DSE-421 : Project Report

Yash Dhiman and Shirish Shekhar Jha

Roll No: 21317   Roll No: 2410705

Under the Supervision of

Dr. Samiran Das

Department of Data Science and Engineering
Indian Institute of Science Education and Research Bhopal
Bhopal - 462066, India

April, 2025

Exploring Novel Time-Frequency Feature for Audio Classification

Contents

Introduction

Audio classification in AI/ML refers to the process of analyzing and categorizing audio signals into predefined classes using machine learning algorithms. It involves extracting meaningful features from raw audio data to train a model to recognize patterns that distinguish different classes. Audio classification is a very broad field which encompasses many sub-fields like:

- Speech Recognition
- Speaker Identification
- Emotion Detection

Our project aims to analyze, propose, and implement techniques to boost performance of machine learning/deep learning models on audio classification tasks and in particular deals with emotion detection/recognition from given input audio sample. The emotions represent the different classes.

Through this project, we seek to identify novel TF features that improve the robustness of emotion recognition systems, ultimately contributing to the development of more expressive and adaptive audio-based AI models.

Problem Statement

Emotion detection from audio signals is a crucial task with applications, to name a few, in healthcare, psychology, and entertainment. Traditional feature extraction techniques rely on handcrafted spectrogram-based features such as Mel-Frequency Cepstral Coefficients (MFCCs). However, these may not fully capture the intricate time-frequency (TF) characteristics of emotional speech.

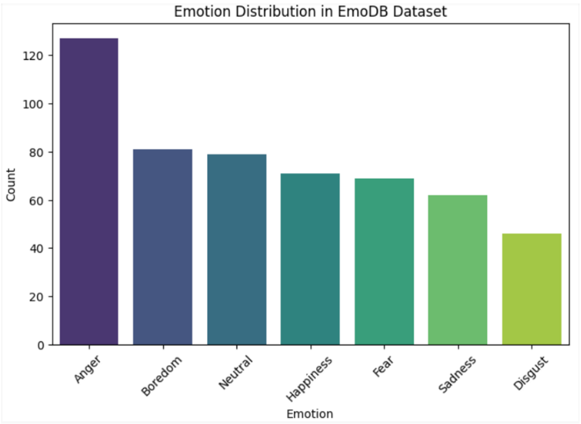
This project aims to explore novel TF representations to enhance the accuracy of emotion classification by leveraging advanced signal processing techniques-such as-Short-Time Fourier Transform (STFT), Wavelet Transform, and Dictionary-Based Learning—with the aim to extract more discriminative features from speech signals.

The goal is to analyze and compare these features with traditional spectrogram methods (MFCC), investigating their effectiveness in preserving temporal and spectral information relevant to different emotions.

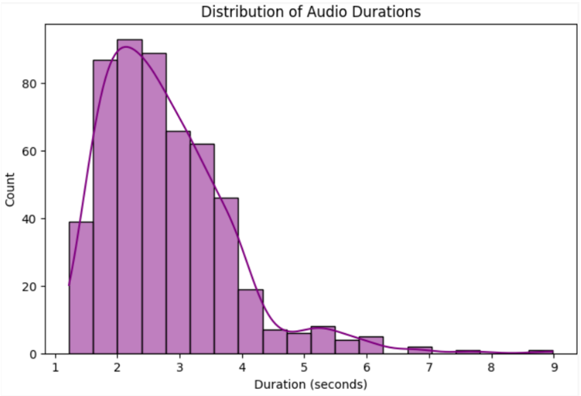
Dataset Analysis

The dataset we are using is EmoDB dataset which is a freely available German emotional database. The database is created by the Institute of Communication Science, Technical University, Berlin, Germany. **10** professional speakers (**5** males and **5** females) participated in data recording. The database contains a total of **535** audio samples. The EmoDB database comprises of **7** emotions:

1) anger	(127 samples)
2) boredom	(81 samples)
3) anxiety/fear	(69 samples)
4) happiness	(71 samples)
5) sadness	(62 samples)
6) disgust	(46 samples)
7) neutral	(79 samples)



Minimum audio sample length	1.225 sec
Maximum audio sample length	8.978 sec
Mean audio sample length	2.779 sec
Standard Deviation	1.028 sec



