MIDAS - IIITD (Internship Task, 2021)
Task - 1 (Speech Processing) - Detailed Explanation

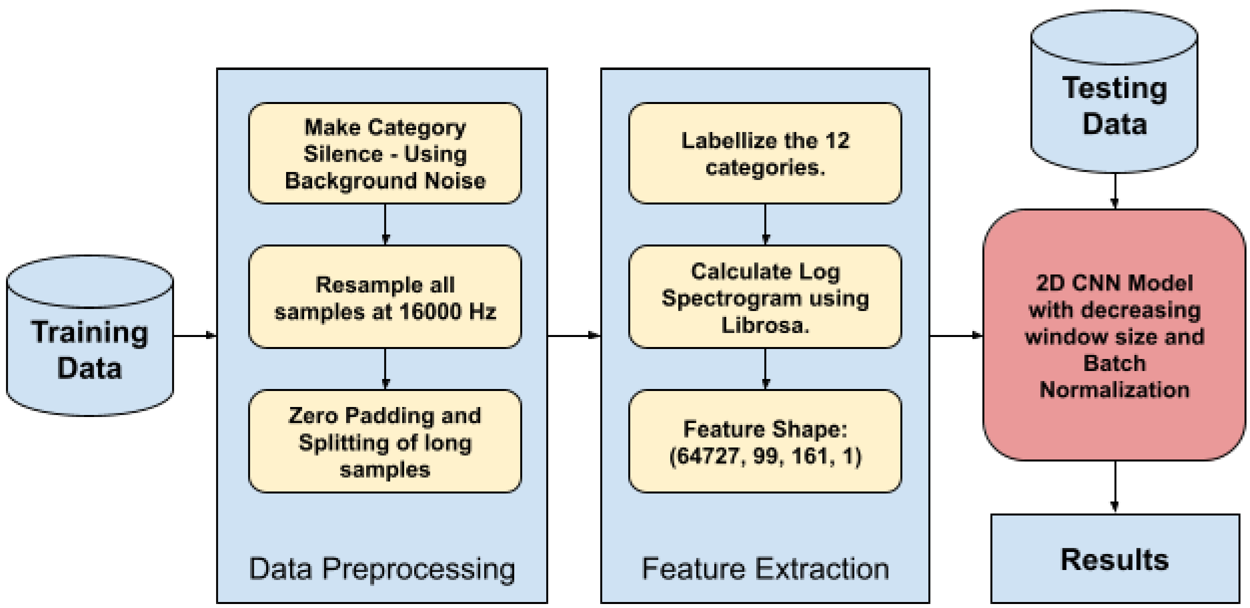
1. Task Introduction

In this competition, you're challenged to use the Speech Commands Dataset to build an algorithm that understands simple spoken commands. Dataset details are present at Kaggle.

2. Introduction to Proposed Solution

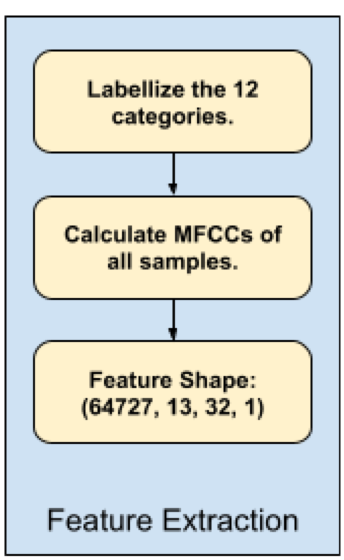
The following two flow charts represent the **three proposed and implemented methodologies** for the task.

2a) Using Log Scaled Normalized Spetrograms:

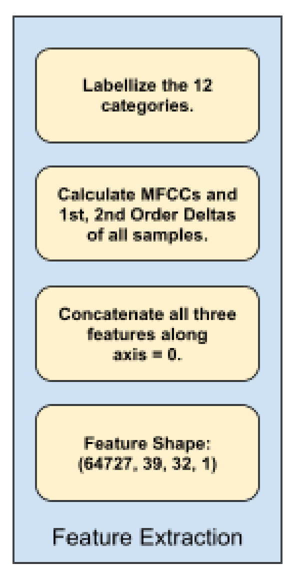


The other two methods **differ only in the feature extraction methodologies** shown below:

2b) Using MFCCs (First 13):



2c) Using MFCCs and 1st, 2nd Order Deltas (First 13):

