ARI5121 – Applied NLP – Assignment
Keyword Speech Recognition
Deadline: 30th June 2020

Description

In this assignment, the task is to build a single-word speech recognition system. The aim is to be able to recognize a set of predefined words (keywords), as accurately as possible from short audio files. Of course, real speech and audio recognition systems are much more complex than this problem, but going through this assignment should give you a basic understanding of the techniques involved.

Dataset

The dataset you will use is the **Google Speech Commands Dataset** (https://storage.cloud.google.com/download.tensorflow.org/data/speech_commands_v0.02.tar.gz). This dataset consists of about 65,000 one-second long utterances of 30 short words e.g. "yes", "no", "up", "down", "left", "right", "on", "off", "stop", or "go". For the purposes of this assignment, you will select any 10 words of your choice.

Training/Testing Split

In order to run proper evaluations, you will need to split the dataset of the 10 selected words into appropriate training/validation/test sets. Around 70% of the speakers for each word chosen should be used for training. A further 10% of the data should be used for validation (especially required for the second method below), leaving 20% of the speakers for testing. Validation and testing should therefore be *speaker-independent*. It is best to make this split explicitly, as the sklearn method of performing train/test splits will not have any notion of speakers.

Implementation

Based on the material discussed in the previous lectures/tutorial, you will need to provide two separate classifier implementations:

Method 1: MFCC+GMM Models [40%]

In the first method, the task will be to perform feature extraction from raw speech in the form of MFCC feature vectors for the utterances. You may use an efficient library such as "Python Speech Features" (https://python-speech-features.readthedocs.io/en/latest/#) for this purpose. You need to build an appropriate GMM model for each word/class, and then perform GMM-Bayes style classification on a test set (a general GMM-Bayes implementation was provided for you in your notes for Week 2). You may need to tune the GMM size based on the F-Score/Accuracy you observe etc.

Method 2: CNN Models [40%]

In the second method, the task will be to perform spectrogram extraction from the raw speech – which results in an image matrix. The size of the images should be equivalent as all recordings are a second long. You will need to build an image-based CNN architecture (it is highly suggested to use Keras for this) to build a classifier for these 10 word classes. We are therefore treating the problem as an image classification problem. The spectrograms you extract will all be equivalent in time. You should not need to use very high resolution FFT bands when getting the spectrograms. Something like 64 channels should suffice. Being a popular dataset, you are bound to find help from various sources to solve this task. For instance, some notes on a MATLAB implementation of a very similar approach can be found here (https://www.mathworks.com/help/deeplearning/ug/deep-learning-speech-recognition.html;jsessionid=dd2efd94015d5eeb826eb50c0bc1), including the CNN architecture. It will be up to you to try out different architectures and tweaks to get the best F-Score accuracy.

Submission and Report [20%]

You are to submit your code files as a ZIP file. You should also submit a link to a shared Google Drive folder with your MFCC/Spectrogram images, trained models etc. Along with the deliverable, you will need to submit a short report of not more than 4 pages in a paper-style format (see Interspeech 2020 template). This paper should describe your methodology, the experimentation that led you to the particular choices of hyperparameters/architectures, results and analysis of those results as well as a discussion comparing the two methods.