IT350 Assignment 4 - Report

Sachin Prasanna - 211T058

March 22, 2024

1 Problem Statement

a) Take any dataset (apart from Images) from different domain, convert the data samples into meaningful Images . Apply CNN and Deep CNN with parameter tuning (Filter size, number of filters, Row stride Column stride...etc) compare the results. (3 +3 Marks)

b) Visualize the data samples of different classes after converting into images, visualize the feature map at each layer of Deep CNN. (4 marks)

c) Download any shorts/reels video of Left hand bowler taking the wicket of Right hand batsman and use CNN to convert the video into Right hand bowler taking the wicket of Left hand batsman. (Sample Video https://www.youtube.com/watch?v=WiyqKP0ZSiI) (Bonus marks 2)

2 Dataset

The ESC-50 dataset is a labeled collection of 2000 environmental audio recordings suitable for benchmarking methods of environmental sound classification. The dataset consists of 5-second-long recordings organized into 50 semantical classes (with 40 examples per class) loosely arranged into 5 major categories. I will be predicting these audios on these 5 major categories. Clips in this dataset have been manually extracted from public field recordings gathered by the Freesound.org project. The dataset has been prearranged into 5 folds for comparable cross-validation, making sure that fragments from the same original source file are contained in a single fold.

3 Data Visualisation

A bar graph is plotted in Figure 1 and seen that there are equal instances for each class. The analysis indicates that the dataset is balanced, meaning that each category has a relatively equal number of samples. A balanced dataset is desirable for training machine learning models as it helps prevent biases towards any particular class.

Another bar graph is plotted in Figure 2 and seen that there are equal instances for each major category. The analysis indicates that the dataset is balanced, meaning that

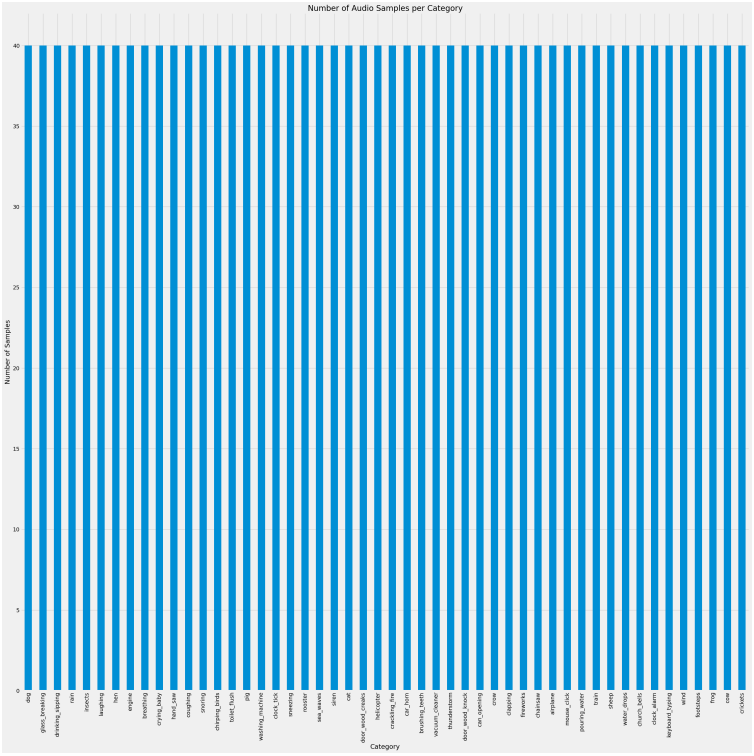

Figure 1: Bar Plots of data with labels

each category has a relatively equal number of samples. A balanced dataset is desirable for training machine learning models as it helps prevent biases towards any particular class.

Then the audio waves are visualised and plotted. To continue the analysis, we can further inspect and analyze the audio waves. Many things can be done, but we will compute the spectrogram, which are meaningful images of the audio. Audio visualisation is shown in Figure 3.

Then, Fourier Transform analysis is performed on a selection of audio signals. This analysis provides insight into the frequency content of the audio signals, which can be useful for tasks such as identifying dominant frequencies or distinguishing between different types of sounds. It is also used to compute spectrogram. FFT analysis is shown in Figure 4.

Spectrogram analysis is performed and audio is converted to images. Results are shown in Figure 5.

Mel Spectrogram analysis is performed and audio is converted to images. A mel spectrogram logarithmically renders frequencies above a certain threshold (the corner frequency). For example, in the linearly scaled spectrogram, the vertical space between 1,000 and 2,000Hz is half of the vertical space between 2,000Hz and 4,000Hz. In the mel spectrogram, the space between those ranges is approximately the same. This scaling is analogous to human hearing, where we find it easier to distinguish between similar low frequency sounds than similar high frequency sounds. Results are shown in Figure 6.