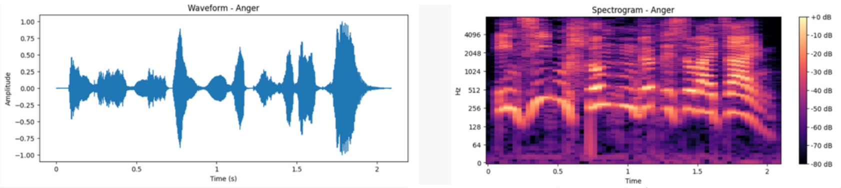
25th percentile : 2.026 sec
(25% of audio files have a duration less than this)

50th percentile : 2.590 sec
(50% of audio files have a duration less than this)

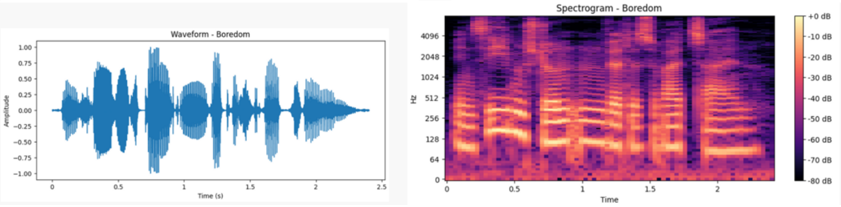
75th percentile : 3.308 sec
(75% of audio files have a duration less than this)

Emotion Sample Analysis

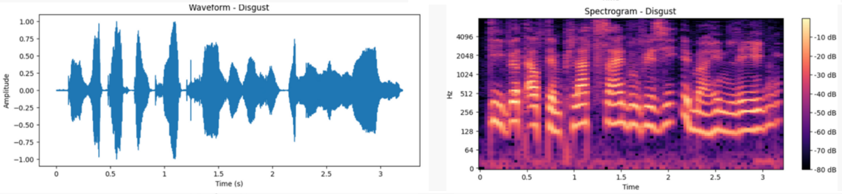
Anger :



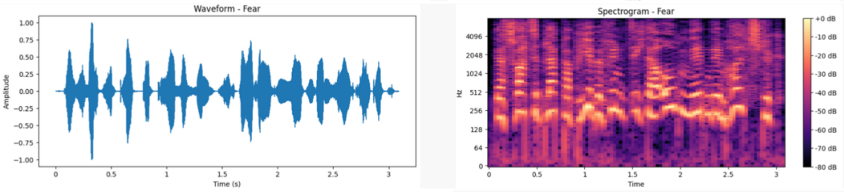
Boredom :



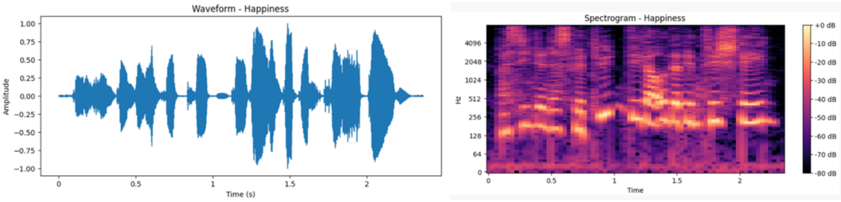
Disgust :



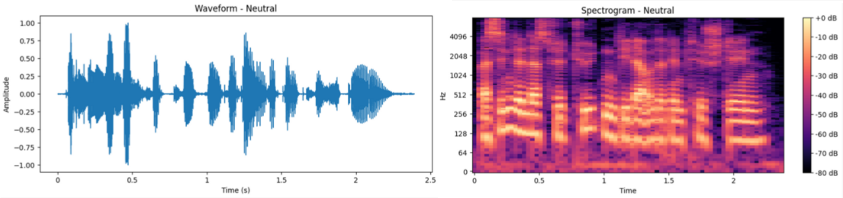
Fear :



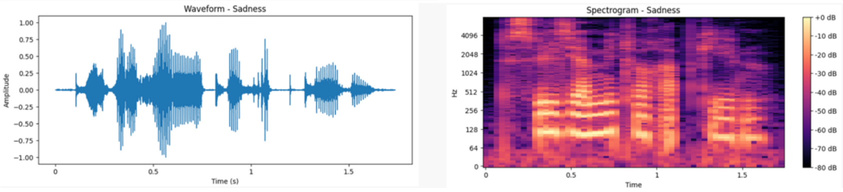
Happiness :



Neutral :



