Assignment - 2 : Multi-Label Audio Classification

Yudhveer
Department of Electrical Engineering,
Indian Institute of Technology
Kanpur
yudhveer20@iitk.ac.in

1 Introduction

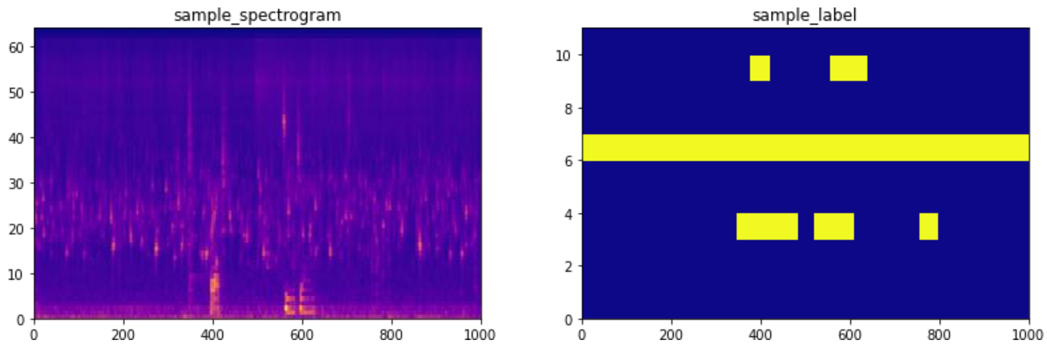
In recent years the classification of sound or audio recognition system has expanded its momentum in various fields like different animals voice recognition, automatic screams detection, the combination of video and audio for crime scene warning system, IoT based solution for urban noise detection in smart cities, sound classification with detection for medical and health care problems, classification of distinct musical instruments and many more. This shows the importance and scope of autonomous sound recognition systems in almost every aspect of not only humans but also other's living organisms like trees and animal's life [1]. This assignment is particularly based on detection of ten distinct audio events - Alarm bell ringing, Blender, Cat, Dishes, Dog, Electric shaver toothbrush, Frying, Running water, Speech, Vacuum cleaner on the basis of mel-spectrogram.

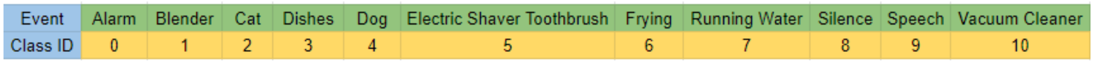
2 Literature Survey

Early SED approaches borrowed techniques from speech recognition and music information retrieval, and thus relied on traditional pattern-classification techniques such as Gaussian mixture models (GMMs) and hidden Markov models (HMMs). However, due to the specific techniques that enable the modelling of elementary units in speech or music, such as state-tying of phonemes or left-to-right topologies for modelling the temporal evolution of phonemes and musical notes, such models are much more useful in speech and music modelling. Because sound events do not generally consist of similar elementary units as speech, GMMs and HMMs are less relevant for SED. Furthermore, these methods are not intended to detect multiple classes at once. They needed specific extensions or setups to perform multilabel classification, such as binary classification for each sound event class, or preprocessing involving sound source separation. Modern pattern classification tools, particularly deep neural networks (DNNs), can, on the other hand, perform multilabel classification more easily: multiple output neurons that are active at the same time indicate the activity of multiple sound classes. This gives DNNs a significant advantage in solving the multilabel classification problem and has spurred them to the leading edge of the field. With DNNs, SED has seen significant improvements in both state-of-the-art performance and the complexity of problems tackled.[2]

3 Methods

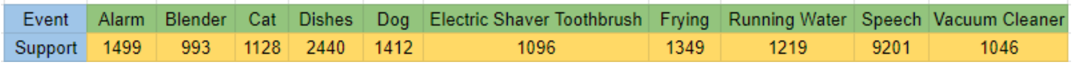
The primary step towards building an efficient classification model was to visualize and analyse the available data.





Event	Alarm	Blender	Cat	Dishes	Dog	Electric Shaver Toothbrush	Frying	Running Water	Silence	Speech	Vacuum Cleaner
Class ID	0	1	2	3	4	5	6	7	8	9	10

Clearly, there is overlap of sound events.Therefore, the classifier should be able to separately identify the class on the basis of their frequency magnitudes and variation in frequency with time.