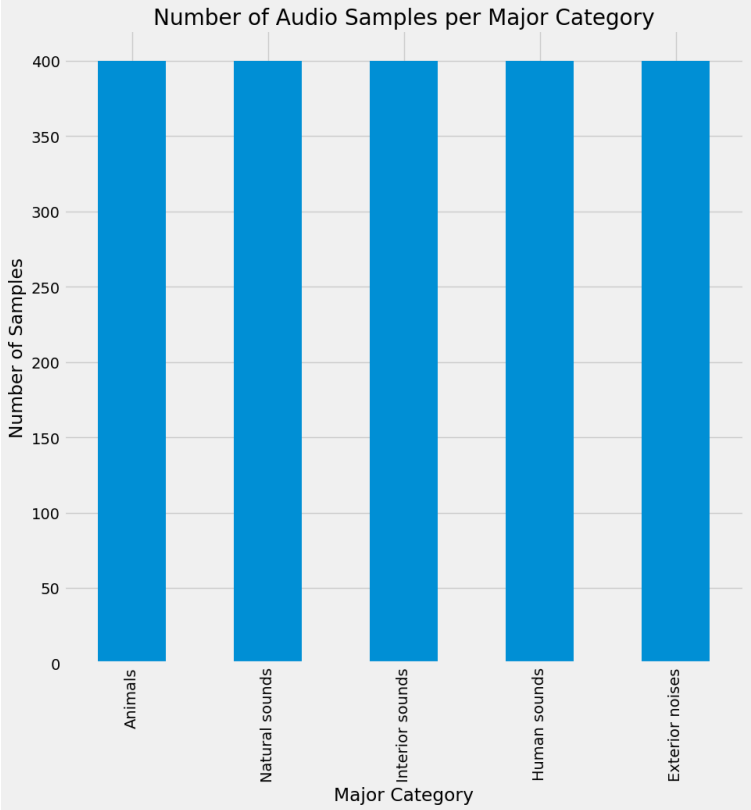

Figure 2: Bar Plots of data with Major Category

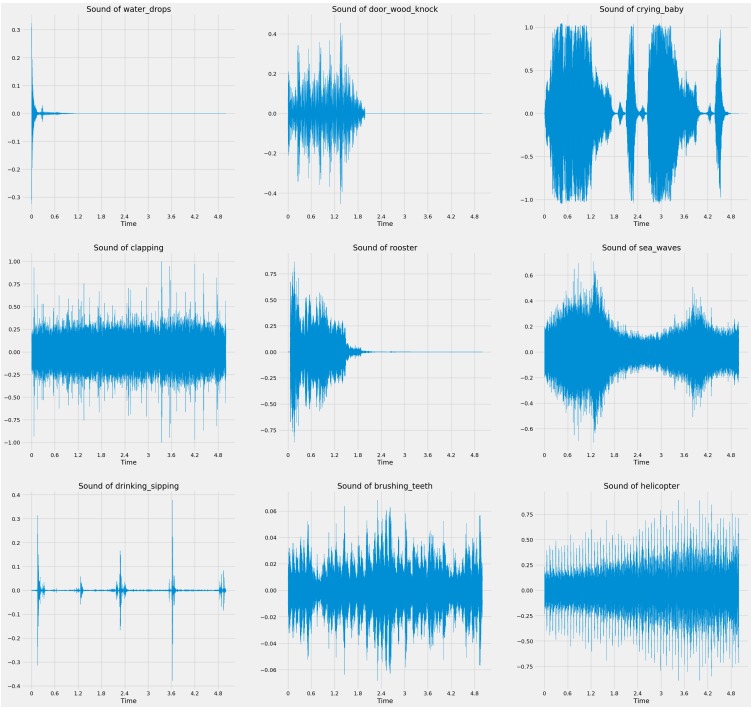

Figure 3: Audio waves of different sounds

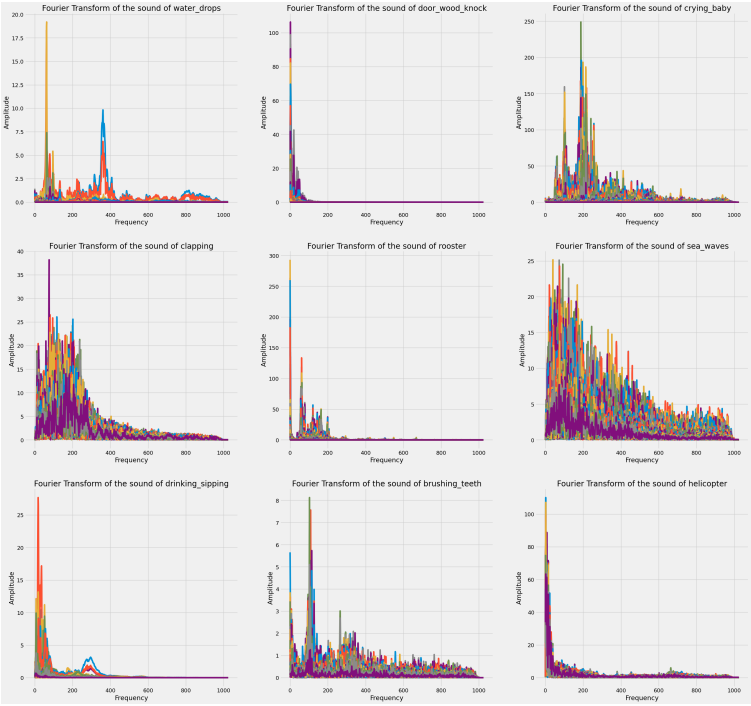

Figure 4: Fourier Transform of audio waves

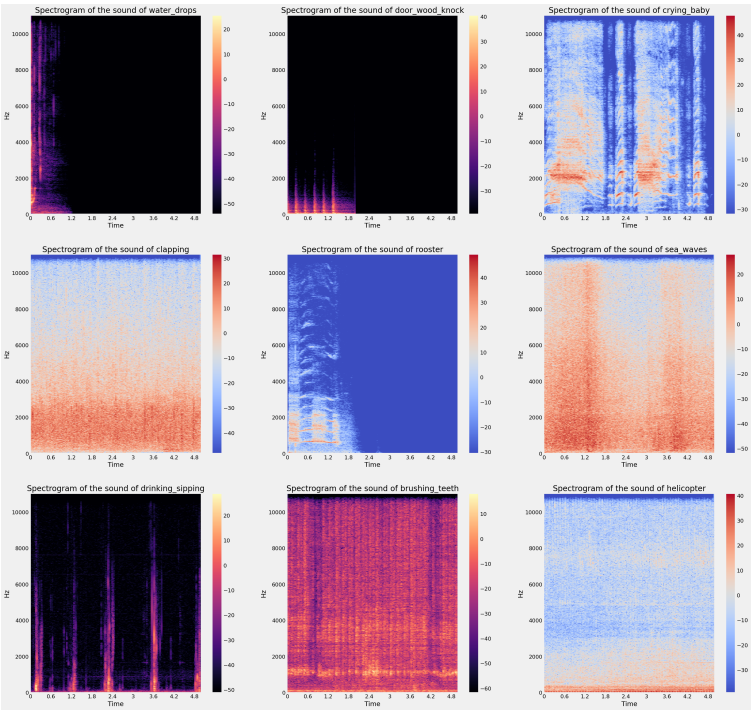

Figure 5: Spectrograms of audio files to Images

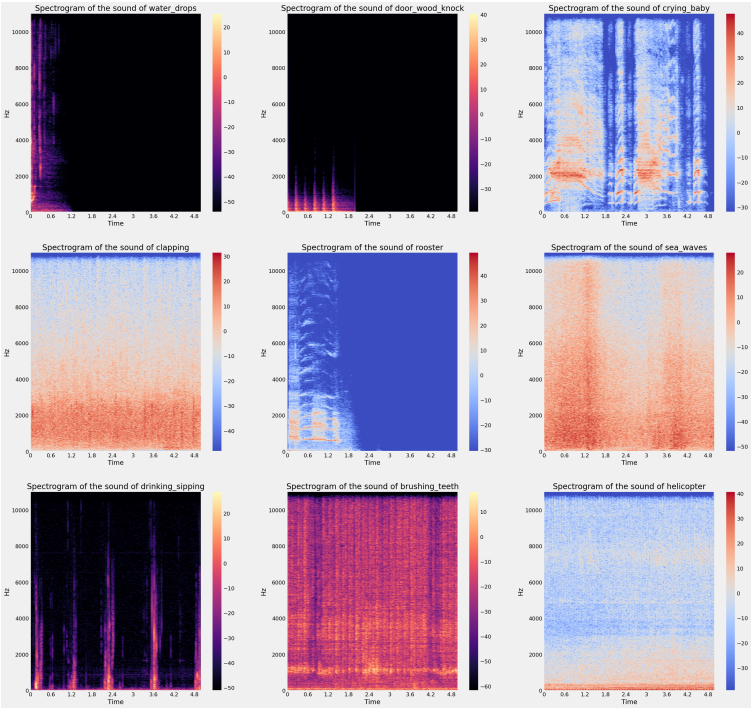

Figure 6: Mel Spectrograms of audio files to Images

4 Results of Simple CNN

The results of the Simple CNNs model with parameter tuning is displayed in Table 1.

5 Results of Deep CNN

The results of the Deep CNNs model with parameter tuning is displayed in Table 2.

6 Images visualised after each Layer of CNN

6.1 Simple CNN

The Simple CNN had 3 convolution layers and their feature maps are all visualised from Figure 7 to 9.

6.2 Deep CNN

The Simple CNN had 6 convolution layers and their feature maps are all visualised from Figure 10 to 15.