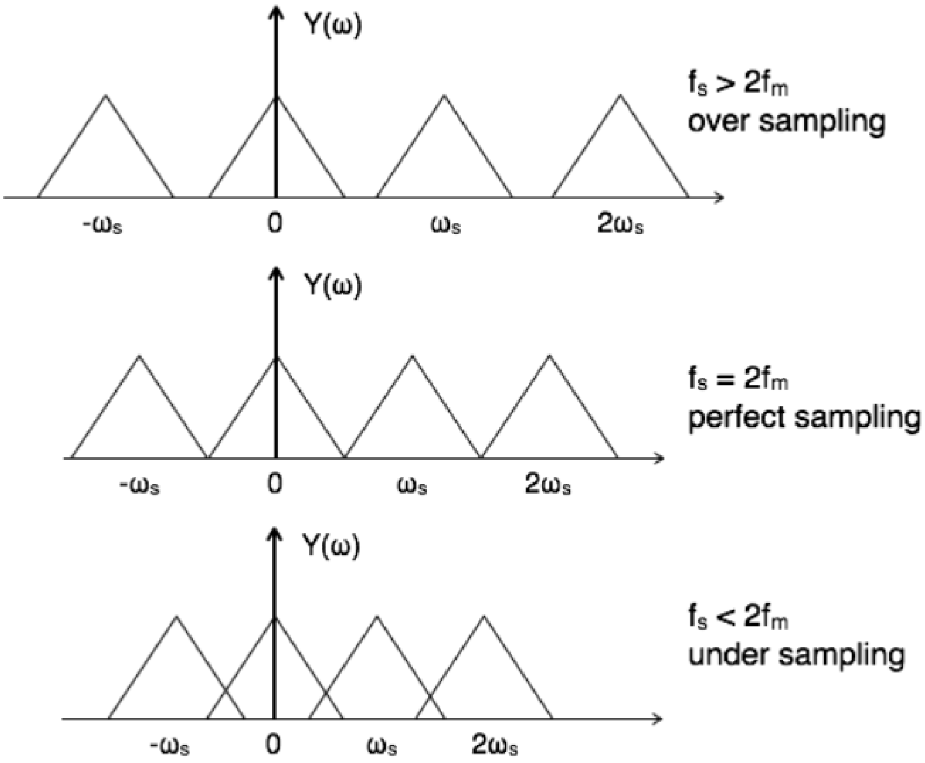
Sadness :



Theory

1. De-mystifying the Sampling Rates

The data was recorded at a 48-kHz sampling rate and then down-sampled to 16-kHz. Human hearing range varies from 20Hz-20KHz, hence in accordance to the Sampling theorem the sampling frequency should be >= 40KHz which justifies the 48-KHz rate. It is interesting to note however that the speech related information (essential characteristics of an audio signal that make speech intelligible and meaningful) is contained in frequencies below 8-KHz therefore we can downsample the audio samples to 16KHz. The high frequency(>8KHz) is mostly environmental sound(e.g. chirping of bird) and the sound produced by musical instruments or vehicles which do not contribute to emotion detection from human speech.



2. Existing Methodology

The task of audio classification is a supervised machine learning task, which means that we provide audio samples with class labels. General pipeline is:

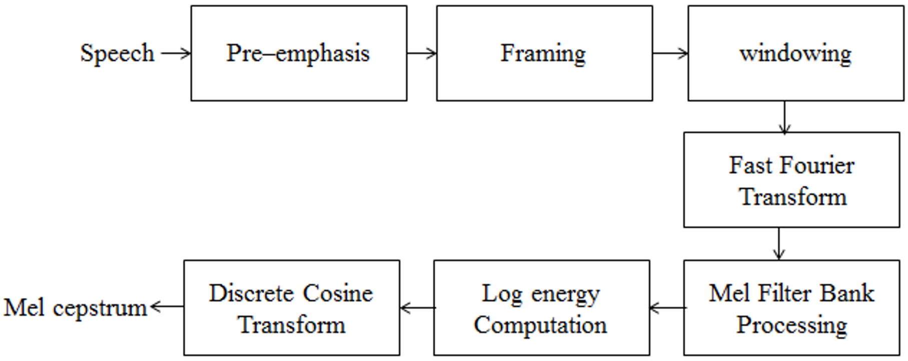
- Data Collection – Gathering audio recordings relevant to the classification task (EmoDB in our case).
- Feature Extraction – Converting raw audio signals into meaningful numerical representations (e.g. spectrogram techniques like Mel-frequency cepstral coefficients (MFCCs)).
- Model Training – Using machine learning or deep learning models (e.g. CNNs, RNNs, transformers) to learn patterns in the extracted features.
- Inference & Prediction – Using the trained model to classify new, unseen audio recordings into the predefined categories.