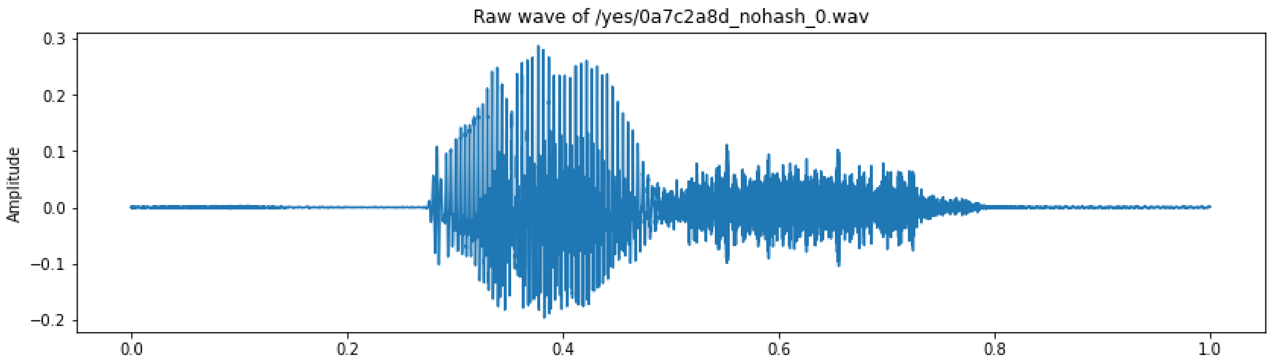


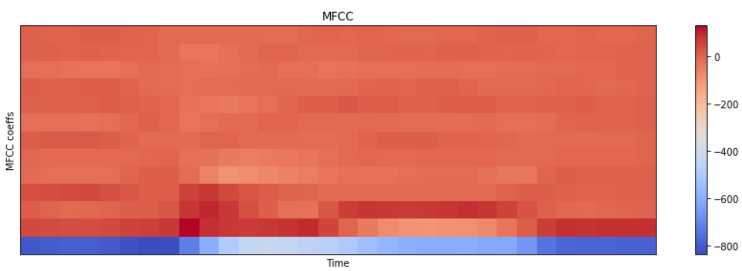
3. Exploratory Analysis of Dataset

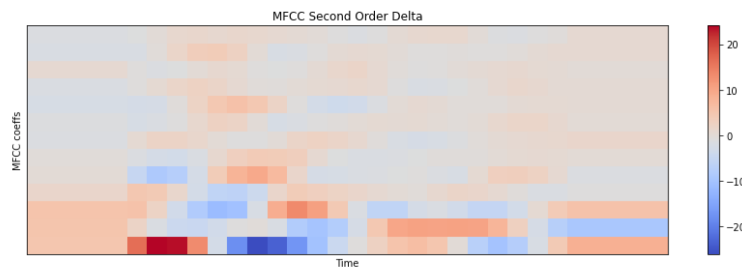
To correctly describe and represent an audio sample, the following features were identified:

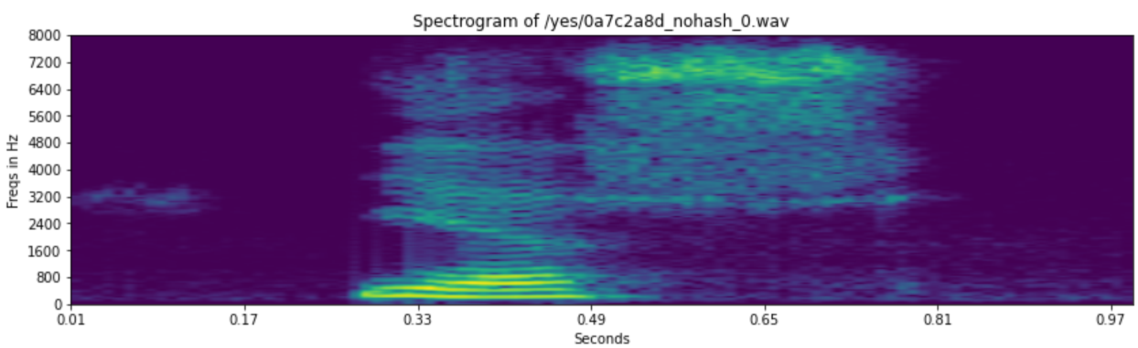
- Normalized Log Scaled Spectrogram of Audio Samples [1]
- Mel Scaled Cepstral Coefficients and their higher order deltas.
- Fourier Transform of Samples. [1]

The same are shown below for a random sample of audio, whose raw plot, sampled at 16000 Hz is shown below. Other plots are in the Jupyter Notbook which compare the same for all categories.