Table 1: Simple CNN Models Performance

Model Description	Train Loss	Train Accuracy	Validation Loss	Validation Accuracy
Vanilla CNN	0.9508	95.41%	1.2512	66.0%
With Padding = 1	0.9512	95.41%	1.2442	67.0%
With Kernel Size 4x4	0.9816	92.47%	1.2888	61.33%
With Stride 2x2	1.0612	84.94%	1.2788	61.67%
With More Filters	0.9530	95.12%	1.2321	67.33%
With Kernel Size 4x4 and Padding = 1	0.9327	97.18%	1.2303	67.0%
With Stride 2x2 and Padding = 1	1.0716	84.82%	1.2527	66.33%
With Kernel Size 4x4, Stride 2x2 and Padding = 1	1.0321	87.59%	1.2453	66.0%
With Stride 2x1	1.0535	85.24%	1.2860	61.0%
With Stride 1x2	1.0165	89.24%	1.2306	67.0%
With 2 Layers	0.9554	94.88%	1.2457	66.33%

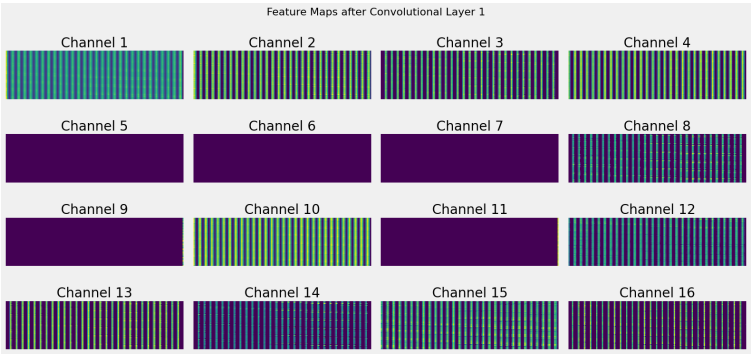

Figure 7: Feature maps after convolutional layer 1

Table 2: Deep CNN Models Performance

Model Description	Train Loss	Train Accuracy	Validation Loss	Validation Accuracy
Vanilla Deep CNN	1.2530	65.06%	1.4099	49.0%
With Padding = 1	1.2777	62.35%	1.3979	49.67%
With Uniform Kernel Size of 3x3	1.2993	60.59%	1.4106	49.0%
With Uniform Kernel Size of 4x4	1.3185	58.82%	1.4890	41.33%
With Uniform Stride of 2x2 and Padding = 2	1.3172	57.94%	1.3799	52.33%
With Increased number of Filters	1.6085	29.53%	1.6042	30.0%
With Increased Kernel Size of 5x5 and Padding = 1	1.2932	61.12%	1.4604	43.67%
With Uniformly Decreasing Kernel Size	1.1981	71.0%	1.3925	50.0%
With Increased Row Stride	1.3228	57.65%	1.3971	49.67%
With Increased Coloumn Stride	1.2418	66.12%	1.3483	55.33%
With Batch Normalization	1.1320	77.65%	1.2308	68.0%
With Batch Normalisation and Uniformly Decreasing Stride	1.0842	82.82%	1.2834	60.67%
With Batch Normalisation and Uniform Stride of 2x2 and Padding = 2	1.1104	79.29%	1.2120	69.0%
With Batch Normalisation and With Increased Coloumn Stride	1.1200	79.0%	1.2766	62.67%
With Batch Normalisation and Padding = 1	1.1636	74.47%	1.2781	63.0%

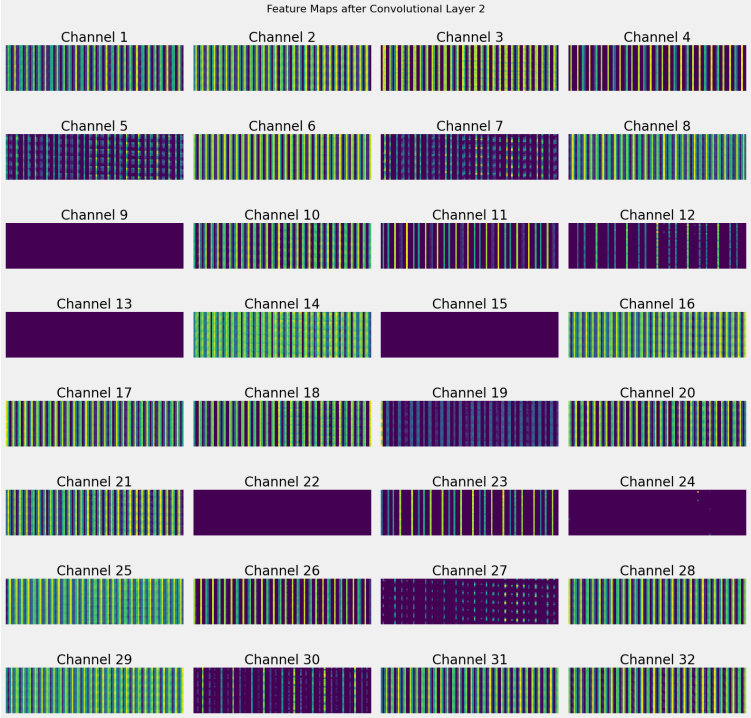

Figure 8: Feature maps after convolutional layer 2

7 Code

The code associated with this report and assignment can be found on the GitHub Repository - https://github.com/sachinprasanna7/Audio-Image-CNN