Steps 1, 3 and 4 are common to all machine learning tasks. However, it the quality of features which determines the quality of learned model. The method of extracting features varies from task to task.

The most widely used (state of art method) feature extraction technique in audio classification tasks is MFCC. It is the feature extraction technique used in voice controlled-virtual assistants like Siri (by Apple) and also in Google Voice Assistant. In order to be able to propose novel TF feature we need to understand the working principle of existing feature extraction techniques.

3. MFCC

Mel Frequency Cepstral Coefficients (MFCC) is a feature extraction technique used heavily in audio classification tasks. It captures the salient features from speech and enables computers to differentiate between distinct words, and sounds. It works well because it is based on the simulation of exact physical phenomena by which humans produce sounds and also how the ears perceive sound. A complete workflow pipeline of MFCCs is given below:



How MFCC helps in emotion detection ?

In practice the total number of features per frame is **39**. However, the output of DCT in the pipeline shown above is **13** values per frame. MFCCs describe the speech spectrum at a given time, but speech is dynamic i.e. MFCCs alone represent only the spectral content of each speech frame but do not capture how it changes over time (frame to frame). To capture time-dependent changes, we compute delta (Δ) and delta-delta (Δ^2) features and keep first **13** of each giving a total of **39** features per frame of individual audio signal.

Different emotions affect how we produce speech, leading to changes in the spectral characteristics of speech signals. Since MFCCs capture the spectral envelope of speech, they can effectively differentiate between emotions based on how they shape the variation/change of MFCC coefficients.

4. Novelty Proposed

4.1 Fractional Calculus! But What is it ?

Fractional calculus extends the concept of integer-order differentiation and integration to non-integer (real or complex) orders. It is a powerful tool for modeling systems with memory, hereditary properties, and anomalous dynamics.

Applications:

- Control theory, viscoelastic materials
- Biomedical signal processing
- Audio and speech modeling

In our context, we apply a **fractional derivative of order** $\alpha = 0.9$ on audio signals.

4.2 How to calculate fractional derivatives ?

Grünwald–Letnikov Fractional Derivative

One classical definition of the fractional derivative is the **Grünwald–Letnikov (GL) approach**.
The GL α-order derivative of function $f(t)$ is defined as:

$$D^\alpha f(t) = \lim_{h \to 0} \frac{1}{h^\alpha} \sum_{k=0}^{\infty} (-1)^k \binom{\alpha}{k} f(t - kh)$$

Where:

$$\binom{\alpha}{k} = \frac{\Gamma(\alpha + 1)}{\Gamma(k + 1)\Gamma(\alpha - k + 1)}$$

Key Insight: The derivative at time t depends on all previous values of $f(t)$ - introducing a memory effect. Unlike classical derivatives, **fractional derivatives depend on the entire history** of the signal. The current output is a weighted sum of all previous inputs:

$$D^\alpha f(t) \sim f(t), f(t - h), f(t - 2h), \ldots$$

This is particularly powerful for analyzing speech signals which exhibit temporal dependencies.

4.3 Integer Order FT vs Fractional Order FT

Applying a fractional derivative ($\alpha = 0.9$) modifies the shape and dynamics of the original signal enhancing subtle transitions and introducing memory-informed signal behavior.

