COMP-5413 - Topics in Natural Language Processing
Assignment 2
Due by 5:00 pm Friday, March 20

March 7, 2020

1 Task: Multi-class Sentiment Analysis using Deep Learning

Implement a scalable and robust Convolutional Neural Network-based solution for the problem of text-based movie review multi-class sentiment analysis.

i. Use the raw **train data of Rotten Tomatoes movie reviews** available at `https://raw.githubusercontent.com/cacoderquan/Sentiment-Analysis-on-the-Rotten-Tomatoes-movie-review-dataset/master/train.tsv` into a Colab notebook.

2 Constraints

- **Model:** The model cannot take advantage of advanced sequence handling components, viz. GRU, RNN, LSTM, and so forth. This assignment stays within the scope of Convolutional (Conv) and dense layers and the associated operations, like pooling (Ave, Max, etc.) and non-linear activation functions (ReLU, Sigmoid, tanh, Softmax, Leaky ReLU, etc.).
- **Training:** You are free to train your model for any **n** number of epochs. However, you must archived the training results you obtained in a period epochs, lets say for every 100 epochs. It is to verify the results one-to-one.
- **Features:** This assignment lays on one or combination of BoW, TF-IDF, and Word2Vec only.
- **Data set:** Split the loaded *train.tsv* Rotten Tomatoes Movie Reviews into this assignments' train and test sets with a ratio of $70:30$ using *sklearn.model_selection* library with random state is set to 2003.
- **Performance evaluation:** It is based on the following metrics - Accuracy, Recall, Precision, and Figure-of-Merit (f-1) score.

3 Deliverable

i. A 3 to 5-page limit formal Scientific/Engineering report in IEEE format that clearly elaborates all the steps you carried out in completing the task.

A formal report by default covers: topic title, abstract, introduction, background/literature review, proposed model, experimental analysis/comparisons with other methods/approaches/applications if applicable, conclusion and references.

The write-up should take advantage of figures, plots, charts, flow charts, diagrams, tables, graphs, or such tools to clearly communicate the findings to any reader. The complexity of the writing must trade off between storytelling and presenting scientific/engineering concepts.

ii. All source code/script files in a separate directory.

iii. A trained model named in the format of *student id_1dconv_reg*.

iv. A Github link that contains all the details of your model.

4 Submission

All soft-copies must be submitted to the assignment folder on the D2L before the deadline, while a hard-copy of the report must be handed in to the instructor.

5 Evaluation

i. It will be evaluated based on the merit of the solution and quality of the report.

ii. The top-3 models according to the test results will be awarded with a certificate, and 1 bonus mark.

6 Hints

i. Way of including source code in the report:

Let's consider there is a function written that acts differently based on an input string. Then, one of the excellent ways for adding the code of the function as follows.

Firstly, explain the function or the script in the main context of the report (again you can use flow charts or pseudo code), then include the code/script in actual code format (not in regular font) in Appendix as an example given below.

Example 1: The function *myCallService(url: String, urlFreeVer: String)* accepts two arguments: the URL of build variant from *BuildConfig* object and a preset URL for free version. It launches a second activity if the URL belongs to the Premium build. Otherwise, it produces a warning message to the user. The actual code snippet is given in Appendix 7.1.

ii. Include data visualizations, model architectures (not just benefiting from copy-paste approach. Avoid screenshot, unless it is a high quality image), performance analysis through plots and charts.

iii. You are free to exploit any number of preprocessing stages, such as tokenization, removal of stop-words and punctuation, normalization (stemming or lemmatization), extended stop-word removal, one-hot label encoding, label normalization, etc.

iv. You are encouraged to get the plots of training time performances with respect to epochs.

v. There is a **zero tolerance for plagiarism**.

7 Appendix

7.1 myCallService

```
fun callService(url: String, urlFreeVer: String) {
        // calling code here
        val url = "http://www.myver.com"
        if (url.equals(urlFreeVer)) {
            Toast.makeText(this, "Free ver", Toast.LENGTH_LONG).show()
        }
        else{
            // Setting up an intent to open new activity
            val myIntent = Intent(this, nextActivity::class.java)

            // starting the 2nd activity
            startActivity(myIntent)
        }
}
```

Listing 1: myCallService Custom Function.