Assignment 1

2024 Fall Statistical NLP – CSED523

Submission Policy

Please submit your answer and justification for **Part A** in a PDF file format named as ***"studentid_name.pdf"*** (e.g., 20232586_DaeheeKim.pdf).

If the justification is not appropriate, you will not receive any points even if the answer is correct.

For **Part B**, submit the provided files in a ZIP archive named ***"studentid_name.zip"*** (e.g., 20232586_DaeheeKim.zip).

Your final submission to PLMS should include **one PDF file** and **one ZIP file**.

For any further questions regarding the assignment, please contact TA via email.

TA e-mail: andrea0119@postech.ac.kr

Late Submission Policy

Please note that late submissions will incur a penalty of 20% for each 24-hour period after the deadline.

(Up to 24 hours late: 20% deduction, Up to 48 hours late: 40% deduction, Up to 72 hours late: 60% deduction …)

Part A [60 points]

In this part, you will be given questions about statistical NLP, which we covered in the lectures.

Unless explicitly stated otherwise, you must include a justification for your answers to receive full points. Read the questions carefully and provide both your answers and justifications.

---

1 Foundations [7 points]

1.1. Define entropy in the context of language modeling. How does it measure the uncertainty of a language model? – (Explain it by suggesting the difference between high entropy and low entropy) (4 points)

1.2. Explain by taking the example for 6 different levels of linguistic processing from phonetics/orthography, phonology, morphology, syntax, semantics, discourse/pragmatics. Take the example sentence: “You may say I'm a dreamer, but I'm not the only one.” (3 points)

2 Regular Expressions and Edit Distance [25 points]

2.1. Write regular expressions for the following languages. (13points) **(No Justification)**

a. The set of all alphabetic strings; (1 point)

b. The set of all lower-case alphabetic strings ending in a *b*; (2 points)

c. The set of all strings from the alphabet *a, b* such that each a is immediately preceded by and immediately followed by a *b*; (2 points)

d. The set of all binary strings (composed of 0 and 1 only) such that the string starts and ends with the same digit and contains no consecutive repeating characters; (3 points)

e. The set of all strings with two consecutive repeated words; (ex. NLP NLP) (2 point)

f. all strings that start at the beginning of the line with an integer and that end at the end of the line with a word; (By “word”, we mean an alphabetic string separated from other words by whitespace) (3 points)

2.2. Calculate character-level minimum edit distance between given sequence and sequence A, B and C. Follow indicated edit cost for each sub problems. (12 points) (Hint: **You can ignore common words in each sequence.**)

Given Sequence: Statistical NLP is interesting

Sequence A: Deep NLP is interesting
Sequence B: Statistical NLP is difficult
Sequence C: Deep NLP is difficult

a. Edit distance between given sequence between Sequence **A**

[Insertion = 1, Deletion = 1, Substitution = 1] (3 points) (**Table for first word**)

b. Edit distance between given sequence between Sequence **B**

[Insertion = 5, Deletion = 1, Substitution = 1] (3 points) (**No Justification**)

c. Edit distance between given sequence between Sequence **C**

[Insertion = 1, Deletion = 5, Substitution = 1] (3 points) (**No Justification**)

d. Edit distance between given sequence between Sequence **A**

[Insertion = 1, Deletion = 1, Substitution = 5 (3 points)] (**No Justification**)

3 N-gram Language Models [18 points]

3.1. Suppose we build a simple **unigram** language model (no add-one smoothing, no unknown words) from the following snippet of text. (14 points)

<s> the cat wore the hat </s>
<s> the hat was on the cat </s>
<s> i want a blue hat the cat said </s>
<s> but alas the cat’s hat was red </s>

a. Suppose we see the word ‘the’, what is the probability that the next word is ‘hat’, i.e. P(hat|the)? (Don’t forget to include the tokens <s> and </S> in your counts.) (3 points)

b. What if we adapt “Laplace Smoothing” without changing other conditions from a? (3 points)

c. What if we adapt “Good-Turing Smoothing” without changing other conditions from a? (3 points)

d. Discuss which smoothing method is more suitable for different types of corpora. Provide examples of scenarios where one smoothing method may be preferred over the other. (Please suggest scenarios for both smoothing method) (5 points)

3.2. Suppose we we train a trigram language model with Laplace smoothing on a given corpus. The corpus contains V word types. Express a formula for estimating P(w3|w1,w2), where w3 is a word which follows the bigram (w1,w2), in terms of various n-gram counts and V. Use the notation c(w1,w2,w3) to denote the number of times that trigram (w1,w2,w3) occurs in the corpus, and so on for bigrams and unigrams. (4 points)

4 Text Classifications [10 points]

4.1. Given the following short movie reviews, each labeled with a genre, either comedy or action:

Class(Comedy) : fun, couple, love, love
Class(Action) : fast, furious, shoot
Class(Comedy) : couple, fly, fast, fun, fun
Class(Action) : furious, shoot, shoot, fun
Class(Action) : fly, fast, shoot, love

What would be the class of a new movie that contains **“fast, couple, shoot, fly”** in its review? Assume a naïve binarized Bayes classifier and use add-1 smoothing for the likelihoods. (10 points)

Part B [40 Points]

In this part, you will implement several approaches for part-of-speech (POS) tagging. To build your POS tagger, you will use data from the Penn Treebank that includes POS tagged sentences.