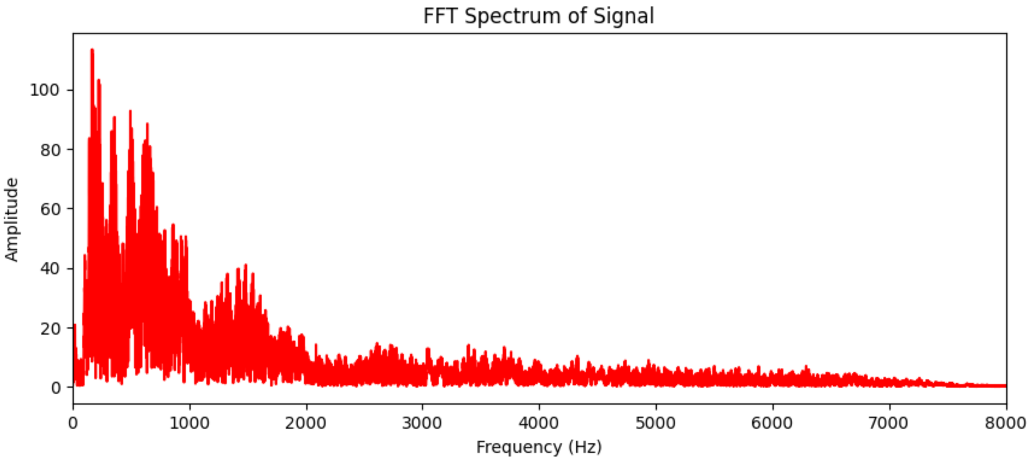


Figure 1: Raw FFT

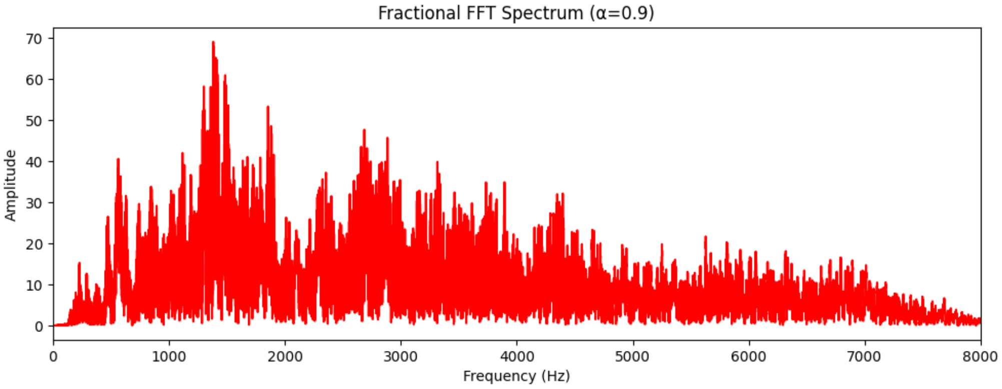
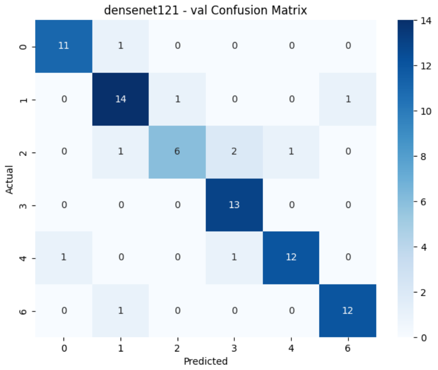


Figure 2: Fractional FFT ($\alpha = 0.9$)

Results

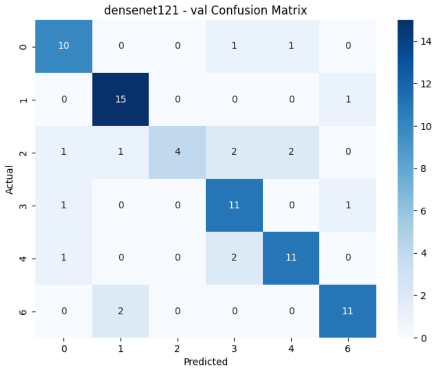
Model: DenseNet121

Class	Prec.	Recall	F1
0	0.92	0.92	0.92
1	0.82	0.88	0.85
2	0.86	0.60	0.71
3	0.81	1.00	0.90
4	0.92	0.86	0.89
6	0.92	0.92	0.92
Acc.		0.87	
Macro avg	0.88	0.86	0.86
Weighted avg	0.87	0.87	0.87



Integer Based

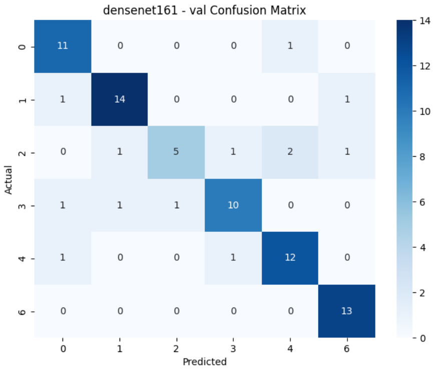
Class	Prec.	Recall	F1
0	0.77	0.83	0.80
1	0.83	0.94	0.88
2	1.00	0.40	0.57
3	0.69	0.85	0.76
4	0.79	0.79	0.79
6	0.85	0.85	0.85
Acc.		0.79	
Macro avg	0.82	0.77	0.77
Weighted avg	0.81	0.79	0.79



