasks.

3 Task 1: Generating Learning Curves (40 points)

How large of a dataset do you need to make a decent-performing POS tagger that you can ship?

A learning curve is a useful tool that lets you visualize how a model's performance depends on the amount of training data. The learning curve plots a performance measure evaluated on a fixed test set (y-axis) against the training dataset size (x-axis).

3.1 Generating and evaluating a bigram HMM model

Use the provided `train_hmm.py` to generate an intermediate file containing the transmission and emission probabilities from the observed data.

```
$ python train_hmm.py tagfile.tgs textfile.txt model.hmm
```

Then use the provided `viterbi.py` to tag the test set.

```
$ python viterbi.py model.hmm input.txt output.txt
```

Finally use the provided `tag_acc.py` to obtain an accuracy score.

```
$ python tag_acc.py gold-tags.tgs your-tags.txt
```

3.2 Subtask 1: The Curve (20 pts.)

Make a learning curve! For this subtask:

1. Use the first 21 sections of the PTB to generate a bigram HMM model for POS tagging
2. Tag section 22 with the generated model to evaluate its performance
3. Vary the amount of training data provided to the model and re-evaluate its performance
4. Plot a learning curve from your testing (please label axes)

You can vary the amount of training data by creating smaller sub-corpora from the original full corpus through random sampling.

3.3 Subtask 2: Analysis (20 pts.)

From the learning curve that you have generated in the previous subtask:

1. How does the size of the dataset impact the performance of the model/system?
2. How do you think the learning curve will change for datasets of different languages? There is no single 'correct' answer to this question, but it has to make sense!

3.4 Submission

Place your generated learning curve and your analysis in a PDF file named `writeup.pdf`.

There are more written questions later; please append your work to the same PDF document for ease of grading.

4 Task 2: Building a Better System (60 pts.)

Clearly we can do better!^{TM} Now it is up to you to improve the model.

4.1 Subtask 1: Your own improved HMM tagger (40 pts.)

For this task, you will implemement your own HMM tagger, improved upon the provided code. You are constrained to using a HMM, and you can only train on the provided training data (no external resources/libraries allowed). Some relatively simple but cost-effective measures include implementing a trigram HMM, or smoothing the probability estimates.

Feel free to make use of or modify the existing scripts provided for your needs.

It may be helpful to know that many tagging schemes encode tags in such a way that if you replaced every tag by its first letter only, the tags would still be meaningful, only more coarse.

Measure the accuracy of your new model against the same dataset as in the previous task.

Once you feel satisfied with the improvements, run your shiny new tagger on `ptb.23.txt`. You will be graded for this task on the tagging accuracy on this testing data, which will be invisible on your end.

4.2 Subtask 2: Analysis (20 pts.)

1. What modifications did you make?
2. How much improvement did your new model deliver on section 22 (`ptb.22.txt`)?
3. Why do you think that these modifications improved the accuracy of this labelling task?

4.3 Reminders

Please check that your improved model is actually an improvement.

On the other hand, a positive improvement is an improvement no matter how small ☺

4.4 Frequently Encountered Problem

Your improvements may reduce the throughput of your tagging program by an order of magnitude. While we do not grade you on your code efficiency or speed, in the interest of your personal productivity, consider making use of your learning curve to sample a smaller but appropriate amount of data to test your system.

4.5 Submission

Name your tagging program `tagger.py`. Your program should take in three commandline arguments similar to `viterbi.py`.

```
$ python tagger.py model.hmm input.txt output.txt
```

If you modified your `train_hmm.py` (e.g. to print trigram data instead of bigrams), please include it as well. Otherwise the default handout version will be used.

Output your tags to a file `ptb.23.tgs`.

Place your analysis (answers to the questions) in `writeup.pdf`.

5 Bonus Task: Better Evaluation (5 points)

Write a new evaluation script that calculates quantities that help elucidate the differences between the two models further, by looking at errors in a *more fine-grained way*.

In your answer, explain what your script does and show its output.

5.1 Submission

Name your evaluation script `my_tag_acc.py`.

Dump the output to `eval.txt`.

Place your explanation in `writeup.pdf`.

6 Final Submission

Zip the following files up as `hw01_{your_andrew_id_here}.zip` and submit the compressed archive to Autolab:

1. `writeup.pdf`
2. `tagger.py` (Your tagger from Task 2)
3. `ptb.23.tgs` (Your output from Task 2)
4. `train_hmm.py` (optional modified HMM trainer)
5. `my_tag_acc.py` (optional bonus task)
6. `eval.txt` (optional bonus task)
7. `README` (optional)

If you have consulted with other students or resources, you must indicate it in the `README` file.