ECS7001P - NN & NLP
Assignment 1: Word Representation, Text Classification, Machine Translation, and Pre-Trained Transformers

January 24th, 2023

Text classification and machine translation are among the most popular NLP applications, particularly in an industrial setting. In this assignment, you will first implement a text classifier capable of performing a real-world task, and then a machine translation system. You will reach these goals by stages, each achieved during one lab session:

- In Part A of the assignment, to be carried out in Lab 1, week 1 (24th January 2023) you will learn how to train the simpler form of word embeddings using `word2vec`. [15 points]
- In Part B (Lab 2, week 2) you will learn how to train more complex word embeddings using LSTMs and use them in a text classifier. [15 points]
- In Part C (Lab 3, week 3) you will implement different approaches to text classification, including CNNs, and experiment with using word embeddings within those. [20 points]
- In Part D (Lab 4, Week 4) you will start developing a machine translation system,...
- which you will complete in Lab 5, week 5. [30 points total]
- Finally, in Part E (Lab 6, week 6) you will learn how to use pre-trained BERT models and apply them to a familiar task (aspect-based sentiment analysis) [20 points].

When all parts of the assignment are completed, you will have to submit two things:

- A PDF report describing what you did (instructions below);
- Your completed Python code.

The deadline for returning all completed parts of the assignment (Parts A, B, C, D and E) is **10:00:00 Thursday 9th March 2023**.

Part A: Word Embeddings with Word2Vec [15 points]

For this part of the assignment, worth 15 marks in total, you will have to carry out the steps specified by the Lab script ("Lab 1: Skip-gram Model for Word2Vec" - see Colab notebook separately provided). The marks are broken down among the several parts of the assignment as follows (please refer to the lab script):

1. Preprocessing the training corpus [2 marks].
   Your report must include in your report the output of the 'Sanity check' at the end of this Section

2. Creating the corpus vocabulary and preparing the dataset [2 marks].
   Again, you must include the output of the sanity checks at the end of the Section

3. Building the skip-gram neural network architecture [5 marks].
   To get the credit, you must include the output of the `model.summary()` *command in the sanity check part*

4. Training the models (and reading McCormick's tutorial) [3 marks].
   One point for each answer to one of the three questions

5. Getting the word embeddings [1 marks].
   To get the credit, you must include in your report the output of the instruction printing the DataFrame (summarized)

6. Exploring and visualizing your word embeddings using t-SNE [2 marks].
   Include in the report the output of the `plt.annotate` *command*

Part B: Using LSTMs for Text Classification [15 points]

For this part of the assignment, worth 15 marks in total, you will have to carry out the steps specified by the Lab script ("Lab 2: - LSTMs for Text Classification" - separately provided). These marks are broken down among the several parts of the assignment as follows (please refer to the lab script):

1. Section 2, Readying the inputs for the LSTM [1 marks].
   For this part, show the output you obtain from the sanity check.

2. Building the model (section 3 of the script): [4 marks].
   For this part, show the structure of the model you obtain.

3. Section 4, training the model [3 marks].
   For this part, show the plot of training and validation accuracy through the epochs and comment on the optimal stopping point for the model.

4. Evaluating the model on the test data (section 5) [2 marks].
   For this part, show the output of the command printint test_loss and test_accuracy.

5. Section 6, extracting the word embeddings [1 marks].
   For this part, show the output of `model.summary()`

6. Visualizing the reviews [1 mark].
   For this part, you should include in the report the output of the command printing out the idx2word map.

7. Visualizing the word embeddings [1 marks].
   For this part, you should include the word embeddings for 10 of the words.

8. Section 9 [2 marks].
   For this paper, you have to write down your answers to the questions. 2 points each for questions 1 and 2.

Part C: Comparing Classification Models [20 points]

For this part of the assignment, worth 20 points in total, you must carry out the steps specified by the Lab script ("Lab 3: Comparing Classification Models" - separately provided). These marks are broken down among the different parts of the assignment as follows (please refer to the lab script):

1. Build a neural network classifier using one-hot word vectors (Model 1), and train and evaluate it [5 marks]. *In your report, include the output of the* `model.summary()` *command to show your model structure, and plot training and validation loss in a graph.*

2. Modify your model to use a word embedding layer instead of one-hot vectors (Model 2), and to learn the values of these word embedding vectors along with the model [2.5 marks].
   Again, include the output of `model.summary()`*, and plot training and validation loss.*

3. Adapt your model to load and use pre-trained word embeddings instead (Model 3); train and evaluate it and compare the effect of freezing and fine-tuning the embeddings [2.5 marks].
   Again, include the output of `model.summary()`*, and plot training and validation loss.*

4. One way to improve the performance is to add another fully-connected layer to your network. Try this (Model 4) and see if it improves the performance. If not, what can you do to improve it? [5 marks]
   Plot the training and validation loss of the new model, and other models you try. In your report, describe the differences you see and discuss why they occur.

5. Build a CNN classifier (Model 5), and train and evaluate it. Then try adding extra convolutional layers, and conduct training and evaluation. [5 marks].
   Again, include the output of `model.summary()`*, plot training and validation loss, describe the differences you see and discuss why they occur.*

Part D: Neural Machine Translation [30 marks]

For this part of the assignment, worth 30 marks in total, you will have to carry out the steps specified by the PDF document "Labs 4 and 5 - Neural Machine Translation" that can be found in the Lab 4 and 5 folder on QM+. As for the previous parts of the assignment, you will not need to write all the code, but to fill the gaps left in the script called `nmt_model_keras.py` that you will find in the `nmt_lab_files` folder. These marks are broken down among the several parts of the assignment as follows (please refer to the lab script):

1. Task 1: Implementing the encoder [5 marks].
   Your report must include the code and an explanation.

2. Task 2: Implementing the decoder and the inference loop [12.5 marks].
   Again, you must include the code, an explanation, the BLEU score, and a sample of the output

3. Adding attention [12.5 marks]. *Again, you must include the code, an explanation, the BLEU score, and a sample of the output*

Part E: Using Pre-trained BERT [20 marks]

For this part of the assignment, you must carry out the steps specified by the Lab script ("Lab 6 - Aspect-Based Sentiment Analysis with BERT" - separately provided). These marks are broken down among the several parts of the assignment as follows (please refer to the lab script):

1. Task 1: Data preprocessing [4 marks].
   Your report must include the code for dataset setup and conversion to index and mask sequences, with correct use of separator tokens between text and aspect.

2. Task 2: Basic classifiers using BERT: Model 1 and Model 2 [6 marks].
   The code for these models is provided: run it, and compare the performance you get against the models you built for this task in Lab 4. Explain the differences you see.

3. Task 3: Advanced classifier using BERT: Model 3 [10 marks]. *You must construct, train and test a CNN or LSTM model using BERT embeddings in a similar way to Model 2. Compare the performance you get against Task 2 above. Briefly discuss and explain the differences you see. You must include the code and your explanation.*

Submission

Please submit one zip file with all your answers to Parts A-E together.

As well as code, you should include text explanations, descriptions and answers to specific questions as necessary and as specified above. Code should be in Python; explanatory text can be either as a separate report in PDF format (not Word, please), or included together with the code as a Jupyter/Colab notebook.

For each section, marks will be awarded for correctness of code and classifier performance, but also for clarity of explanations and justifications.