Acoustic Event Classification using Deep Convolutional Neural Network

Project done as a part of the course EE698V (Machine Learning in Signal Processing)

Jaskaran Kalra
180322

Shivam Tulsyan
180723

***Abstract*—Acoustic Event Classification is an important but a rather challenging problem. In this report, we use a deep convolutional neural network for the given task. Our model architecture uses stacked convolution layers with pooling layers to extract features from a spectrogram-like representation of the given audio files.**

I. Introduction

The ability of deep convolutional neural networks (CNN) to learn discriminative spectro-temporal patterns makes them well suited to acoustic event classification. Audio data converted into spectrogram like representations can be feeded to CNNs to capture these spectro-temporal aspects and make sense out of them.

The project problem statement has been divided into 2 tasks i.e. Task 1 and Task 2, where Task 1 is, given an audio file corresponding to a single event, predict that event. For task 2, given an audio file containing a sequence of events occurring one after the other, predict that sequence of events. A sequence can contain at least 1 and at most 5 events. To avoid redundancy, we explain the approach used for Task-1 in detail and later highlight the improvisations made to complete Task-2.

II. Data Preparation

The given data comprises of 1761 audio files with each file being the sound recording of a particular acoustic event from a given set of 10 events including children-playing, dog-barking, drilling, etc. The Short-time Fourier Transform provides time-localized frequency information for situations in which frequency components of a signal vary over time like most of the real life acoustic events. We take the absolute value of the STFTs of the audio samples and perform zero-padding in the "time frame dimension" to ensure same number of frames for every spectrogram. The number of frames for every spectrogram is fixed to 401. Studies have shown that humans do not perceive frequencies on a linear scale. For example, we are better at detecting differences in lower frequencies than higher frequencies. In 1937, Stevens, Volkmann, and Newmann proposed a unit of pitch such that equal distances in pitch sounded equally distant to the listener. This is called the mel scale. We perform a transformation on our derived spectrograms and convert them into the mel-scale with number of mels = 513 to use for our model.

The next step is to split the data into training and validation sets. An important observation, here, was that the data contains multiple audio files from the same source for a particular class. If we make a random split, there are high chances of the training set ending up with examples from every source for a particular class. This can cause the model to over-fit to the source identity for every source. To avoid such overlaps between the training and validation data, we use a custom splitting function for splitting the dataset.

III. Convolutional Neural Network

A CNN is a multi-layered network including convolutional layers, max-pooling layers and some fully connected layers. The CNN model-architecture that we use is inspired from the given paper [1]. It comprises of 6 convolutional layers and 3 fully-connected layers. Following the 2 convolutional layers for large feature kernels, we use 1D feature kernels of sizes 3x1 and 1x5 in succession. This is done so that our model is able to extract patterns across both time and frequency. To introduce non-linearity and take care of vanishing gradients, we have used the Rectified Linear Activation Function (ReLU). Also, to avoid over-fitting we use l2 regularization wwith coefficient 0.001 for the convolution layer parameters and have added dropout layers with dropout probabilities = 0.7 and 0.3 for the fully-connected layers. The model architecture we use is tabulated as follows:

Layer	Ksize	Stride	Filters	Output Shape
Input	-	-	-	(401, 513, 1)
Conv1	(3, 7)	(1, 1)	32	(401, 513, 32)
Conv2	(3, 5)	(1, 1)	32	(401, 513, 32)
Pool1	(4, 3)	(4, 3)	-	(100, 171, 32)
Conv3	(3, 1)	(1, 1)	64	(100, 171, 64)
Conv4	(3, 1)	(1, 1)	64	(100, 171, 64)
Pool2	(4, 1)	(4, 1)	-	(25, 171, 64)
Conv5	(1, 5)	(1, 1)	128	(25, 171, 128)
Conv6	(1, 5)	(1, 1)	128	(25, 171, 128)
Pool3	(1, 3)	(1, 3)	-	(25, 57, 128)
FC1	-	-	256	(256,)
FC2	-	-	64	(64,)
FC3	-	-	10	(10,)