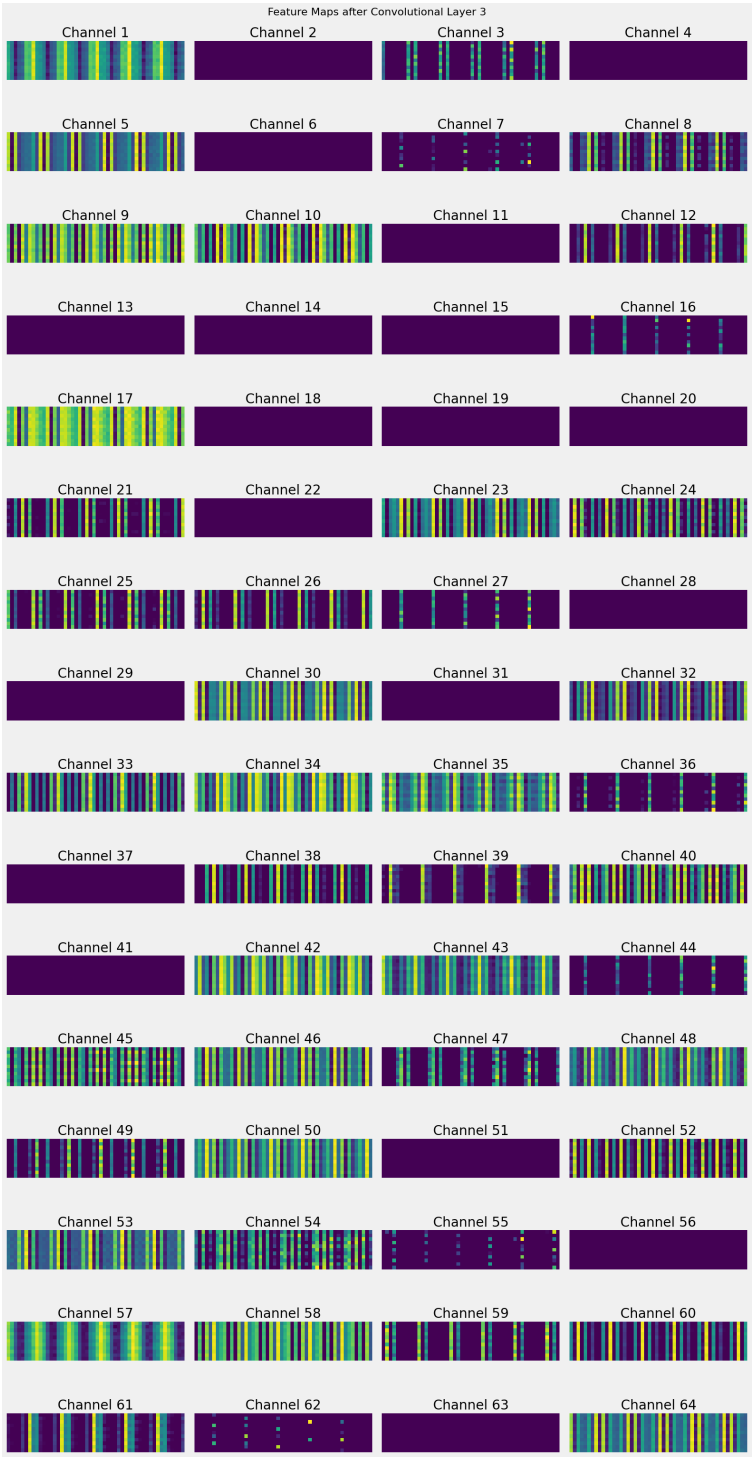

Figure 9: Feature maps after convolutional layer 3

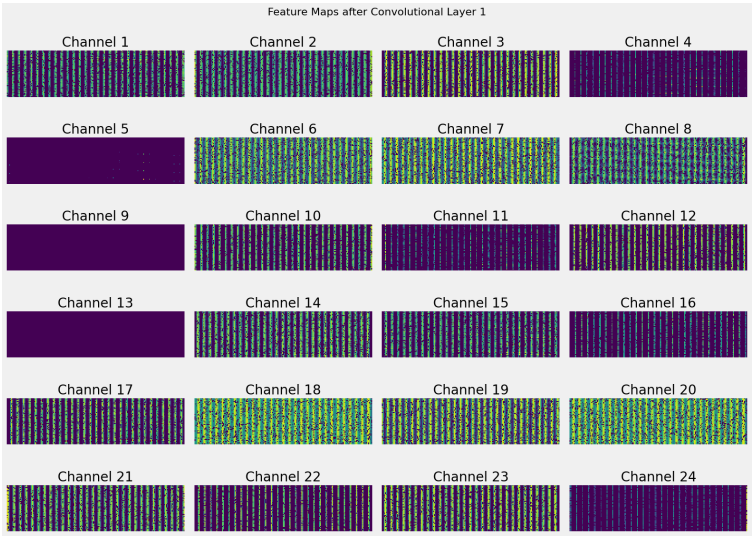

Figure 10: Feature maps after convolutional layer 1