4. Details of Experiments Performed

4a) Using Normalized Log Spectrograms:

1. First, all samples were zero padded to 1 second samples to make sure that all samples are of equal dimensions. [2]
2. It was noticed that **many samples were greater than 1 second.** To tackle this, an idea was borrowed from [3] to split longer samples into new samples. This increases the training size also.
3. Normalized Log Specs, a computation of the FFT computed on overlapping windowed segments of the signals, are calculated using Librosa for all features.
4. The y-axis is converted to a log scale, and the color dimension is converted to decibels. **This is because humans can only perceive a very small and concentrated range of frequencies and amplitudes.** [4]

4b) Using 13 MFFC's:

1. It was noticed that the spectrograms did not give good results.
2. This could be attributed to the fact that the sample durations are too short for valuable feature extraction over a continuous domain.
3. MFCCs were selected since these are known to be similar to the perceived frequency scale in humans [5].
4. Only the first thirteen are selected, which is the norm in current research [6].
5. These gave much better results and are also **computationally inexpensive.**

4b) Using 13 MFFC's and 1st and 2nd Order Deltas:

1. To further improve the accuracy, the deltas of MFCCs, the first and second order derivatives, were concatenated to the MFCCs.
2. **The benefit of deltas over MFCC features is that they are used to represent the temporal information.** [7]
3. They all are concatenated to increase dimension size.

5. Model Details

- 2 Dimensional CNNs are used.
- Window sizes are varied from high to low while filters are increased.
- Binary Crossentropy Loss is used for training.
- Batch Normalization is added with dropouts to prevent overfitting.
- Models are shown below:

Model: "model"

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 99, 161, 1)]	0
batch_normalization_1 (Batch	(None, 99, 161, 1)	4
conv2d_4 (Conv2D)	(None, 95, 157, 8)	208
max_pooling2d_1 (MaxPooling2	(None, 47, 78, 8)	0
dropout_1 (Dropout)	(None, 47, 78, 8)	0
conv2d_5 (Conv2D)	(None, 45, 76, 16)	1168
conv2d_6 (Conv2D)	(None, 43, 74, 16)	2320
max_pooling2d_2 (MaxPooling2	(None, 21, 37, 16)	0
dropout_2 (Dropout)	(None, 21, 37, 16)	0
conv2d_7 (Conv2D)	(None, 20, 36, 32)	2080
conv2d_8 (Conv2D)	(None, 19, 35, 32)	4128
max_pooling2d_3 (MaxPooling2	(None, 9, 17, 32)	0
dropout_3 (Dropout)	(None, 9, 17, 32)	0
flatten (Flatten)	(None, 4896)	0
dense (Dense)	(None, 32)	156704
batch_normalization_2 (Batch	(None, 32)	128
dense_1 (Dense)	(None, 64)	2112
batch_normalization_3 (Batch	(None, 64)	256
dense_2 (Dense)	(None, 12)	780

Total params: 169,888
Trainable params: 169,694
Non-trainable params: 194

Used for 4a

Model: "model_1"

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	[(None, 13, 32, 1)]	0
batch_normalization_6 (Batch	(None, 13, 32, 1)	4
conv2d_18 (Conv2D)	(None, 9, 28, 8)	208
dropout_6 (Dropout)	(None, 9, 28, 8)	0
conv2d_19 (Conv2D)	(None, 7, 26, 16)	1168
conv2d_20 (Conv2D)	(None, 5, 24, 16)	2320
dropout_7 (Dropout)	(None, 5, 24, 16)	0
conv2d_21 (Conv2D)	(None, 4, 23, 32)	2080
conv2d_22 (Conv2D)	(None, 3, 22, 32)	4128
dropout_8 (Dropout)	(None, 3, 22, 32)	0
flatten_1 (Flatten)	(None, 2112)	0
dense_3 (Dense)	(None, 32)	67616
batch_normalization_7 (Batch	(None, 32)	128
dense_4 (Dense)	(None, 64)	2112
batch_normalization_8 (Batch	(None, 64)	256
dense_5 (Dense)	(None, 12)	780

