Project1: Detecting Dementia using Natural Language Processing:

Description: This project aims to develop a predictive model using both machine learning models and Generative AI (GAI) techniques. The model effectively utilizes linguistic features extracted from text data to predict dementia diagnosis. one reference has been given as a sample. The database used for this project is available at DementiaBank (talkbank.org)Links to an external site. We have imported all the transcript files into CSV format for training (Dementia_db.csv, Control_db.csv) and testing datasets.
Link to the project page on Canvas:
https://stavanger.instructure.com/courses/15609/pages/project
Tips:

1. The main evaluation of your work is based on the quality and how many of the points mentioned in the project guideline page on Canvas (https://stavanger.instructure.com/courses/15609/pages/project) are addressed.
2. Do a literature review; look at some related articles with a similar focus to find recent trends and gaps or challenges in the field.
3. First, you need to read the chat files and extract relevant information such as patient ID, MMSE score, labels, transcripts, and so on.
4. Do preprocessing steps, such as removing unnecessary notation in texts and punctuation and removing outliers.
5. Try different word representation techniques, including TF-IDF, fastText, Bert, or traditional linguistic features.
6. Optional step: feature selection
7. Try with different classifiers: SVM, DT, ...
8. Keep the test set untouched.
9. Divide the train set into new train and development sets and train your model on the new train set, and optimize it (using hyperparameter tunning and a cross-validation approach.
10. At least train 4 classifiers and report the result on the untouched test set. The performance metrics should cover accuracy, precision, recall, F1-score, confusion matrix heatmap, and AUC and roc curve.
11. It is really important that you interpret the result.
12. For writing the report, you should use the overleaf file requested in the guideline (https://www.overleaf.com/latex/templates/association-for-computing-machinery-acm-large-2-column-format-template/qwcgpbmkkvpq).
    Link to DementiaBank: https://dementia.talkbank.org/
    Link to dataset paper: https://dementia.talkbank.org/ADReSS-2020/

Project 2: Bag of Words Document Classification using Feedforward Neural Network and Recurrent Neural Network (RNN)

Goal: To familiarize yourself with training neural networks on textual data.
Task: You will have to address a classification task with different architectures (of different complexities).
Steps:

- Step 1: you will work with bag-of-words as features(e.g., TF-IDF, CountVectorizer).
- Step 2: you will use static pre-trained word embeddings (word2vec, fasttext, GloVe)
- Step 3: you will employ RNN for word embedding.

Dataset: You will be working with a portion of the Arxiv-10 (Glenn and Goldman 1976) dataset, which consists of 80,000 English-language scientific articles. Each article includes an abstract and its corresponding research field, categorized into 10 subfields within computer science, physics, and mathematics. Your objective is to predict the research field based on the abstract, making this a multiclass classification task. It is provided as a gzipped comma-separated text file, with one data instance corresponding to a paper. It has two columns:
• abstract (input)
• label (output)
Tips:

1. Before training, you have to split it into training and development parts. Make sure to keep the distribution of labels as similar as possible in both subsets.
2. Implement a different number of layer fully-connected feed-forward neural networks (multi-layer perceptron), which train on bag-of-words features and evaluate it on the development dataset.
3. Load pre-trained word embeddings for the input text and merge the sequences of word embeddings into a fixed-size vector. Apply a classifier over this fixed-size vector.
4. Optional: you are welcome to pre-train your own embeddings on a corpus of your choice for word2vec, fasttext, or GloVe . You can use any English corpora available on the net (not less than 10 million word tokens).

5. Try Different ways of merging the sequence of embeddings into a fixed-size vector representation: averaging them, summing them, etc.;
6. Optional: Different ways of dealing with out-of-vocabulary words;
7. Different pre-trained embedding models (built using different algorithms: word2vec, fasttext, GloVe);
8. Existing pre-trained models vs. word embeddings that you pre-trained yourself;
9. Frozen or fine-tuned word embeddings. Please document your choices and experiments in the report.
10. You can try e different kinds of RNN: Simple (Elman's) RNNs, LSTMs, and GRUs.
11. For an RNN network to put the input into the fixed length, you can use the following approach: Using the last state of the RNN, Using the first state of the RNN (if your network is bidirectional), Using both, concatenated/added/max-pooled, Adding/max-pooling every stat.

Regarding the report:

- At least 4 pages (max 10 pages) in 2-column ACM conference style in latex (Here is the link to the template on O)
- Writing should be high quality technical writing not copy pasting think of this as a writing practice for your thesis
- Describe the task, dataset, methods you use and discuss their limitations you saw in the experiments
- Don't forget to include the github link to your code in the report along with a README.md in github repository which mentions how to reproduce your results in the report.
- Very important: Document who did what and justify you did equal share of the tasks