Natural Language Processing [CS4120/CS6120] Assignment 1

Instructor: Professor Lu Wang

Deadline: February 13th at 11:59pm on Blackboard

For the programming questions, you can use Python (preferred), Java, or C/C++. You also need to include a README file with detailed instructions on how to run the code and commands. Failure to provide a README will result in a deduction of points.

This assignment has to be done individually. If you discuss the solution with others, you should indicate their names in your submission. If you use ideas from any online forums, you should also list the URLs. **Violation of the academic integrity policy is strictly prohibited, and any plausible case will be reported once found**.

1 Naive Bayes for Text Categorization (10 points)

Given the following short documents (1-8), each labeled with a class:

1. banana, carrot, cucumber, pea : **vegetable**
2. lotus, pea, rose, lily, hibiscus : **flower**
3. banana, pea, potato, basket : **vegetable**
4. hibiscus, grape, school, mango, apple : **fruit**
5. hibiscus, lotus, lily, apple, banana : **fruit**
6. lotus, hibiscus, pea, rose : **flower**
7. lily, hibiscus, rose, cucumber : **flower**
8. apple, mango, grape, rose : **fruit**

And documents:

1. D1 : rose, lily, apple, carrot
2. D2 : pea, basket, lotus, grape

Your task is to compute the most likely class for D1 and D2. You will start with building a Naive Bayes classifier and using add-λ smoothing (with $\lambda = 0.1$). For this question, show your work of computing prior, conditional, and posterior probabilities. Do not write/submit code this question.

2 Word Sense Disambiguation (10 points)

Given the following sentences:
Her right arm lay in the cushioned parapet before her. She raised her hand, and made a light, quick movement toward the right. No one but her lover saw her. Every eye but his was fixed on the man in the arena. He turned, and with a firm and rapid step he walked across the empty space.

2.1 [5 points]

Using WordNet (`http://wordnetweb.princeton.edu/perl/webwn`) to determine the total number of senses for each *open* class word. No need to write/submit any code. You should use the base-form of each word (e.g. *raise* for *raised*) when calculating the number of senses.

2.2 [5 points]

For each sentence, determine how many distinct combinations of senses exist based on results from question 2.1.

3 Language Modeling (35 points)

Your task is to train trigram *character-level* language models from the following English training data and then test the models:
`http://www.nltk.org/nltk_data/packages/corpora/gutenberg.zip`

For all questions, please submit both the code and output.

Data preprocessing instructions:

1. Remove blank lines from each file.
2. Replace newline characters with spaces.
3. Remove duplicate spaces.
4. Replace characters in training set that appear ≤ 5 times as "UNK".

3.1 [10 points]

Across all files in the directory (counted together), report the unigram, bigram, and trigram character counts. You can submit them in separate files.

3.2 [10 points]

For the given test dataset:
`https://www.dropbox.com/s/s1u6tvw8b69c1nw/test_data.zip?dl=0`.
Calculate the perplexity for each file in the test set using linear interpolation smoothing method. For determining the λs for linear interpolation, you can divide the training data into a new training set (80%) and a held-out set (20%) , then using grid search method.

- First, report the λs chosen from the grid search, and explain why it's chosen (i.e. leveraging the perplexities achieved on the held-out set).
- Report the perplexity for each file in the test set.
- Some files in test data are not English. Identify them as follows: Using the calculated perplexity scores, report the names and scores of the fifty files with highest perplexity score. Manually check them and report if they are French or not.

When you report perplexities, you can output them into a file with the following format:
file name, score
file name, score
...
file name, score

3.3 [10 points]

Build another language model with add-λ smoothing (use $\lambda = 0.1$).

- Report the perplexity for each file in the test set.
- Then identify non-English documents as follows: Using the calculated perplexity scores, report the names and scores of the fifty files with highest perplexity score. Manually check them and report if they are French or not.

When you report perplexities, you can output them into a file with the following format:
file name, score
file name, score
...
file name, score

3.4 [5 points]

Based on your observation from above questions, compare linear interpolation and add-lambda smoothing by listing out their pros and cons.

4 POS Tagging - HMM (45 points)

The training dataset is a subset of the Brown corpus, where each file contains sentences of tokenized words followed by POS tags, and where each line contains one sentence:
`https://www.dropbox.com/s/f0b6vovfcmzaez0/brown%20_corpus_spring2019.zip?dl=0`
The test dataset (which is another subset of the Brown corpus, containing tokenized words but no tags) is the following:
`https://www.dropbox.com/s/n8xdkhfaonja16r/Test_File.txt?dl=0`
Information regarding the categories of the dataset can be found at:
`https://www.dropbox.com/s/ujnz04e62e9603j/CONTENTS.txt?dl=0`

Your task is to implement a part-of-speech tagger using a bigram HMM. Given an observation sequence of n words w_1^n, choose the most probable sequence of POS tags t_1^n.

[Note: During training, for a word to be counted as unknown, the frequency of word in training set should not exceed a threshold (e.g. 5). You can pick a threshold based on your algorithm design.]

4.1 [5 points]

Obtain frequency counts from the collection of all the training files (counted together). You will need the following types of frequency counts: word-tag counts, tag unigram counts, and tag bigram counts. Let's denote these by $C(w_i, t_i)$, $C(t_i)$, and $C(t_{i-1}, t_i)$ respectively. Report these quantities in different output files.

To obtain the tag unigram counts and the tag bigram counts, you will need to separate out each tag from its word. For each sentence found in the training data, add a start token and an end token to the beginning and the end of the sentence.

4.2 [5 points]

A transition probability is the probability of a tag given its previous tag. Calculate transition probabilities of the training set using the following equation:

$$P(t_{i-1}, t_i) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})}$$

4.3 [5 points]

An emission probability is the probability of a given word being associated with a given tag. Calculate emission probabilities of the training set using the following:

$$P\left(w_i \left| t_i\right.\right) = \frac{C\left(w_i, t_i\right)}{C\left(t_i\right)}$$

4.4 [10 points]

Generate 5 random sentences using the previously learned HMM. Output each sentence (with the POS tags) and its probability of being generated.

4.5 [20 points]

For each word in the test dataset, derive the most probable POS tag sequence using the Viterbi algorithm; pseudo-code can be found in the textbook `https://web.stanford.edu/~jurafsky/slp3/ed3book.pdf` under **Figure 8.5**. Submit the output in a file with the following format (where each line contains no more than one pair) :

<sentence ID=1>
word, tag
word, tag
...
word, tag
<EOS>
<sentence ID=2>
word, tag
word, tag
...
word, tag
<EOS>