AUDIO EVENT DETECTION

Introduction

Our task focussed on identifying the rare sound events from artificially created mixture audios which contained three target categories viz. baby cry, gun shot, and glass breaking using Artificial Intelligence. The data we have worked on contains mixtures of everyday audio and sound events of interest at different event-to-background ratio, providing a larger amount of training conditions than would be available in real recordings.

Requirements

- Numpy
- Sklearn
- Pandas
- Keras
- Matplotlib
- Librosa
- Sounddevice

Audio Dataset

TUT Rare Sound Events 2017 consisted of isolated sound events for each target class and recordings of everyday acoustic scenes to serve as background. The data we worked on to train our model contained background noise along with the rare sound events. The testing set also consisted of similar mixtures.

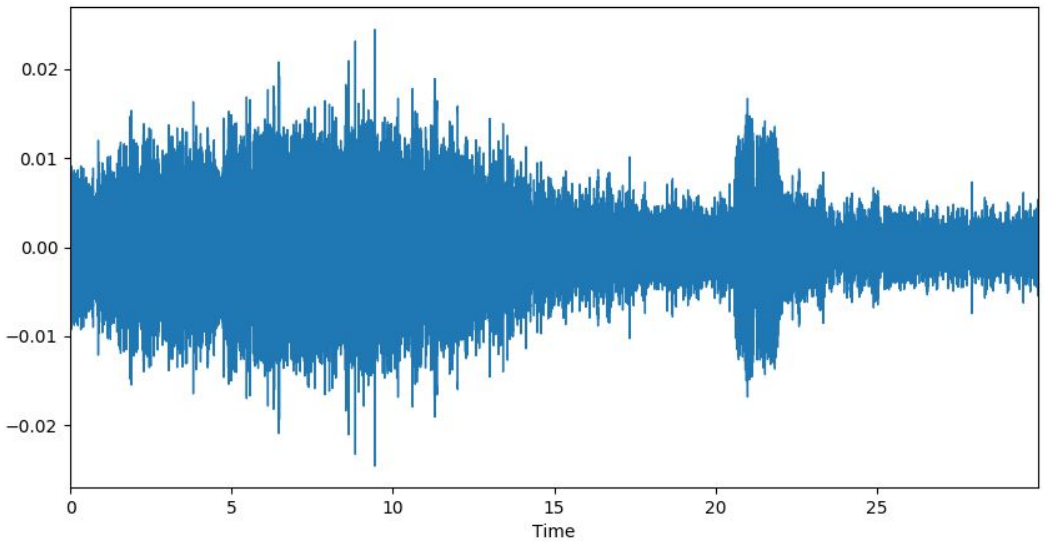
The target sound event categories were:

- Baby crying
- Glass breaking
- Gunshot

Our training data and testing data consisted of 1,491 and 1,496 audio recordings (30 seconds each) respectively along with background noise, making up 6 gigabytes of memory from each of the two.

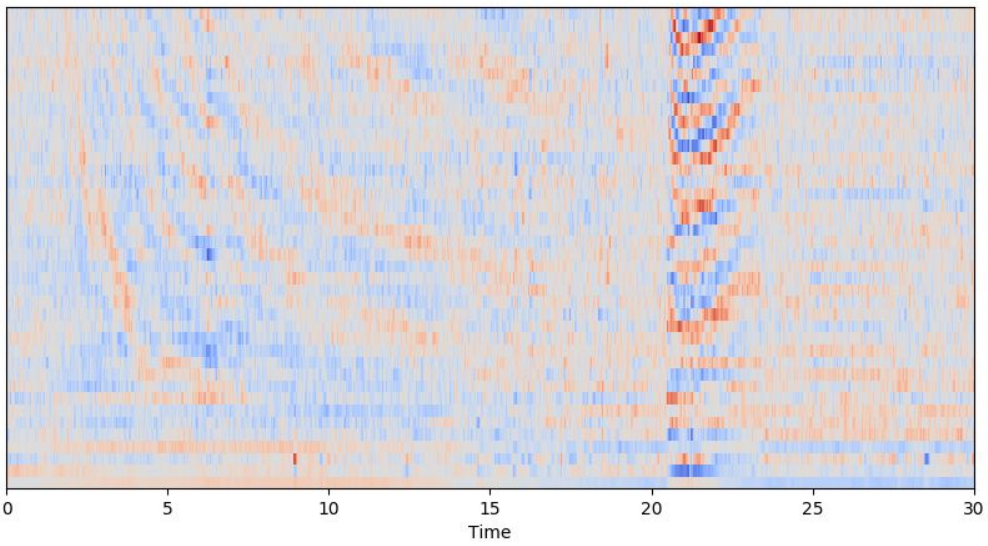
Data Processing and Feature Extraction

The provided data contained three **csv** files for each of training and testing from which we extracted the names and labels. The ones having **NaN** as labels were converted to **None** using **Pandas** library. Then the audio data was converted into numpy arrays using **librosa** which looked something like the picture below:



(Baby cry audio)

After the audio processing, our task was to identify the useful audio from the prevalent background noise. We used **Mel Frequency Cepstral Coefficients (MFCC)** which took the input and represented it on a scale which was essentially logarithmic so as to closely resemble the human auditory system. The MFCC representation is shown below:



(Baby cry audio)

After successfully extracting the feature for each audio of training and testing data and thereby converting the audio into an image as shown above, we stored it in a pickled file named **'featurised.csv'**.

Model Design

We used a hybrid model consisting of Recurrent neural network (RNN) and 2D-Convolutional neural network (CNN).

Model Architecture

Our model consisted of four 2D CNN followed by Max pooling 2D and Dropout, further followed by Global average pooling and a simple RNN, which is shown in the image below:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 39, 1291, 16)      80
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 19, 645, 16)       0
_________________________________________________________________
dropout_1 (Dropout)          (None, 19, 645, 16)       0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 18, 644, 32)       2080
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 9, 322, 32)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 9, 322, 32)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 321, 64)        8256
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 4, 160, 64)        0
_________________________________________________________________
dropout_3 (Dropout)          (None, 4, 160, 64)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 4, 160, 128)       8320
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 4, 160, 128)       0
_________________________________________________________________
dropout_4 (Dropout)          (None, 4, 160, 128)       0
_________________________________________________________________
global_average_pooling2d_1 ( (None, 128)               0
_________________________________________________________________
reshape_1 (Reshape)          (None, 128, 1)            0
_________________________________________________________________
simple_rnn_1 (SimpleRNN)     (None, 20)                440
_________________________________________________________________
dense_1 (Dense)              (None, 4)                 84
=================================================================
Total params: 19,260
Trainable params: 19,260
Non-trainable params: 0
```

(Model architecture)

Training and Testing

The model was trained using 6 GB of data consisting of 1491 recordings of 30 seconds each. It was run for 120 epochs.

The metrics were as follows:

- Training Accuracy: 79.007%
- training f1 score: 79.476%

The model was tested using 6 GB of data consisting of 1496 recordings of 30 seconds each. The metrics were as follows:

- Testing Accuracy: 71.056%
- testing f1 score: 74.042%

	0	1	2	3
0	205	14	7	21
1	2	215	19	14
2	1	84	128	37
3	14	94	126	515

(Testing Confusion Matrix)

References

- http://dcase.community/challenge2017/task-rare-sound-event-detection
- https://medium.com/@mikesmales/sound-classification-using-deep-learning-8bc2aa1990b7
- http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/