NTMI — Exercise 5 — HMM Project

March 1, 2018

You will report on an experiment based on *lab4*, for that the only implementation effort required is an *iterative* version of the Viterbi algorithm. Note that in lab4 we provided a *recursive* version of it. Below we present your tasks.

1 Implementation (2 points)

You can reuse all functions and classes from your completed notebook *lab4*, but to score the 2 points this section is worth, you should replace the *recursive* implementation of the Viterbi algorithm we provided with your own **iterative implementation**.

2 Jupyter notebook with experiments (3 points)

Now you will use your notebook to report on a complete experimental setup. You should use markdown cells to explain your steps and you should comment your python code. Here you do not need to explain the model formally, this was already done in lab4. It is enough to simply explain your thoughts, for example:

- "*now I am estimating the model parameters using the PTB training set*";
- "*now I am searching through a combination of values for the smoothing parameters*";
- "*now I am running the Viterbi algorithm to predict tag sequences*".

We provide 3 files that split the PTB data from nltk into training, development, and test sets. These files contain 0-based positions of the sentences that belong to each set.

Training For estimating model parameters by maximum likelihood, you should use the sentences corresponding to those indicated in `training.ids`.

Development You will be training Laplace-smoothed HMMs, and you must recall that Laplace smoothing requires a constant which we call a *pseudo count*. That constant is a parameter of our smoothing technique. With respect to our model design, we call that constant a *hyperparameter*. Each choice of hyperparameter essentially leads to a new instance of our HMM model.

You should search for good hyperparameters for transition and emission distributions. You should tune 2 pseudo counts, namely, α shared across all transition distributions, and β shared across all emission distributions. Consider values such as $\{0.01, 0.1, 1, 10\}$.

In order to compare models trained with different hyperparameters you can use perplexity (or tagging accuracy) on a development set. A *development set* (or *validation set*) is a small set that is neither part of the training set neither part of the test set. We reserve such data to validate some of our design choices before testing our model for real. In our case, we are talking about the choice of α and β. We then use perplexity (or tagging accuracy) to decide which configuration of hyperparameters works best. This is called *model selection*. Once we have decided, we select our best model and report test results in terms of perplexity (or tagging accuracy).

Concretely,

- train a number of models using different values of α and β;
- evaluate performance on development set (sentences corresponding to those in `development.ids`) in terms of both quality metrics we know so far (i.e. perplexity and tagging accuracy);
- select you best models, namely, the best one according to perplexity and the best one according to tagging accuracy (it's not necessarily the case that the same model will perform best under both metrics).

Test For your best configurations of hyperparameters (according to each criteria, namely, perplexity and accuracy) report perplexity and tagging accuracy on test set (sentences corresponding to those in `test.ids`).

Examples Print a few examples (e.g. 3) of sentences, gold-standard tag sequences, and the predictions by your best models.

3 Report (5 points)

You should write a report structured like a research paper. We provide you with an overleaf template for that.[1] The template is from a conference on computational linguistic and natural language processing—you do not have to follow their instructions because those instructions are specific to a certain conference.

You are expected to have sections reflecting the following:

[1] https://www.overleaf.com/14157021zhywsnnmqrpw

- abstract
- introduction: what you are trying to do (e.g. language modelling, POS tagging);
- model: present your model formally (e.g. this is the HMM, it factorises like this and that, this is the probability of a sentence under the HMM); if you want to draw a graphical model you can use for example tikz-bayesnet;[2]
- parameter estimation: explain your estimation procedure (e.g. we use a dataset of observations and count this and divide by that);
- experiments: explain your data (report some details such as number of unique words/tags for training/development/and test), present results (both development results and test results) using nice visual aids (e.g. tables and plots), and comment on your decisions.
- conclusion: summarising remarks

You report should have 2-3 pages in length.

4 Submission

Upload your *notebook of experiments* and *report in pdf* on Blackboard before *March 9 2018 at 23:59*.

[2] https://github.com/jluttine/tikz-bayesnet