Master in Artificial Intelligence

Language Modelling – Year 2024-2025

Lab Assignment – Neural models for word vector representation

In this exercise, we will develop a neural model for generating vector representations of words, traditionally known as *word embeddings*, from "large" text corpora written in a natural language. These vector representations are fundamental to a variety of NLP tasks, and are the origin of current language models such as ChatGPT or Llama, as they capture the semantics of words and their contextual relationships in the form of vectors. These same fundamentals apply to current language models, but on a much larger scale. Specifically, we will implement a simple model that focuses on predicting a word given its surrounding context, e.g., "I like diving into the [MASK] pool every morning.", where the masked token could be "swimming".

The exercise consists of training a variety of word-embeddings with different parameters, and then perform the following tasks:

1. Explore some word vectors using cosine similarity, dimensionality reduction and data visualisation techniques.
2. Initialise a text classification model with your own embeddings to observe the impact in its performance.

For more details on the specifics of the exercise, you are recommended to consult the document *lm_guide.pdf*

Dataset: The dataset for training the model is provided along with the materials for this exercise. It consists of 30,000 English-language newspaper sentences in plain text format. You will need to pre-process the data and create training samples from it. For enhanced model performance, you can also incorporate supplementary training data (see below).

Word-embeddings model to be implemented:

- **Model based on the prediction of words given their context:** The model will take as input the context (*n* words both preceding and following) of a given word (target), and try to predict this target word. Therefore, the input to the model will be a set of context words, and the output will be the target word:

  Sentence: "I like diving into the swimming pool every morning"
  Examples (*n*=2):
  Context: "like diving * the swimming", target word: "into"
  Context: "diving into * swimming pool", target word: "the"
  Context: "into the * pool every", target word, "swimming"

Architecture: You should implement a neural network architecture that includes:

1. An input layer that accepts word(s) as input.
2. An Embeddings layer that associates each word ID with a vector, which will be updated during training. The values of this layer will be the embeddings of each word.
3. One or more hidden layers processing these inputs.
4. An output layer that generates the predictions for the model.

Experiments: It is suggested to experiment with at least a context size of 2 (both right and left), and to analyse the results of these variations, e.g., increasing it up to 5, 8, or 10. You could also learn embeddings of different dimensions (50, 100, 300, etc.).

Required Functionalities:

1. Train the models using a training set, and validate them with a development set for adjustment of hyperparameters.
2. Qualitative analysis of the results:
    a. Semantic Similarity: For the target words, you are required to calculate their semantic similarity in relation to other words in the vocabulary. This will be done using a specific metric, namely cosine similarity, which compares the vectors generated for each word, and measures the semantic closeness between two words. For each target word, the 10 most semantically similar words (based on cosine similarity) must be displayed, and the most relevant results should be discussed in the report.
    b. Visualisation using t-SNE: To gain a visual understanding of how words are grouped based on their distributional representations, the t-SNE (t-Distributed Stochastic Neighbour Embedding) dimensionality reduction technique must be used. This technique represents word vectors in a two- or three-dimensional space, making it easier to identify groups of words with similar or related meanings. The resulting visualisation will be a graph displaying these groupings. Example code is provided with the materials for this exercise. Again, you are required to visualise the target words using the *visualize_tsne_embeddings* and *visualize_all_tsne_embeddings* functions. For *visualize_tsne_embeddings*, you must generate two graphs: (i) a visualisation of the target words before training the network and (ii) a visualisation of the target words after training the network.
3. Quantitative analysis of the models: You should analyse the impact of using your pre-trained word embeddings in a text classification model, by comparing their performance with a baseline model trained from scratch. You will be provided with example code for training the classification model without pre-trained embeddings. Using this as a reference, you will integrate your pre-trained embeddings into the model and evaluate how they affect classification accuracy and overall performance. Your analysis should highlight the benefits and potential drawbacks of using pre-trained embeddings, as well as the influence of their parameters (e.g., dimensionality, window size, training epochs, etc.).

Additional experiments:

1. The provided corpus was downloaded from the *Leipzig Corpora Collection.*[1] It is part of the 30K dataset from the 2024 English News collection. To improve model performance, you can train using a larger corpus (e.g., 100K, 300K, 1M), or by combining multiple files. Note that using a larger corpus will notably increase training time.
2. Finally, you may want to compare your best word-embeddings to publicly available models. In this case, you can download official vectors from *fastText*[2] or download models from gensim such as by using: *gensim.downloader.load("word2vec-google-news-300").* You can initialise your text classification model with different embeddings and compare their performance.

Submission

The documentation should include a short manual providing instructions on how to run the code for training and evaluating the models, an analysis of the results obtained, and a discussion of the differences observed in relation to the parameters. The documentation should not exceed 3 pages.

The exercise will be carried out in pairs and will follow the same monitoring and defence process outlined in the original statement.

The exercise must be uploaded to the virtual campus, in the designated section, no later than May 16 at 23:59. Only one group member should submit the practical.

The compulsory defence will take place during the last session of the course (May 20). Students who do not attend the defence, unless they have a valid justification, will receive a mark of 0 for the practical.

[1] https://wortschatz.uni-leipzig.de/en/download/English#eng_news_2024

[2] https://fasttext.cc/docs/en/crawl-vectors.html