Fractional Based ($\alpha = 0.9$)

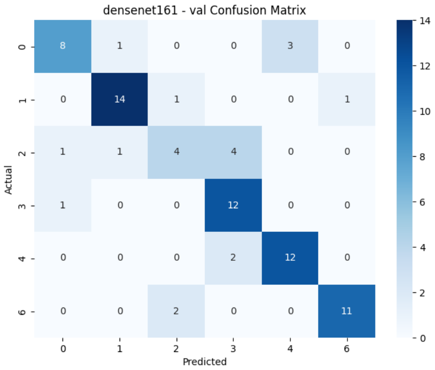
Model: DenseNet161

Class	Prec.	Recall	F1
0	0.79	0.92	0.85
1	0.88	0.88	0.88
2	0.83	0.50	0.62
3	0.83	0.77	0.80
4	0.80	0.86	0.83
6	0.87	1.00	0.93
Acc.		0.83	
Macro avg	0.83	0.82	0.82
Weighted avg	0.83	0.83	0.83



Integer Based

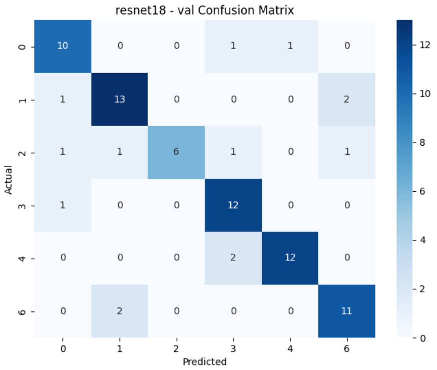
Class	Prec.	Recall	F1
0	0.80	0.67	0.73
1	0.88	0.88	0.88
2	0.57	0.40	0.47
3	0.67	0.92	0.77
4	0.80	0.86	0.83
6	0.92	0.85	0.88
Acc.		0.78	
Macro avg	0.77	0.76	0.76
Weighted avg	0.78	0.78	0.78



Fractional Based ($\alpha = 0.9$)

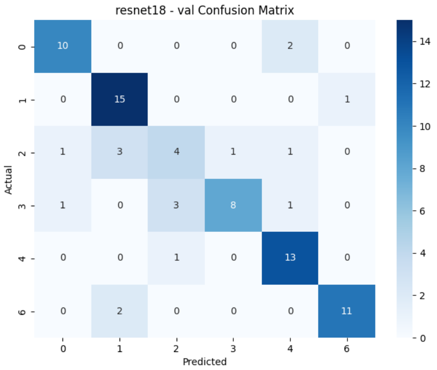
Model: ResNet18

Class	Prec.	Recall	F1
0	0.77	0.83	0.80
1	0.81	0.81	0.81
2	1.00	0.60	0.75
3	0.75	0.92	0.83
4	0.92	0.86	0.89
6	0.79	0.85	0.81
Acc.		0.82	
Macro avg	0.84	0.81	0.82
Weighted avg	0.83	0.82	0.82



Integer Based

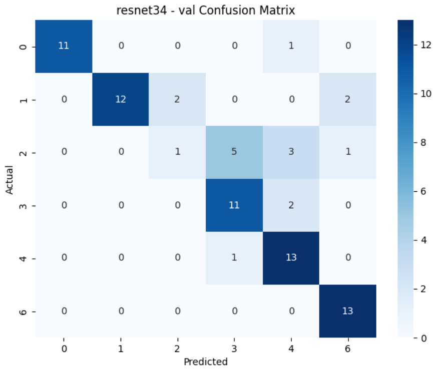
Class	Prec.	Recall	F1
0	0.83	0.83	0.83
1	0.75	0.94	0.83
2	0.50	0.40	0.44
3	0.89	0.62	0.73
4	0.76	0.93	0.84
6	0.92	0.85	0.88
Acc.		0.78	
Macro avg	0.78	0.76	0.76
Weighted avg	0.78	0.78	0.77



Fractional Based ($\alpha = 0.9$)

