CSCI544: Homework Assignment №2

Due on Feb 08, 2023 (before class)

Introduction

This assignment gives you hands-on experience on using HMMs on part-of-speech tagging. We will use the Wall Street Journal section of the Penn Treebank to build an HMM model for part-of-speech tagging. In the folder named *data*, there are three files: *train*, *dev* and *test*. In the files of *train* and *dev*, we provide you with the sentences with human-annotated part-of-speech tags. In the file of *test*, we provide only the raw sentences that you need to predict the part-of-speech tags. The data format is that, each line contains three items separated by the *tab* symbol '\t'. The first item is the index of the word in the sentence. The second item is the word type and the third item is the corresponding part-of-speech tag. There will be a blank line at the end of one sentence.

Task 1: Vocabulary Creation (20 points)

The first task is to create a vocabulary using the training data. In HMM, one important problem when creating the vocabulary is to handle *unknown* words. One simple solution is to replace rare words whose occurrences are less than a threshold (e.g. 3) with a special token '< unk >'.

Task. Creating a vocabulary using the training data in the file *train* and output the vocabulary into a txt file named *vocab.txt*. **The format of the ocabulary file is that each line contains a word type, its index in the vocabulary and its occurrences, separated by the *tab* symbol '\t'. The first line should be the special token '< unk >' and the following lines should be sorted by its occurrences in descending**

order. Note that we can only use the training data to create the vocabulary, without touching the development and test data. What is the selected threshold for unknown words replacement? What is the total size of your vocabulary and what is the total occurrences of the special token '< unk >' after replacement?

Task 2: Model Learning (20 points)

The second task is to learn an HMM from the training data. Remember that the solution of the emission and transition parameters in HMM are in the following formulation:

$$\begin{aligned} t(s'|s) &= \frac{\text{count}(s \to s')}{\text{count}(s)} \\ e(x|s) &= \frac{\text{count}(s \to x)}{\text{count}(s)} \end{aligned}$$

where $t(\cdot|\cdot)$ is the transition parameter and $e(\cdot|\cdot)$ is the emission parameter.

Task. Learning a model using the training data in the file *train* and output the learned model into a model file in json format, named *hmm.json*. The model file should contains two dictionaries for the emission and transition parameters, respectively. The first dictionary, named *transition*, contains items with pairs of (s, s') as key and $t(s'|s)$ as value. The second dictionary, named *emission*, contains items with pairs of (s, x) as key and $e(x|s)$ as value. How many transition and emission parameters in your HMM?

Task 3: Greedy Decoding with HMM (30 points)

The third task is to implement the greedy decoding algorithm with HMM.

Task. Implementing the greedy decoding algorithm and evaluate it on the development data. What is the accuracy on the dev data? Predicting the part-of-speech tags of the sentences in the test data and output the predictions in a file named *greedy.out*, in the same format of training data.

Task 4: Viterbi Decoding with HMM (30 Points)

The fourth task is to implement the viterbi decoding algorithm with HMM.

Task. Implementing the viterbi decoding algorithm and evaluate it on the development data. What is the accuracy on the dev data? Predicting the part-of-speech tags of the sentences in the test data and output the predictions in a file named *viterbi.out*, in the same format of training data.

Submission

Please follow the instructions and submit a zipped folder containing:

1. A txt file named *vocab.txt*, containing the vocabulary created on the training data. The format of the vocabulary file is that each line contains a word type, its index and its occurrences, separated by the *tab* symbol '\t'. (see task 1).

2. A json file named *hmm.json*, containing the emission and transition probabilities (see task 2).

3. Two prediction files named *greedy.out* and *viterbi.out*, containing the predictions of your model on the test data with the greedy and viterbi decoding algorithms. You also need to submit your python code and a README file to describe how to run your code to produce your prediction files. (see task 3 and task 4).

4. A PDF file which contains answers to the questions in the assignment along with brief explanations about your solution.