NLP & ML Assignment 1: HMM / Embeddings

For this assignment, you have two choices:

(a) build an HMM part-of-speech tagger,

or

(b) implement a word embedding algorithm.

A. Tagger

Implement the tagger in Python. You're welcome to use any libraries you like, as long as it's not e.g. "import hmm" - the tagged code has to be written by you yourself.

The data

Download a PoS-tagged corpus from UD (universaldependencies.org). Report which dataset you use. You can select anything you like. What kind of text is it - news, web, conversation, ..?
Make sure you split it into train and test parts. Choose how large to make them, and report this.

The technique

Train a model using the training data, and evaluate over the test data
Report both the token accuracy and complete-sentence accuracy.

The analysis

What did the tagger do well? What kind of mistakes did it make?
Report the accuracy using 10%, 20%, 30%, …. 90% of the training data. Did you have enough data?
How do you handle unknown words?
Find what the most common errors were, and give some examples of them - as well as some examples of the tagger working correctly.

B. Embeddings

Implement a word embedding, like word2vec (here's a useful paper: https://arxiv.org/abs/1402.3722), or GloVe (https://www.aclweb.org/anthology/D14-1162.pdf).

The data

Find an English dataset. A good place to start is here, https://www.english-corpora.org/ . Don't forget to tokenize, if that hasn't been down! nltk.word_tokenize() is fine enough, but you can use other things too.

The technique

Implement a word embedding technique for converting the corpus into distributional vector representations of each word. You can use the skip-gram negative-sampling technique presented in the lectures, or any other technique that you like. You should implement the code yourself. Don't forget to save the vectors in tab-separated format when your program's done.

The analysis

We need to know how well the embedding system worked. There are benchmarks for this. One of the bigger benchmarks is called BATS, which tests many kinds of lexical semantic relations. The details are here https://vecto.space/projects/BATS/. Load your embeddings into BATS and run as much of the word analogy task as you can (https://vecto.readthedocs.io/en/docs/tutorial/evaluating.html#word-analogy-task). Think about how well the embedding implementation

performed, e.g. did you have enough data? What happened with out-of-vocabulary words? Which lexical semantic relations were picked up well, and which were not?

For both (a) and (b): Assignment hand-in

Essay
Description of what you did and why, describing your general code, and the answers to all of the above questions in the work description. 500-1000 words.

Code
Include your code. A link to a Colab notebook is best. Test the whole notebook first. I will run it myself.

How?
By Innopolis Moodle. I hope it will be working.

When?
By the end of Saturday 14 March 2020. Good luck!