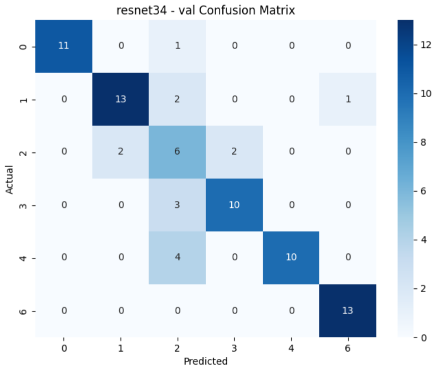
Model: ResNet34

Class	Prec.	Recall	F1
0	1.00	0.92	0.96
1	1.00	0.75	0.86
2	0.33	0.10	0.15
3	0.65	0.85	0.73
4	0.68	0.93	0.79
6	0.81	1.00	0.90
Acc.		0.78	
Macro avg	0.75	0.76	0.73
Weighted avg	0.77	0.78	0.76



Integer Based

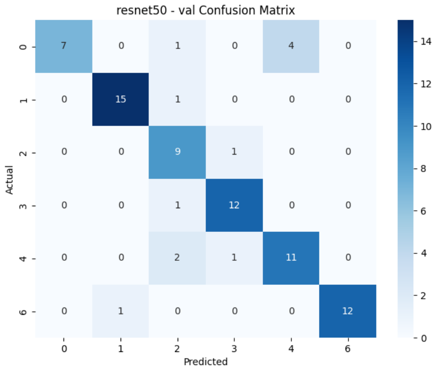
Class	Prec.	Recall	F1
0	1.00	0.92	0.96
1	0.87	0.81	0.84
2	0.38	0.60	0.46
3	0.83	0.77	0.80
4	1.00	0.71	0.83
6	0.93	1.00	0.96
Acc.		0.81	
Macro avg	0.83	0.80	0.81
Weighted avg	0.85	0.81	0.82



Fractional Based ($\alpha = 0.9$)

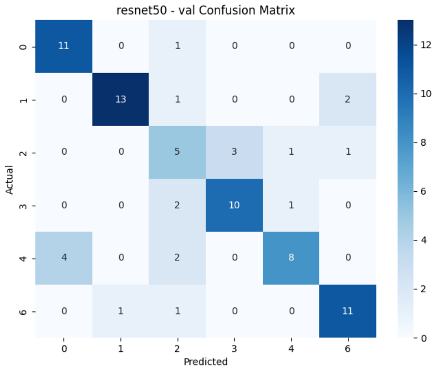
Model: ResNet50

Class	Prec.	Recall	F1
0	1.00	0.58	0.74
1	0.94	0.94	0.94
2	0.64	0.90	0.75
3	0.86	0.92	0.89
4	0.73	0.79	0.76
6	1.00	0.92	0.96
Acc.		0.85	
Macro avg	0.86	0.84	0.84
Weighted avg	0.87	0.85	0.85



Integer Based

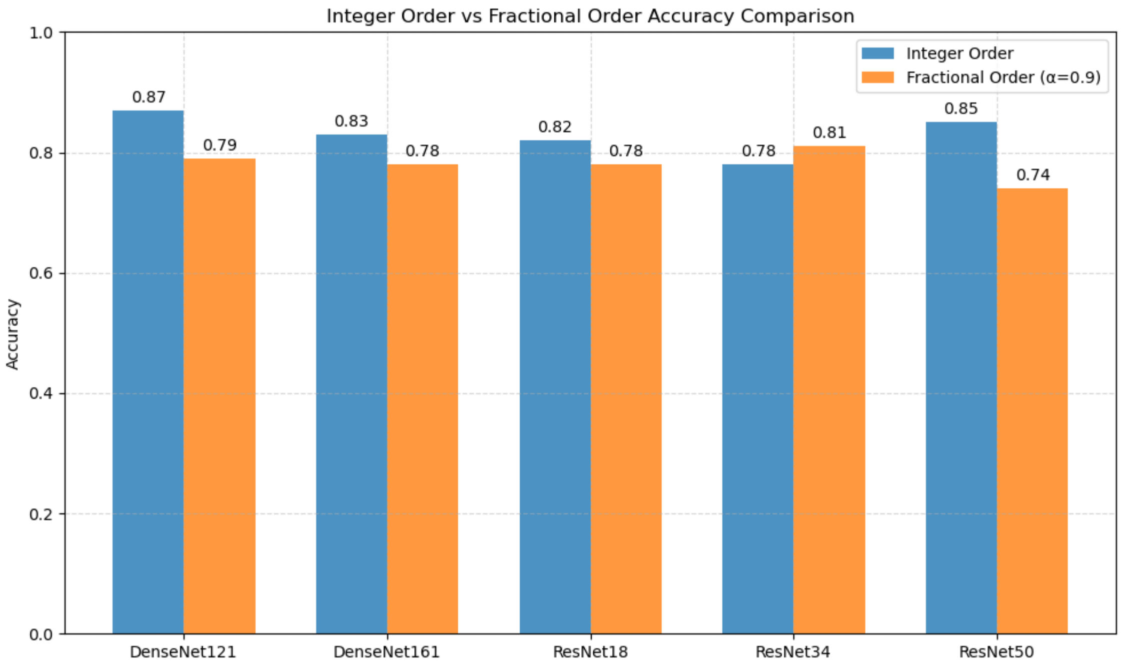
Class	Prec.	Recall	F1
0	0.73	0.92	0.81
1	0.93	0.81	0.87
2	0.42	0.50	0.45
3	0.77	0.77	0.77
4	0.80	0.57	0.67
6	0.79	0.85	0.81
Acc.		0.74	
Macro avg	0.74	0.74	0.73
Weighted avg	0.76	0.74	0.75