Please Submit “test_y.csv” and “pos_tagger.py” only bundled together in a zip file named “studentid_name.zip”.

---

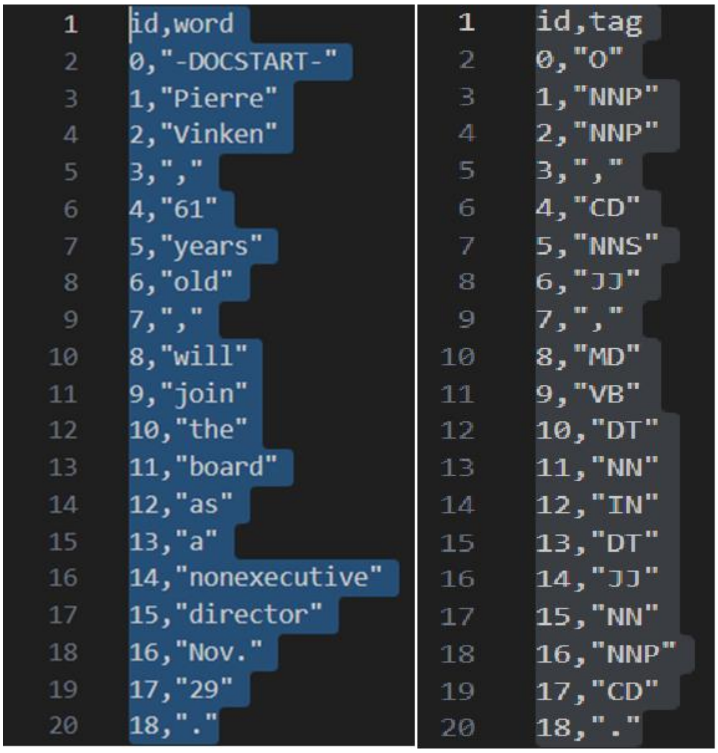
1 POS Tagging with Trigram HMM [40 Points]

Training, Development and Test Data

This dataset contains almost a million words of text from the Wall Street Journal. The sentences in the dataset are written out word-by-word in a flat, column format in the **_*_x.csv_** files, where individual documents are separated by **-DOCSTART-** tokens. Each of these words has a corresponding POS tag, located in the **_*_y.csv_** files. The test POS tags file, **_test_y.csv_**, is missing — your job is to output a reasonable sequence of tags to recreate this file **_test_y.csv_** using the sequences of words in the **_test_x.csv_** file.

For example, “Pierre Vinken, 61 years old, will join the board as a nonexecutive director nov.29.” is labeled as seen in left side of below figure (train_x.csv). Corresponding POS tags are aligned in train_y.csv, listing the part-of-speech for each word in train_x to facilitate further analysis and training of the model.

Your job is implementing HMM to get “test_y.csv” file from “test_x.csv”. Please read the comments in the source code carefully and complete the implementation.



```
id,word
0,"-DOCSTART-"
1,"Pierre"
2,"Vinken"
3,","
4,"61"
5,"years"
6,"old"
7,","
8,"will"
9,"join"
10,"the"
11,"board"
12,"as"
13,"a"
14,"nonexecutive"
15,"director"
16,"Nov."
17,"29"
18,"."
```

```
id,tag
0,"O"
1,"NNP"
2,"NNP"
3,","
4,"CD"
5,"NNS"
6,"JJ"
7,","
8,"MD"
9,"VB"
10,"DT"
11,"NN"
12,"IN"
13,"DT"
14,"JJ"
15,"NN"
16,"NNP"
17,"CD"
18,"."
```

Tips for programming

Installation

```
unzip assn1.zip
cd assn1
conda env create -f assn1.yaml
conda activate assn1
```

Training & Inference

```
python pos_tagger.py
```

Evaluation

```
python evaluate.py -p data/test_y.csv -d data/test_x.csv
```

1. Fill the functions marked as ## TODO in the “pos_tagger.py”. (10 points)

2. Attach screenshots of the code execution that displays running times (5 points)

3. Choose one mode (unigram/bigram/trigram). Describe your chosen mode’s implementation with HMM diagram. (10 points)

4. After running the evaluation, please attach a screenshot of the F1 score results. Additionally, submit 'test_y.csv' bundled together with 'pos_tagger.py'. (The score will vary based on the F1 Score.) (15 points)

* **Justifications or comments are required for your code implementation, similar to the handwriting problems.**