HACETTEPE UNIVERSITY

COMPUTER ENGINEERING

BBM497 NATURAL LANGUAGE PROCESSING LAB.

ASSIGNMENT 2

Name: Kürşat Aktaş

Student Number: 21227949

Subject: Hidden Markov Models and Part-of-Speech Tagging

Content

1 Introduction

In this paper I tried to explain what I had done and what I had observed with this assignment.

2 File Hierarchy and Requirements

Before run the given program be sure that file hierarchy look like this;

- ex2.py
- input_tokens.txt
- test_set.txt
- brown/
    - text files

After that, program can be run with below command;

- python3 ex2.py

If program runs without error it will generate two output files ('output_tokens.txt' and 'output_set.txt') in the same directory with ex2.py.

3 Language and HMM Model

When designin my model I have taken the following decisions;

a) I didn't used sentence and word boundaries. These tokens (!.?...) used for sentence seperations and single space used for word seperation. In the HMM side, I observed that two ('.' and '.-hl') POS tags used for above tokens. So when deciding sentence beginnig I used these two tags.
b) My language model is not case sensetive. Before doing any process program firstly converts all letters to lowers case.

4 Tasks

Firstly I trained my program with brown corpus. My main data structure look like;

```
words_tags => [ "word1" => [ "tag1" => "word1_tag1_count"],
                             [ "tag2" => "word1_tag2_count"],
                             …
               "word2" => ….
              ]
```

With the help of above structure I can access most of the informations when I need. I'm also storing occurence count of each tags and frequencies of tags bigrams.

4.1 Task 1

Program reads brown corpus and parses word tag pairs successfully.

4.2 Task 2

2.1- *initial_tag_prob()* function calculates the probability of a given tag of being at the beginning of a sentence

2.2- *transition_prob()* function calculates the probability of succession tags

2.3- *emission_prob()* function calculates the emission probability of given word by given tag

4.3 Task 3

Program successfully reads ‘input_tokens.txt’ and outputs tagged sentence to ‘output_tokens.txt’.

4.4 Task 4

Program successfully reads ‘test_set.txt’ and outputs tagged sentence to ‘output_set.txt’. *viterbi()* function finds the best tag sequence for given sentence.