Guna Sekhar Sai Harsha Lagudu (ASU ID :1223118858),
Abrar Ilyasbhai Paliwala (ASU ID : 1223111292)

Introduction:

In this project, we have developed a TinyML model for speech recognition and deployed it on Arduino nano 33 BLE sense . The developed model can detect 5 words (i.e, all, none, never,must,only). We have used an on board microphone to receive the input voice and the output can be seen on the serial monitor. The output shows the probabilities of each word and the word with highest probability is the predicted output. The developed model has 96.43% test accuracy.

Experiment:

In this project, we have recorded the data using a website Open Speech Recording(Open Speech Recording (harvard.edu)). We have recorded the data by using 5 different voices and each class has 60 audio clips in '.ogg' format and hence, our dataset consists of 300 audio clips each of length 1 second. The collected data is then converted to '.wav' format by using CustomDataset_KWS_mHealth.ipynb - Colaboratory (google.com).

Algorithm:

The workflow of our work is shown in Figure 1. Initially, we collected the data using Open speech Recording. Then we used Edge Impulse to preprocess the data, train and deploy the model.

Figure 1: The workflow of the project

1. **Preprocessing the data:** For preprocessing the data, we used Mel-frequency cepstral coefficients (MFCC). In sound processing, the **mel-frequency cepstrum** (**MFC**) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency**.**
2. **Model:** The Neural network architecture summary is shown in Figure 2. The input layer has 650 features and it is followed by a reshaping layer, which maps the input features into 13 columns. Then a series of 1-D convolution layers with RELU activation and dropout

layer is applied followed by a flatten layer and then output layer. The output layer is a fully connected layer with 5 neurons and softmax activation layer.

3. **Training:** For training we have used 80% dataset and 20 % as validation set. We have trained our model with 100 epochs and a learning rate of 0.005.
4. **Deployment:** For deployment, we have created an arduino library using Edge Impulse. Initially, our trained model is quantized and the quantized model uses 5.0KB of RAM & 35.1KB of flash memory while the unquantized model uses 8.4KB of RAM & 39.3KB of flash memory. Using this library, we have created a model for speech detection on arduino.

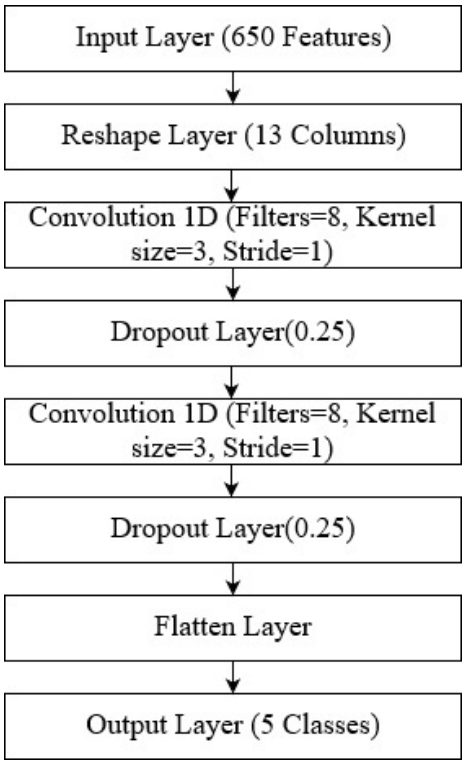

Figure 2 : The Neural Network Architecture

Results:

After training for 100 epochs, we have training accuracy of 98.46% and loss of 3.74%, the validation accuracy is 99.8% and loss is 3.09% and the testing accuracy is 96.43%. The confusion matrix for training and test dataset is shown in Figure 3 and 4 respectively. The visualization of model predicted data is shown in Figure 5, here, the plots in red were incorrect predictions. Later, we did real time prediction by speaking the words near to the arduino microphone and observed the prediction by the embedded system. For this, we have used different voices from 2 different people other than voices from people used for training sets. We asked them to utter each word 2 times in different tones, hence a total of 20 observations are used for real time prediction. Out of

20 observations, the embedded system is able to predict 17 observations correctly i.e an accuracy of 85%.

	ALL	MUST	NEVER	NONE	ONLY
ALL	100%	0%	0%	0%	0%
MUST	0%	100%	0%	0%	0%
NEVER	0%	0%	100%	0%	0%
NONE	0%	0%	0%	100%	0%
ONLY	0%	0%	0%	10%	90%
F1 SCORE	1.00	1.00	1.00	0.95	0.95

Figure 3: The confusion matrix for training dataset

	ALL	MUST	NEVER	NONE	ONLY	UNCERTAIN
ALL	100%	0%	0%	0%	0%	0%
MUST	0%	100%	0%	0%	0%	0%
NEVER	0%	0%	91.7%	8.3%	0%	0%
NONE	0%	8.3%	0%	91.7%	0%	0%
ONLY	0%	0%	0%	0%	100%	0%
F1 SCORE	1.00	0.97	0.96	0.92	1.00	

Figure 4: The confusion matrix for testing dataset

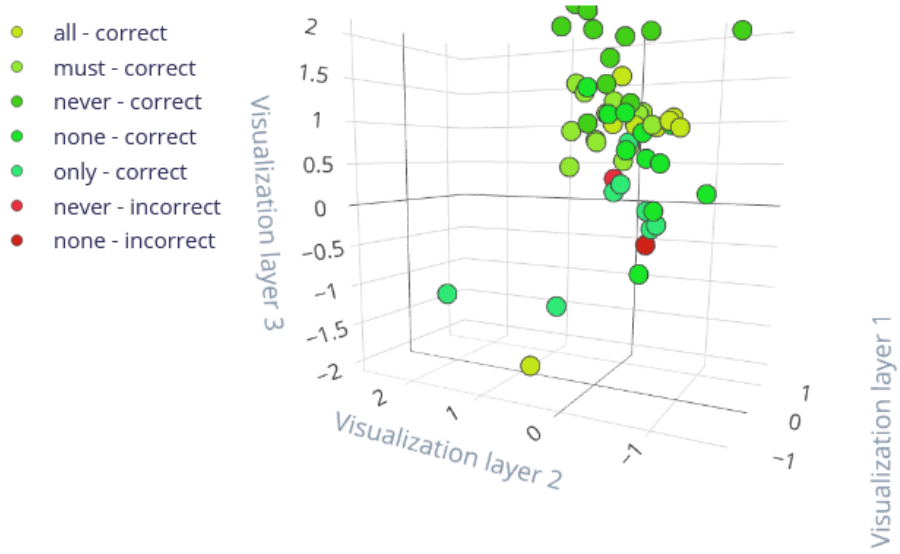

Figure 5: The visualization of model predicted data on Testing set

Discussion:

We successfully implemented a model to classify speech commands. The model has test accuracy of 96.43% while the real time accuracy is 85%. The accuracy of real time prediction can be increased by training the model with more data. One of the difficulties we have faced during testing the real time prediction is if the microphone is far from the source it might make a wrong prediction due to the background noise as it dominates our sound. The future improvements are a) the model can become more robust by adding the background noise to the data, b) the audio clips should be collected from different people with different accents and different tones.