Kaiying Lin
ICS637-Final Project

Detecting Dementia

1. Introduce the problem. What problem are you trying to solve? What are the inputs and outputs of your model?

This project aims to detect dementia using the recordings of conversations between patients/non-patients and consultants. The training data will be the audio recordings and the output will be the binary prediction of patients/non-patients.

2. Introduce the dataset. Machine learning is about learning from data, and the model will only be as good as the dataset. Clearly explain where your data came from.

The datasets, or the audio recordings are from the Dementia Databank. This databank is reliable because many previous studies have performed similar tasks on some of the datasets in this Databank. This dataset is only used as training set and validation set. The model is evaluated on a test set in a task called ADRess challenge task. This challenge is a shared task which aims for detecting one type of dementia, Alzheimer's disease. The reason of using this as a test set is because there are control (non-patient) datapoints and we don't need to create control datapoints by ourselves, which will be the case if we use other folders in Dementia databank.

3. Introduce the model. Which machine learning model(s) are you using? Is there any particular reason?

I decided to fine-tune an existing audio pre-trained model, wav2vec. This model has shown promising results in some literature, especially when the training data additionally contains the text transcription. Wav2vec converts the audio waves into vector representations, and these vector representations can be further used for speech recognition, audio classification.

Additionally, I also fine-tuned another model called Yamnet. Previous literature also fine-tuned the same model to achieve decent accuracy on Dementia datasets, so I chose this model.

4. Specify features and pre-processing. Describe the feature representation and any transformations applied. This may include:

• Feature units, e.g. inches, meters, or unit-less.
• Feature properties, e.g. categorical, ordinal, integer, real-valued, one-hot.

• Transformations, e.g. min-max scaling, standardization, log(x) or log(x + 1).
• Missing data methods, e.g. removal, padding, interpolation, imputation, iterative imputation.
• Augmentation, e.g. gaussian noise.

The training data are autoloaded with the built-in function 'load_dataset'. This function transformed the audio and waveforms are decoded as 1-dimensional arrays. The Wav2vec and Yamnet require the dataset to be in a sampling rate of 16000kHz, so the training data was resampled to 16000kHz.

5. Specify data splits and how they are used.
Example: The total dataset contained 1,000 examples. The data was randomly permuted, then divided into three subsets: 80% training, 20% validation, and 20% test. Models were trained on the training set, while the validation set was used for early stopping, hyperparameter optimization, and model selection. The test set was used to evaluate the final model.

I used the English/Pitt folder in the Dementia Databank as the dataset. It contains ~1200 audio files and I split the dataset into 80% training set and 20% validation set. The validation set is used to evaluate the model after each epoch of training, and also used for hyperparameter tuning. The test set is just the test set in ADRess 2021 Challenge task.

6. Specify the hyperparameter search space. This is the list of hyperparameters that were optimized, and the range of values that were explored. This can be difficult to describe succinctly since hyperparameter tuning is usually an iterative process involving the experimenter. Ideally, you use a hyperparameter optimization framework like Sherpa [1], but in the case where hyperparameter optimization was mostly done by hand, you can simply state the range of values you explored (min and max) for each hyperparameter and the total number of models you tried.
Example: For the K-Nearest Neighbor classifier we tried different values of K and different distance metrics. We tried all odd values of K from the set of integers between 1 and 99, {1,3,5,...,99}. We tried the L1 and L2 distance metrics.

Most of the hyperparameters were the default of training_args as in https://huggingface.co/docs/transformers/tasks/audio_classification. Only the learning rate was tuned after I trained the first model with learning rate 1e-8. The learning rates I tried were 5e-8, 1e-8, 5e-7, 1e-7. I found by chance the model requires two different learning rates and I needed to stop the model and reenter the new learning rate. Therefore, the first learning rate is 1e-7 as the optimal and the second learning rate one is 1e-5. With these learning rates, the Wav2vec model achieved ~80% training auroc scores, and ~70% validation auroc scores.

No hyperparameters were tuned for the Yamnet model. The model achieved 70-80% training accuracy and 60% validation accuracy.

7. Explain how hyperparameters were optimized. Explain if you tuned the hyperparameters by hand, or exhaustively tested every hyperparameter combination in the search space. This can be a simple statement of the metric and validation set used. State any optimization algorithms you used, e.g.
1
Random Search, Grid Search, Bayesian Optimization, Population-Based Training. Cite any software packages you used to implement these algorithms (e.g. optuna or sherpa, [1]).
Examples:
(a) After trying all combinations of hyperparameters in the search space, the model with the highest
accuracy on the validation set was selected.
(b) A total of 50 different models were trained, with random combinations of hyperparameters selected from the search space. The model with the highest validation set MSE was selected and evaluated on the test set.
(c) Hand-tuning was used to train a total of 20 different models with different hyperparameters. The model with the best validation set AUROC was selected and evaluated on the test set.

Since there is only one hyperparameter, learning rate, to be tuned, I used a grid search and conducted the first training with different learning rates 5e-8, 1e-8, 5e-7, 1e-7. 1e-7 was the optimal one. The second training only utilized 1e-5 so no further hyperparameter tuning was conducted.

8. Evaluate model on clean test set. When quantifying performance, remember to specify the metric and dataset for every number you present.
Example:
(a) The performance of the final model on the test set was .98 AUROC. (b) The model achieved an accuracy of 80% on the held-out test set.

Wav2vec model: the performance on a clean test set (ADRess Challenge task) achieved 68% auroc score. The model was not trained specifically on data of Alzheimer's patients, and the validation auroc score also achieved only 70%~, so this evaluation results seem reasonable.

Yamnet: the performance on the test achieved 50% auroc score. More adjustments on the model should be considered.

9. Explain any differences in the train/test datasets. If possible, provide justification for why you expect the model to generalize despite differences.
Example:

(a) The test set is from a later time than the training set, so data drift could harm model performance.

As mentioned above, the test data only includes one particular type of dementia disease, Alzheimer's disease. The training data, the Pitt folder, contain data of patients with multiple types of dementia diseases, as well as Alzheimer's disease. The model was trained on the training data so it should be able to generalize onto the test data, as the training data also included data of patients with Alzheimer's disease. However, since the training data does not specifically include data of patients with Alzheimer's disease, so the worse evaluation auroc score seems reasonable for me. Also test scores are often lower than the training scores so this result is expected.