Total params: 80,800
Trainable params: 80,606
Non-trainable params: 194

Used for 4b and 4c

6. Results

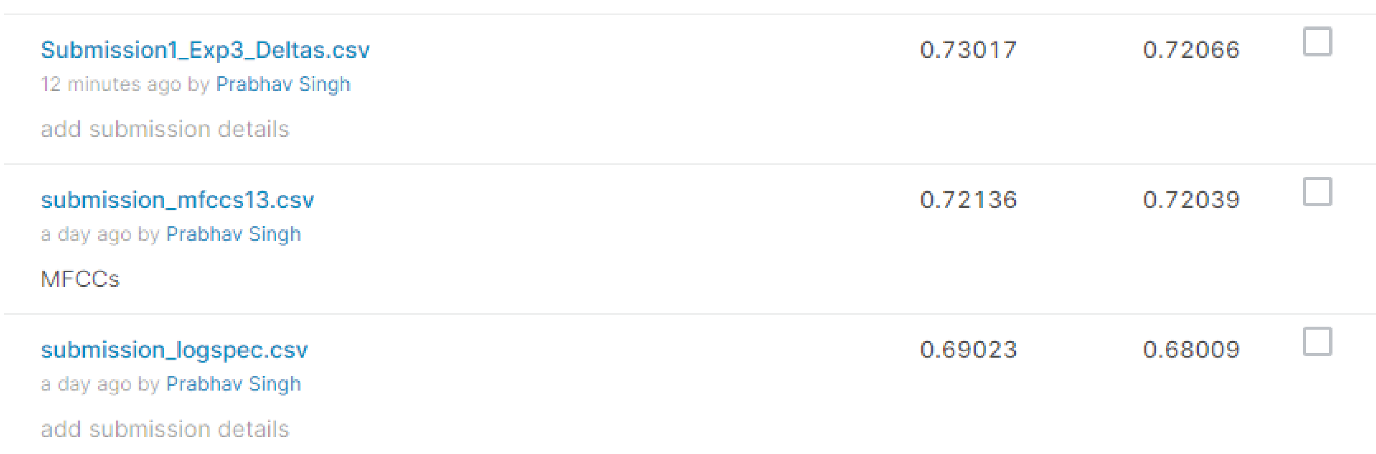
Firstly, the training settings and parameters are shown for all three experiments in the below table:

Experiment	Validation Size	Training Convergence Time	Batch Size	Epochs
Spectrogram	15%	3500 Seconds	16	5 Epochs
MFCCs	15%	500 Seconds	32	10 Epochs
MFCCs and Higher Order Deltas	15%	2400 Seconds	32	20 Epochs

The saved models were tested in batches to preserve RAM and the results were as follows:
** The Rank is calculated manually since the competition has expired and does not provide ranks.*

Experiment	Validation Accuracy	Validation Loss	Accuracy (Public)	Accuracy (Private)	Rank *
Spectrogram	90.010%	0.0412	68.009%	69.023%	948
MFCCs	93.15%	**0.0317**	72.039%	72.136%	889
MFCCs and Higher Order Deltas	**93.22%**	0.0342	**72.066%**	**73.017%**	843

The below screenshot shows the results on Kaggle:



6. Discussion

1. It is clear from the results that, **the use of MFCCs in conjugation with higher order deltas, gives the best results for the dataset.**
2. This could be attributed to the **following reasons:**
    a. The samples are short and a continuous representation including windowing, like spectrograms, are not suitable for that.
    b. The Deltas of MFCCs can represent temporal data well enough and hence outperform MFCCs alone.
    c. MFCCs work upon the human peripheral auditory system. They possess human perception sensitivity with respect to frequencies and hence it is best to understand human speech. [7]
    d. MFCCs are in general, a more compressible feature in respect to spectrograms. [7]
3. Further, one major advantage of MFCC is - Since it is much smaller in feature dimensions, the model can be trained for a longer time with a higher batch size to facilitate better learning.
4. One flaw noticed is that the validation accuracy is much higher than the testing accuracy. This was due to a large imbalance in data points of the “UNKNOWN” class.

6. Other Possible Methods

1. One method that was experimented with, was the use of **Higher Statistical Features (HSFs) of Prosodic Features** like spectral rollof, bandwidth, RMSE etc [8]. Used independently, they did not give good results but they could be used in **multimodal fashion with MFCCs** to improve learning.
2. Measures to prevent overfitting on the UNKNOWN class should be looked into. L2 Regularization and Oversampling could be used.
3. The use of Multimodal Input is worth exploring. For this, **Automatic Speaker Recognition (ASR)** could be used to extract the words itself and this could be used in conjugation with the MFCCs to give better results.

7. Reference In Text

[1] David S., Speech representation and Data Exploration
[2] Zero Padding and its Advantages
[3] Splitting Samples into Smaller Segments, Qing Wei
[4] Understanding the Mel Spectrogram, Leland Roberts
[5] Understanding MFCCs, Prateestha Nair
[6] MFCCs and its Features, Springer (Appendix)
[7] Speaker Recognition Using MFCC and Delta Delta MFCCs with ANNs, Singh et al., IJARSE 2016
[8] Significance of prosodic features for automatic emotion recognition, John et al., AIP 2020

7. Reference Used for Development of Code

[1] Music Feature Extraction in Python, Saket Doshi
[2] Musical Instrument Sound Classification using CNN, Muhammad Ardi