Master in Artificial Intelligence

Language Modelling – Year 2022/23

Lab Assignment

In this exercise we will implement toyLM, a small language model that will be used to generate natural language texts for a specific topic. In order to help the development of the code we include, in addition to this problem definition, a `LM_lab_guide.pdf` file and a folder with Python notebook materials and examples of components that will be helpful in the implementation of the assignment. Specifically, you are asked to implement two language models, based on two different architectures:

1. toyLM_ngram: A non-neural model based on n-grams and Markov chains. The model shall support a variable previous context size (n-grams, with n≥1).

2. toyLM_LSTM: A neural model based on long short-term memory networks (LSTMs). The model should be a `tensorflow.keras.Sequential` object. A default architecture is proposed for the network: (i) a first `tensorflow.keras.layers.Embedding` layer that transforms words into vectors of size 20, (ii) a `tensorflow.keras.layers.LSTM` layer with output dimension 64 and a (small) `max_seq_length` of 10, and (iii) a `tensorflow.keras.layers.Dense` output layer, computing the probability distribution for the next word through a softmax function. You can consider other dimensions of the network if you wish in order to generate a language model.

The functionalities to be supported by both toyLMs are:

1. **To train the models** using a training set (`train.txt`) and a development set (`dev.txt`), the latter used for internal evaluation. Specifically, given a prior context in a sentence, the models should be able to learn to compute a probability distribution and predict what the next word in that sentence should be.
2. **To evaluate the trained models**. The trained model will be used to calculate the perplexity on a set of sentences (`test.txt`), in order to estimate how effective our model is.
3. **To generate new sentences automatically.** The trained model should be able to generate new sentences from an initial context (e.g., randomly set). This functionality should be configurable so that: (i) the next most probable word is always generated, (ii) the next word is generated based on the probability distribution generated by the model in the current context, (iii) the next word is generated based on the 50 most probable words of that probability distribution. Try to automatically generate, for each strategy, at least 10 sentences of approximately 15 to 30 words in length.

As an example, we have included with this description a corpus of speech excerpts from Queen Elizabeth II of Great Britain (`HerMajestySpeechesDataset`), already divided into `train.txt`, `dev.txt` and `test.txt` files, which you can use to train the toyLM models for that domain. In addition, consider training a second toyLM for another domain of your choice. As a suggestion, a possible option would be to consider one of the books (or excerpts) freely available at https://www.gutenberg.org/ by downloading the plain text version.

Submission

Deliverables for this assignment must include, alongside the code, a brief but descriptive user manual, indicating how to use the code to train, evaluate and generate sentences. It should also include an analysis of the differences between both toyLM language models, their advantages and disadvantages, as well as the design decisions made during development.

The assignment will be carried out in groups of two.

Assignment deliverables must be uploaded to the virtual campus of the university in which both students are enrolled, in the section provided for that purpose, no later than Thursday May 4, at 23:59. Only one member of the group must upload the deliverables.

The students will defend their work in the last lab class of the bimester (May 9).