Manual

There are 2 modes plus one bonus feature after the benchmarking mode.
Across the project there are numerous “Todo”s that direct you to the code that can be changed to use different parameters mentioned in the report, I have labelled the Todos with a number after or “all” if all are required for each function

Training an average (Todo 1):

At the bottom of the CNN_Classifier.py file, there is a function call to benchmark_models() which runs multiple training sessions for a given model.

Hyperparameters

At the top of the file (from line 32) there is are hyperparameters that you can choose from, the ones you will need to change to replicate benchmarks are:

- LR
- DROPOUT
- MAX_DOC_LEN
- FILTER_SIZES
- N_FILTERS
- NUM_EPOCHS
- THRESHOLD

You can also change the datasets and number of models to train via:

- project = ‘tensorflow’
- num_iters = 10

To change the average used in the precision, recall and f1, go to line 187 to change this between binary and macro

Preprocessing

The preprocessing steps start around line 285 of CNN_Classifier.py, for which you can change various techniques used:

- Clean string - enabled by default - comment out line with `pp.clean_str` to remove
- Stopword removal - uncomment 2 lines downloading stopwords and `apply(pp.remove_stopwords)` to enable
- Stemming and lemmatization - where the function `pp.tokenize_words()` is called, change the `preprocess=”stem”` or `“lemmatize”` to enable either, or.
- Lowercase - in preprocessing.py on line 54, add .lower() to the end

You can use these changes to replicate the benchmarks taken in the report

User Input Prediction (Todo 2)

You can use the function `user_input_prediction()` alongside `benchmark_models()` to enter your own input text for the model (last model benchmarked) to predict the class of the input.

Uncomment the last line in the file to enable the feature, enter “quit” to stop running the model

Hyperparameter tuning (Todo 3)

To run the hyperparameter tuning, comment out the benchmark model and user_input_prediction lines and uncomment `tune_hyperparameters()` near the bottom, you can change the parameter search in the `define_search_space()` function, you can also set them to a single value if you dont want it to search a particular parameter (eg `trial.suggest_categorical(“hidden_size”, [[]])` this will set the hidden size to `[]` statically.

You can also change the number of samples at line 428
Make sure to change the storage_path to a reasonable location (must use absolute path), it is currently not setup.

Run the CNN_Classifier.py file and it should start training multiple models (note this will be very slow if you are not using cuda)

Baseline:

The baseline is the one used in lab 1 (br_classification-baseline.py), the only changes required are the project, and the precision, recall, and f1_score averages between binary and macro

Calculating Wiscoxon Rank-sum test

When you run either of the models benchmarks, they will print either `”proposed_results_array”` or `”baseline_results_array”`
Simply copy the entire line and paste it into the `calculate_rank_sums.py` file replacing the current variable data. Do this for both models benchmarks. Then change the project to whichever the models were tested on.
Running the file will print the results and add it to the related csv.