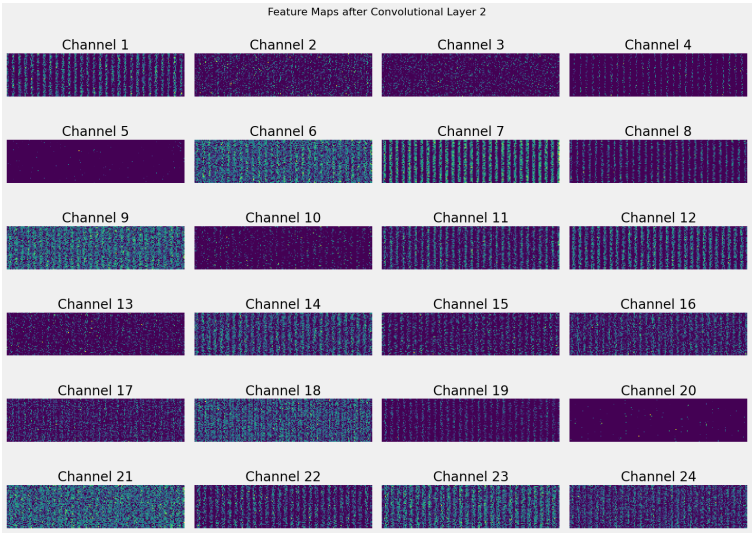

Figure 11: Feature maps after convolutional layer 2

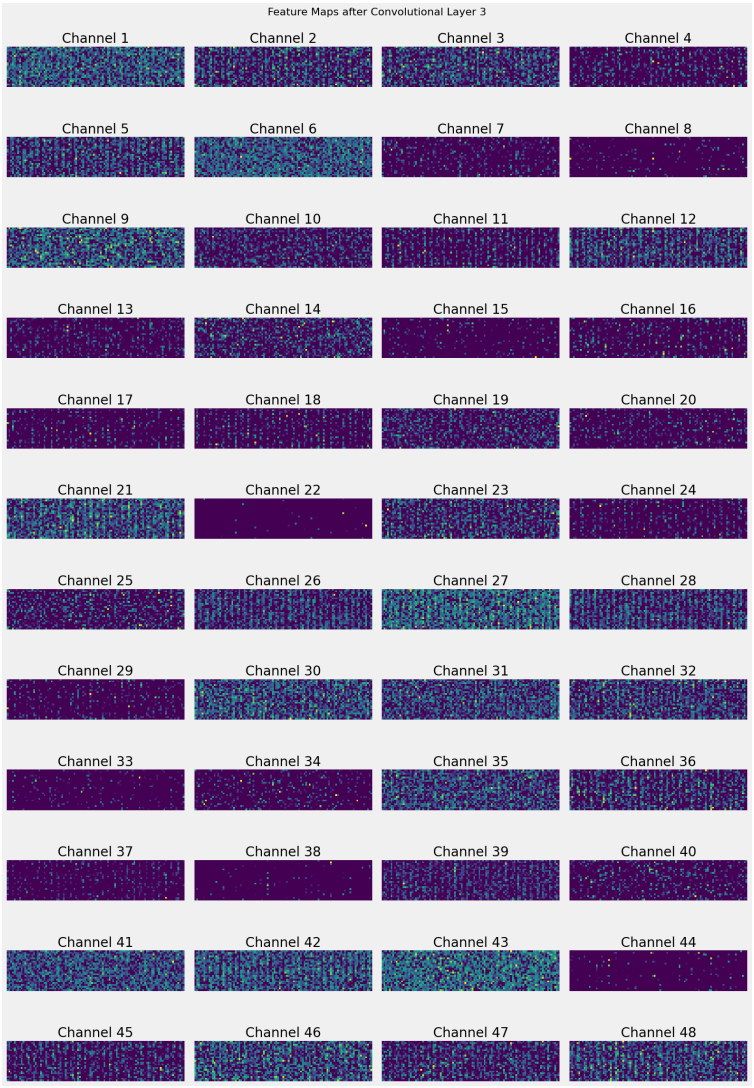

