Capstone Proposal for Machine Learning Nanodegree

By Muhammad K. Chughtai

Domain Background

Speech recognition is an area that has been gaining traction for some time with evolving AI techniques. Many smart devices such as Alexa. Siri, etc. have successfully used speech recognition to aid in simple day to day activities. Speech recognition has been done mainly for the English language but there is limited information available for the Arabic language. Arabic is not only the official language of many countries around the world, but it is also the language of the Quran, the holy book for the religion of Islam.

The proposal here is to apply ML techniques on Quranic recitations to identify the person reciting (or speaking) the Quran. This study was inspired by the paper titled "Quran Reciter Identification: A Deep Learning Approach" (https://ieeexplore.ieee.org/document/8539336). This paper is using a Neural Network for Quranic speaker identification and was able to correctly identify the reciters.

My personal motivation to solve this problem is to show that ML techniques can not only work for the Arabic speaker identification, but these techniques can be further enhanced to solve other problems in the Arabic language. The reason to use Quranic recitation data for this project is because Quranic recordings are readily available on the internet, whereas Arabic speech samples are not as readily available.

Problem Statement

The problem being solved here is to show how well the ML techniques for speech recognition work for the Arabic language. Neural Network is the method of choice in the above-mentioned paper, and that is what will also be used here. However my Neural Network will be a different version as I don't have access to the exact code & data used in the paper. Since sound is a time series data, it makes sense to use Neural Network for training. Although **time permitting**, I am curious to see if any of the simpler models will also work. I'd also like to mention that the MP3 files for one of the reciters listed in the paper is missing on the website, so I will need to choose a different reciter which may vary my results.

Once the model is trained then the trained model will be used to predict the speaker. Other metrics (recall/accuracy/etc.) will also be looked at to see how well the model is working. I will consider the problem solved if the reciter is identified with a reasonably high accuracy, however I may not be able to get close to the accuracy described in the above-mentioned paper.

Datasets and Inputs

The dataset for this can be found here: https://everyayah.com/data/status.php

The website has a list of reciters (or speakers) and for each reciter there are labelled MP3 files available for download. I am planning to choose at least five reciters (the website has a lot more).

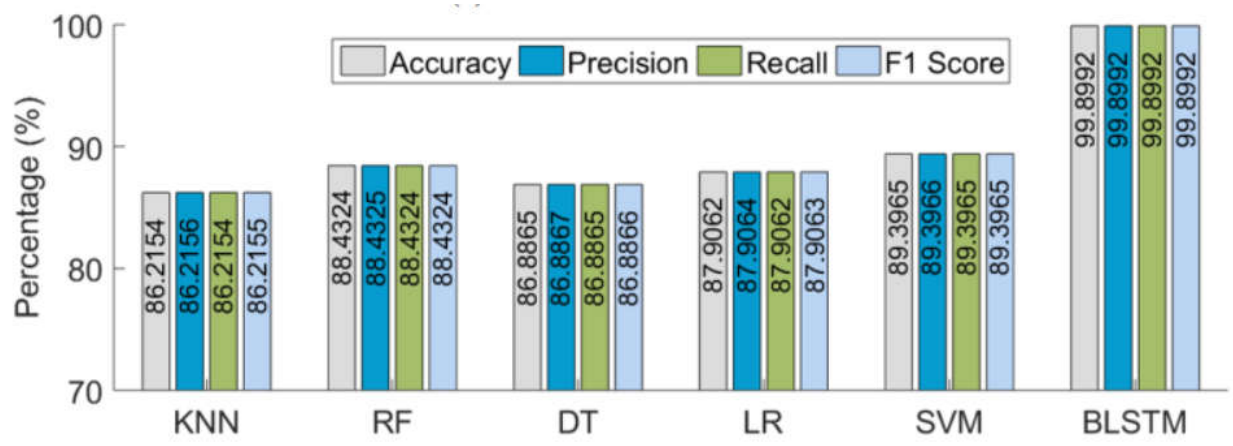
There are 6,236 verses in the Quran so each reciter has more or less as many MP3 files. It may be impractical to train on all the 6,236 verses so I am planning to choose a subset for training & then splitting them up for training/testing. The available MP3 files are sampled at various kbps (e.g. 32, 64, 128, 192, etc.) For the most part I am sticking to 64kbps since the voice quality will be decent and the download size per reciter is in the ~500-800MB range. My research shows that the difference in kbps should not be a big problem.

Solution Statement

The audio features will be extracted from the MP3 files. Then a Neural Network will be used to train the model and make a prediction to identify the speaker. I will most likely use a LSTM NN for this project, and maybe make it Bidirectional [2] which is what is used in the above-mentioned paper [1].

Benchmark Model

One benchmark will be for the trained model to identify the speaker correctly majority of the time. A comparison will be made to some of the benchmarks (recall/accuracy/etc.) discussed in [1]. The paper authors for [1] used a BiLSTM NN to generate their results and compared their results to previously published results in other papers. Here is an example from the paper:



Evaluation Metrics

Recall/Accuracy/Precision data will be collected from the trained model along with a confusion matrix to show how well the model is performing.

Project Design Outline

The project will be developed by going through the following steps in sequence:

- Analyze input data: The project will start by analyzing the input data which is in MP3 format that represents sound. Basic analysis of the sound data will be done to understand the fundamentals of how sound is represented in MP3 format, and associated attributes such as sampling, length of the sound clip/etc. will be looked at. This data will be reviewed for all reciters across all MP3 files to understand the input data. A brief summary and stats of input data will be published to understand how many verses are used and what the attributes of the MP3 files are.
- Pre-processing: The next step will be to understand if any of the input data needs to be "pre-processed" e.g. if it's necessary to make the length of the audio the same for each verse before the next step, etc.
- Extracting the "right" audio features: This step will extract features from the MP3 files to represent the data in a format that can be used for ML. The above-mentioned paper talks about "MFCC" features, but there are other features that can also be extracted.
- Normalization and train/test split: The data will be normalized and split into training/testing
- Training: A Neural Network will be used for training. Since sound is a time series based data, it makes sense to use Neural Network. Although **time permitting**, I am also curious to see if any of the simpler models will work.
- Predictions: The model will be used for a simple prediction.
- Metrics and Confusion matrix: Relevant metrics such as accuracy/recall/etc. and confusion matrix will be collected. These will be compared to some benchmarks discussed in the above-mentioned paper.

References

[1] "Quran Reciter Identification: A Deep Learning Approach",
https://ieeexplore.ieee.org/document/8539336

[2] "How to Develop a Bidirectional LSTM For Sequence Classification in Python with Keras",
https://machinelearningmastery.com/develop-bidirectional-lstm-sequence-classification-python-keras/