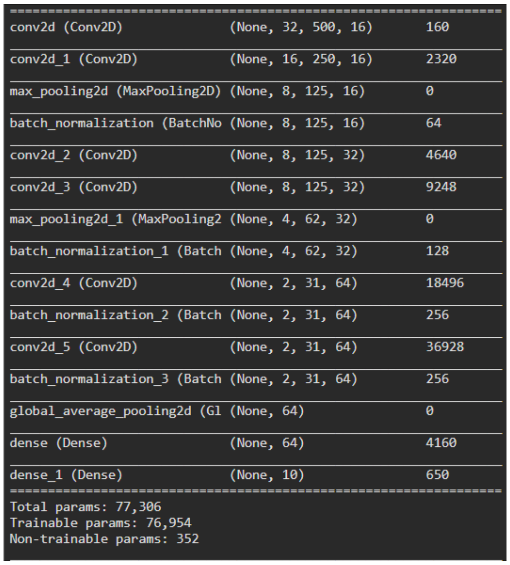
Event	Alarm	Blender	Cat	Dishes	Dog	Electric Shaver Toothbrush	Frying	Running Water	Speech	Vacuum Cleaner
Support	1499	993	1128	2440	1412	1096	1349	1219	9201	1046

Out of 10000 training samples class "speech" is present in 9201 spectrograms. Clearly there is heavy data imbalance.

3.1 Methods

3.1.1 Model-1

This is based on traditional convolutional neural network based approach. Since the mel-spectrograms are matrices of size 64x1000x1 they can be treated as images of dimension 64x1000 with depth of 1 channel. Moreover, each class forms a particular pattern on the spectrogram which are mainly shifted in time and space and shift invarience property of CNNs perfectly handles this. The following are the details of the CNN model which has been used for the sound event detection task.

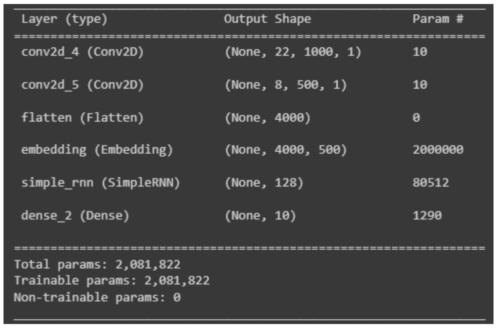


```
=================================================================
conv2d (Conv2D)              (None, 32, 500, 16)       160
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 16, 250, 16)       2320
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 8, 125, 16)        0
_________________________________________________________________
batch_normalization (BatchNo (None, 8, 125, 16)        64
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 8, 125, 32)        4640
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 125, 32)        9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 4, 62, 32)         0
_________________________________________________________________
batch_normalization_1 (Batch (None, 4, 62, 32)         128
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 2, 31, 64)         18496
_________________________________________________________________
batch_normalization_2 (Batch (None, 2, 31, 64)         256
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 2, 31, 64)         36928
_________________________________________________________________
batch_normalization_3 (Batch (None, 2, 31, 64)         256
_________________________________________________________________
global_average_pooling2d (Gl (None, 64)                0
_________________________________________________________________
dense (Dense)                (None, 64)                4160
_________________________________________________________________
dense_1 (Dense)              (None, 10)                650
=================================================================
Total params: 77,306
Trainable params: 76,954
Non-trainable params: 352
_________________________________________________________________
```

Figure 1: Model Architecture and Parameter Details

3.1.2 Method-2

The traditional CNN based performs well when the aspect ratio of the pattern to be recognised remains more or less constant. Since the audio or a sound can be stretched or compressed in time, CNN can't handle this much efficiently. To tackle this problem, a mixture of convolutional and recurrent neural networks is used. Convolutions of the spectrogram with the filters make the patterns statistically unique and these convolved outputs are then flattened. Embeddings are generated out of these flattened arrays and are further passed on to a simple RNN layer of 128 cells and the outputs are passed on to the fully connected dense layers to obtain the output mutlihot-vector.The following are the details of the RCNN model which has been used for the sound event detection task.

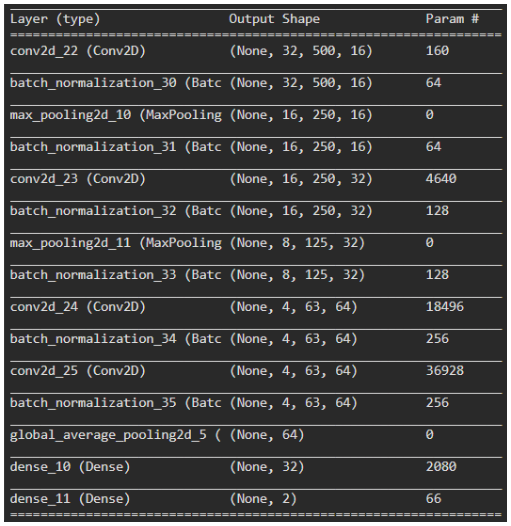


```
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_4 (Conv2D)           (None, 22, 1000, 1)       10

 conv2d_5 (Conv2D)           (None, 8, 500, 1)         10

 flatten (Flatten)           (None, 4000)              0

 embedding (Embedding)       (None, 4000, 500)         2000000

 simple_rnn (SimpleRNN)      (None, 128)               80512

 dense_2 (Dense)             (None, 10)                1290

=================================================================
Total params: 2,081,822
Trainable params: 2,081,822
Non-trainable params: 0
_________________________________________________________________
```

Figure 2: Model Architecture and Parameter Details

3.1.3 Method-3

The prior models were heavily biased towards the "speech" class. This was mainly due to severe data imbalance. To tackle this issue 2 different models are trained and their outputs were combined to get the final output mutlihot-vector.The first model is basically a cnn based binary classifier which separates the class "speech" from the other classes. This was trained on 1600 samples of which 800 contained the pattern corresponding to the class "speech" and the remaining samples did not contain that class. These samples were taken from the provided dataset. The second model was a CNN model with same architecture as the model-1, only the output size was different. This model classifies the remaining 9 classes.The figure below shows the architecture of binary classifying model.