Figure 12: Feature maps after convolutional layer 3

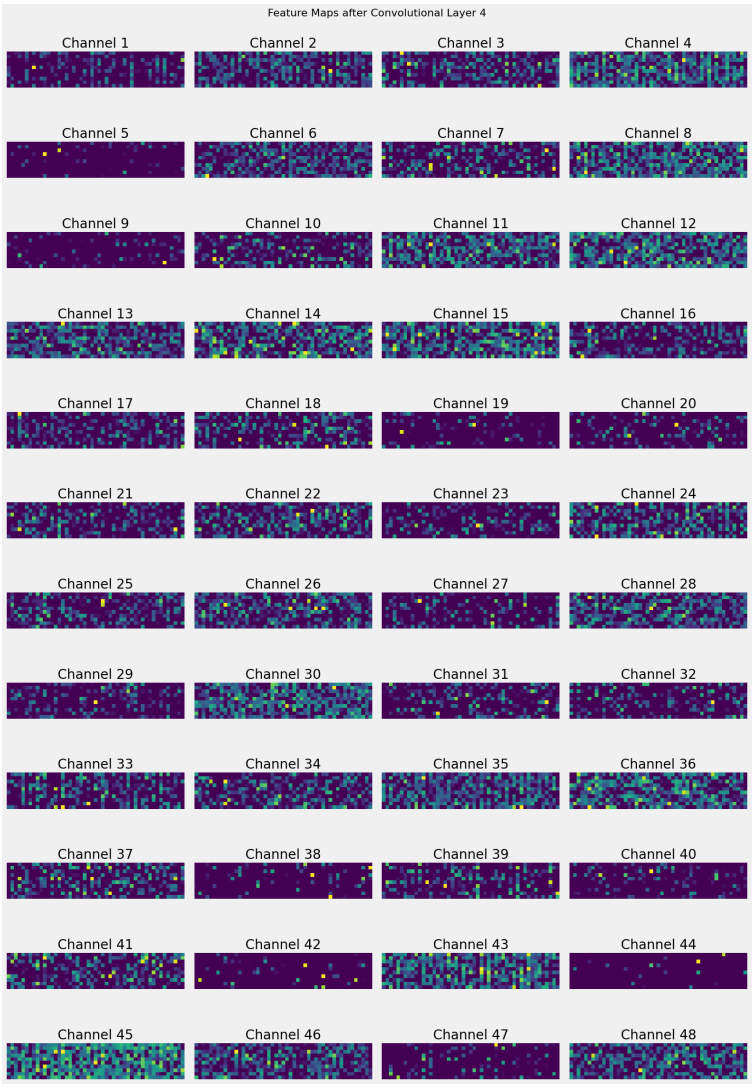

Figure 13: Feature maps after convolutional layer 4

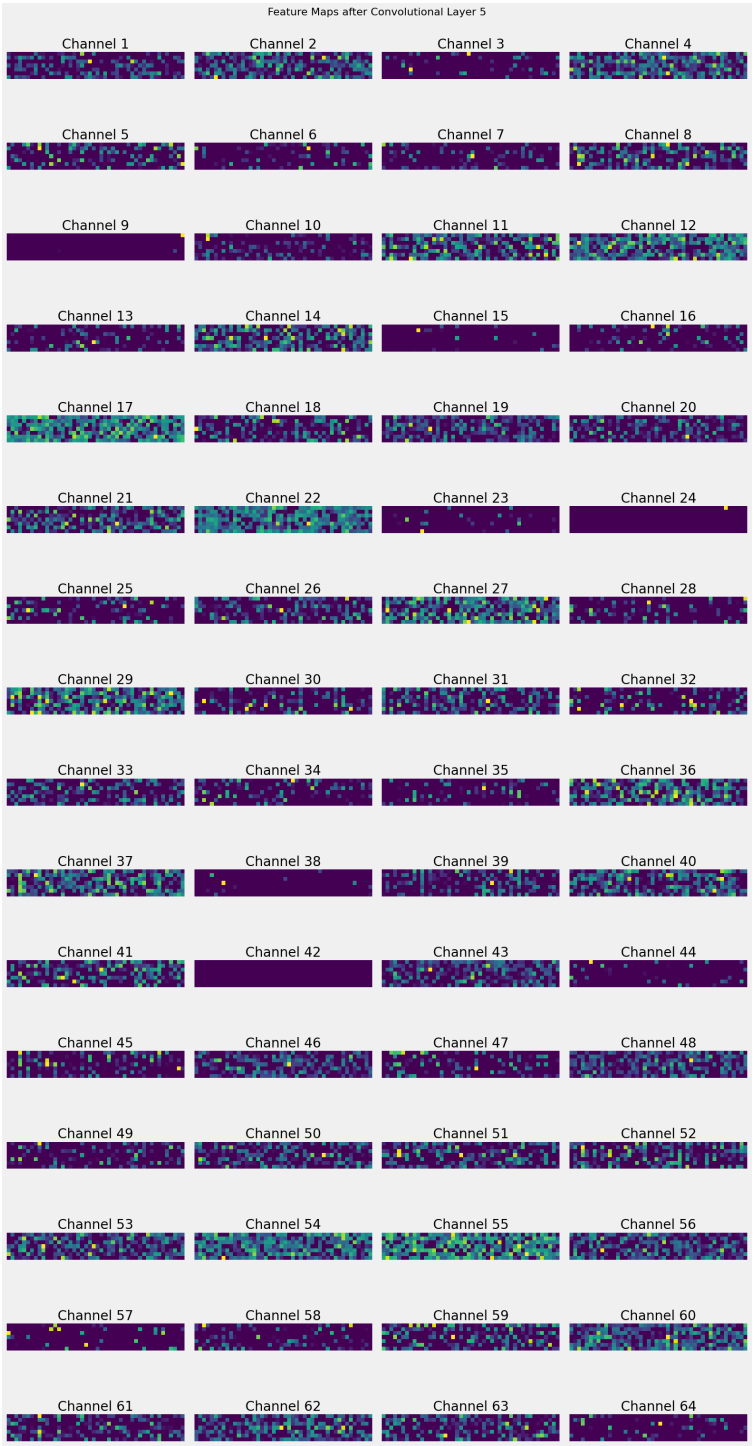