Fractional Based ($\alpha = 0.9$)

Analysis

The results obtained across different models highlight the comparative impact of integer-order and fractional-order feature extractions on model performance. For the DenseNet121 model, integer-order based features achieved a significantly higher accuracy of 87%, compared to 79% under fractional-order ($\alpha = 0.9$). A similar trend is observed in DenseNet161, where the integer-order method achieved 83% accuracy, whereas fractional-order reduced it to 78%. The ResNet18 model also followed this pattern, recording an 82% accuracy with integer-order features and a lower 78% with fractional-order features.



Model performance comparison based upon feature types

Interestingly, the ResNet34 model demonstrated an exception. Here, fractional-order features led to a slight improvement in performance, raising the accuracy from 78% (integer-order) to 81% (fractional-order). This suggests that certain architectures, particularly deeper residual networks like ResNet34, may benefit from fractional-order feature modifications, possibly due to enhanced capacity to capture finer feature dynamics. In contrast, ResNet50 showed a significant performance drop, with integer-order features achieving 85% accuracy and fractional-order extraction only reaching 74%, emphasizing the general robustness of integer-order based features for larger architectures.

Overall, integer-order methods consistently provided better precision, recall, and F1-scores across most models, establishing them as a more reliable feature extraction approach. However, the improvement seen with ResNet34 indicates that fractional-order features, when carefully tuned, can offer meaningful advantages in specific contexts. The graphical comparison below further reinforces these findings, showing a clear trend favoring integer-order performance, except in the case of ResNet34 where fractional-order slightly outperforms.

Conclusion

When working on a research problem it is very crucial to start from the scratch and then proceed in a bottom up-manner understanding each and every step. A similar approach is taken in this project starting from exploration of dataset and then moving on to existing feature extraction methodology along with novelty that was tested, finally concluding with their performance evaluations on various Models. We make the following inferences:

- 5 different models were deployed and tested.
- DenseNet121 was found to be the best performing model amongst them when compared on the basis of Accuracy, Macro Average F1 and Weighted Average F1.
- Amongst DenseNet121, the integer-order type gave consistently higher score values than the fractional-order type.
- In most of the cases, both integer and fractional order-based features gave comparable performances.

Future Work: We can perform an ablation study with multiple orders of fractional features to highlight their inferences and use more complex models to understand their contribution.

References and Literature Review

The following resources and references were used in the completion of this project.

GitHub Repositories

[1] https://github.com/OmarMedhat22/Sound-Classification-Wavelet-Transform

[2] https://github.com/AdityaDutt/Audio-Classification-Using-Wavelet-Transform

[3] https://github.com/samsad35/VQ-MAE-S-code

[4] https://github.com/Jiaxin-Ye/TIM-Net_SER

[5] https://github.com/audeering/ser-uncertainty-quantification

arXiv Literature

[6] https://arxiv.org/pdf/2403.00887v1

[7] https://arxiv.org/pdf/2304.11117v1

[8] http://arxiv.org/pdf/2211.08233v3

[9] https://arxiv.org/html/2410.08321v1