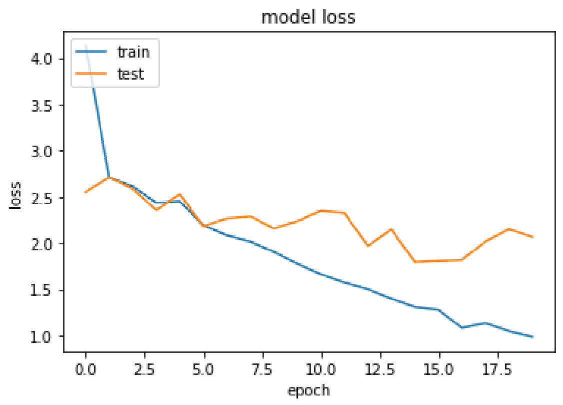

Fig. 1. Model Loss Task 1

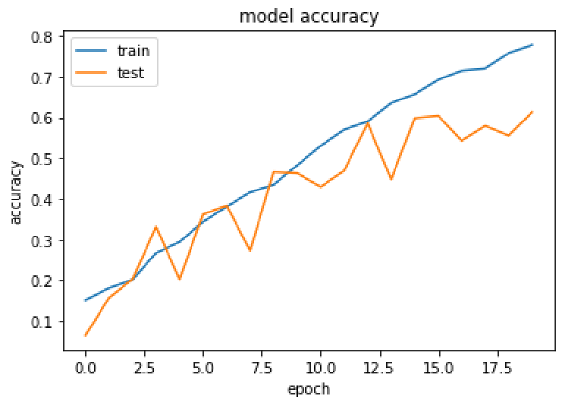

Fig. 2. Model Accuracy Task 1

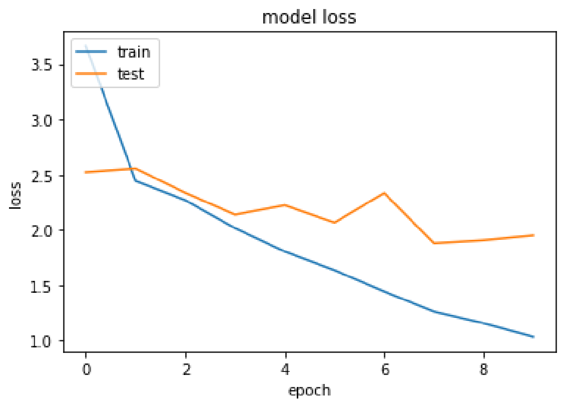

Fig. 3. Model Loss Task 2

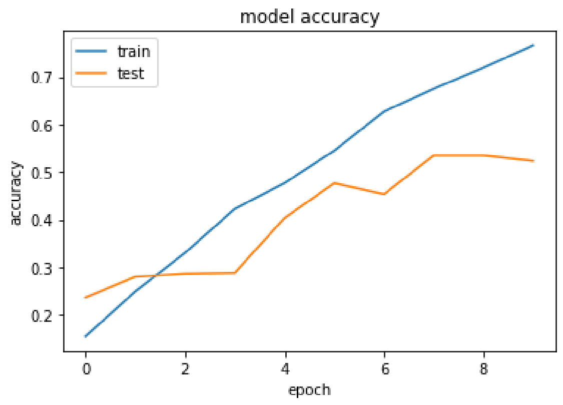

Fig. 4. Model Accuracy Task 2

IV. Results

Since we have a multi-class classification problem, the function chosen to calculate loss is categorical cross-entropy. The evaluation metric used is accuracy on the validation set and the optimization technique used to fit the model is RMSProp, which is a combination of AdaGrad and momentum-based gradient descent.

We observe from the plots that the model starts to over-fit the training set after a certain number of epochs as the training loss keeps on decreasing while the validation loss ceases to improve and keeps oscillating. Thus we use the model trained till 20 epochs as the model to make our final predictions. On training the model for 20 epochs, we get the above plots for model accuracy and model loss on the training and validation sets for task 1. The validation accuracy that we get is **0.6135** and the training accuracy is **0.7784**.

V. Modifications made for Task 2

For the second task, we will be provided with a single audio file which is made by concatenating more than one files from different classes and the task is to predict the correct ordered sequence of events. The spectrogram matrix of such an audio file is calculated and converted to the mel-scale, same as Task 1. Now, how do we make predictions? We use here a sliding window with a window size of 201 frames and a hop-size of 25 frames. For example, if we have a concatenated audio file whose spectrogram gives us 1000 frames in total, we slide our window of the size 201 over the whole spectrogram (taking jumps of 25 frames on each step) and make a prediction on every window. A sliding window of 10 frames and hop size 1 is again used to filter out predictions from these ordered set of predicted classes. From every window we extract out a specific class only if it is predicted at least half of the times and is not same as the last predicted class.

Note that since the number of frames in our input has changed to 201 frames from 401 frames (for Task 1), this task requires us to train another model where the input dimensions are (201, 513). We do not change the model architecture from Task 1. But, the original mel-scaled spectrograms are now padded to 402 frames and each spectrogram is then divided into two spectrograms with 201 frames each. This is done for every training example and the resulting 201-frame spectrograms are then used for training. The plots corresponding to Task-2 model training are attached above.

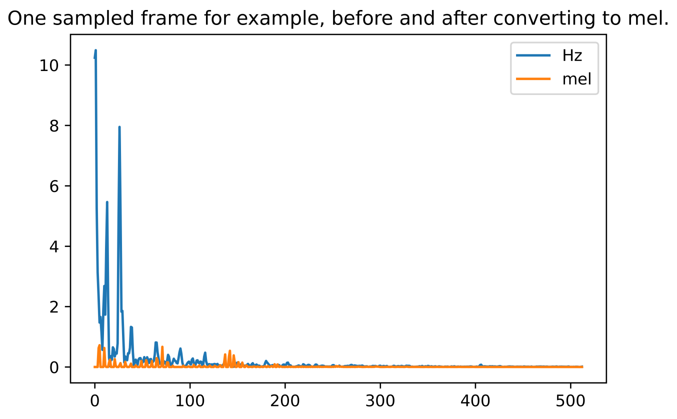

Fig. 5. Mel vs Hz scale

VI. Conclusion

In this report, we fit a deep-CNN model on mel-scale spectrogram of the input images. To visualize how mel-scale is different from the normal scale we plot the both together in Fig 5. To avoid over-fitting we used heavy dropouts layers and regularization. We also make sure to split our data in such a way which prohibits the validation set and the training set to contain "overlapping audio sources". This means that for a particular class, if the training set contains files from source A, the validation set will not contain any example for that class from the source A. Since the test data on which the performance of our model is to be judged can contain audio files from sources which are not present in the data provided to us, by making such a split, we are better able to evaluate the robustness of our model and how well it performs on previously unseen sources. The final models for both the tasks have been used to make predictions on the given test set for evaluation.

Acknowledgment

We thank Prof. Vipul Arora for the course he has taught us on Machine Learning in Signal Processing. Beyond theoretical understanding in this course, working on a project helped us in getting an insight as to how Machine Learning can have significant real world applications in the field of signal processing.

References

[1] Zhichao Zhang, Shugong Xu, Shan Cao and Shunqing Zhang "Deep Convolutional Neural Network with Mixup for Environmental Sound Classification", Shanghai Institute for Advanced Communication and Data Science, Shanghai University, Shanghai, 200444, China, 24 August ,2018