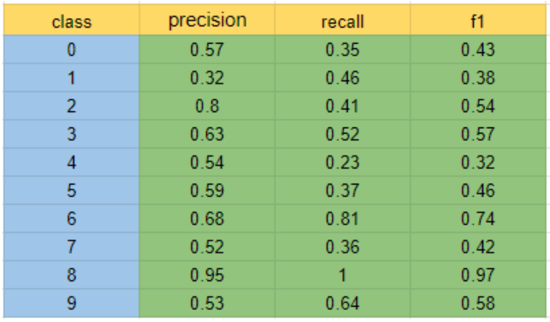


```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_22 (Conv2D)           (None, 32, 500, 16)       160
_________________________________________________________________
batch_normalization_30 (Batc (None, 32, 500, 16)       64
_________________________________________________________________
max_pooling2d_10 (MaxPooling (None, 16, 250, 16)       0
_________________________________________________________________
batch_normalization_31 (Batc (None, 16, 250, 16)       64
_________________________________________________________________
conv2d_23 (Conv2D)           (None, 16, 250, 32)       4640
_________________________________________________________________
batch_normalization_32 (Batc (None, 16, 250, 32)       128
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 8, 125, 32)        0
_________________________________________________________________
batch_normalization_33 (Batc (None, 8, 125, 32)        128
_________________________________________________________________
conv2d_24 (Conv2D)           (None, 4, 63, 64)         18496
_________________________________________________________________
batch_normalization_34 (Batc (None, 4, 63, 64)         256
_________________________________________________________________
conv2d_25 (Conv2D)           (None, 4, 63, 64)         36928
_________________________________________________________________
batch_normalization_35 (Batc (None, 4, 63, 64)         256
_________________________________________________________________
global_average_pooling2d_5 ( (None, 64)                0
_________________________________________________________________
dense_10 (Dense)             (None, 32)                2080
_________________________________________________________________
dense_11 (Dense)             (None, 2)                 66
=================================================================
```

Figure 3: Model Architecture and Parameter Details

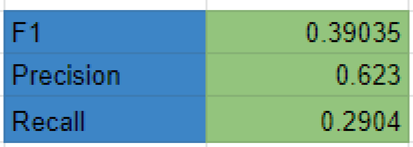
4 Results and Discussion

4.1 Test Results



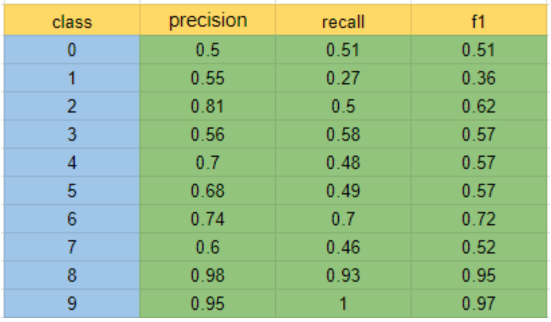
class	precision	recall	f1
0	0.57	0.35	0.43
1	0.32	0.46	0.38
2	0.8	0.41	0.54
3	0.63	0.52	0.57
4	0.54	0.23	0.32
5	0.59	0.37	0.46
6	0.68	0.81	0.74
7	0.52	0.36	0.42
8	0.95	1	0.97
9	0.53	0.64	0.58

Figure 4: Test Results for Model-1. Overall F1 score : 72.32



F1	0.39035
Precision	0.623
Recall	0.2904

Figure 5: Test Results for Model-2.



class	precision	recall	f1
0	0.5	0.51	0.51
1	0.55	0.27	0.36
2	0.81	0.5	0.62
3	0.56	0.58	0.57
4	0.7	0.48	0.57
5	0.68	0.49	0.57
6	0.74	0.7	0.72
7	0.6	0.46	0.52
8	0.98	0.93	0.95
9	0.95	1	0.97

Figure 6: Test Results for Model-3. Overall F1 : 75.66

Note : The results of CRNN model could be have been improved by making more dense architecture and training for more epochs which was not possible due to hardware constraints.

5 References

[1] ZohaibMushtaq, Shun-FengSu, Quoc-VietTran *Spectral images based environmental sound classification using CNN with meaningful data augmentation.*

[2] Annamaria Mesaros, Toni Heittola, Tuomas Virtanen, and Mark D. Plumbley *Sound Event Detection: A Tutorial*