Figure 14: Feature maps after convolutional layer 5

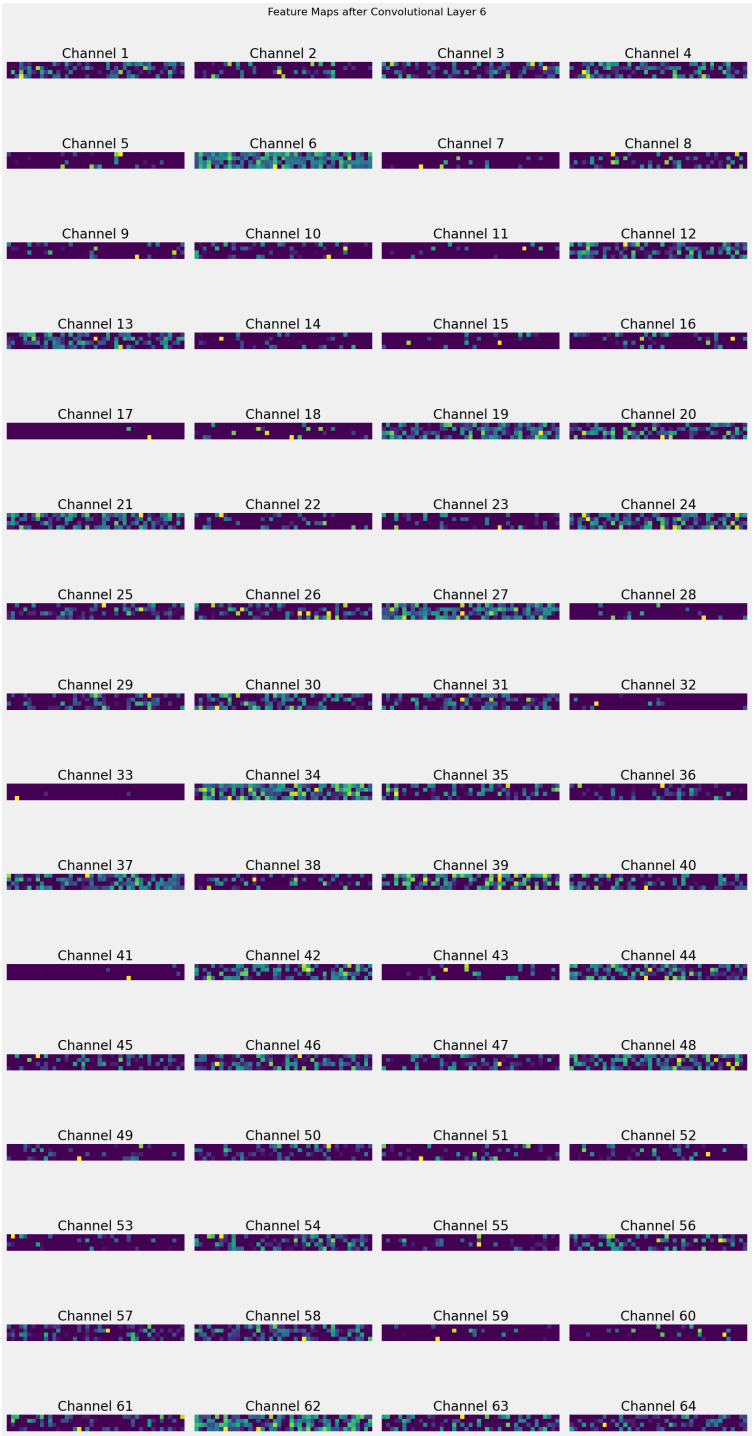

Figure 15: Feature maps after convolutional layer 6