Audio Event Detection

Team name : Vigilantes

Team members : Aashi Pathak (180008)

Neha Kumari (180467)

Course title : Machine Learning for Signal Processing (EE698V)

Introduction

It is an audio classification problem, technically known as "Audio Event Detection".

We are given two tasks

1. Detection of Single Event
2. Detection of Sequence of Events

In task 1 We are given an audio file corresponding to a single event, we have to find out that event. Task 1 example: street_music.

In task 2, given an audio file containing a sequence of events occurring one after the other, we have to find out that sequence of events. A sequence can contain at least 1 and at most 5 events. For example street_music, dog_bark, engine_idling.

Task 1

Audio dataset creation

We are given an audio file that consists of isolated sound events for each target class and recordings of everyday acoustic sound events.
The target sound event categories are:
dog_bark, gun_shot, engine_idling, siren, jackhammer, drilling, children_playing, street_music, air_conditioner, car_horn

For the input (X) purpose we used the given feature extraction function “wav2feat” to convert our audio wav files to numpy arrays and append them in the list ‘X’. As all the numpy arrays were of different column size, we did zero padding to make it of size (513,401) as 401 was the maximum number of columns in an array. Finally the shape of X was (1761, (513, 401)).

For the output (y) purpose we used corresponding labels from the given “labels.csv” file and changed the labels to corresponding index (0-9). Finally we converted our “y” to “one hot vector” and the shape of y was (1761, (1,10)).

Algorithm

We used a Sequential model for training our dataset. This has a neural network containing 1 LSTM layer of 256 hidden nodes followed by a flatten layer, 1 dense layer of 128 nodes followed by a dropout layer and another dense layer on nodes equal to the total number of classes(10) and 30% dropout is trained for 80 epochs. We also used “callbacks” to prevent “overfitting”. Classification decision is based on the network output layer which is of “softmax” type. The system includes evaluation of results using accuracy as metric.

For our model

learning_rate = 0.001

batch_size = 64

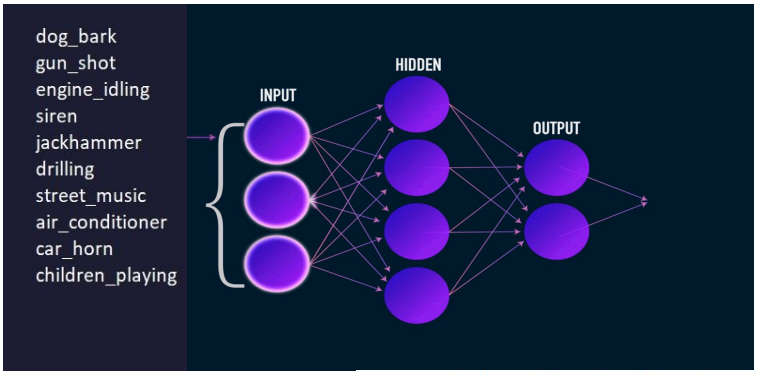
n_epochs = 80

dropout = 0.3

Validation set accuracy = 85.88%

Testing

For testing our model if we take an input numpy array and apply the model.predict function. We get a list of indices(0-9) corresponding to the array. We then can convert these indices to the corresponding labels.
A “feat” file having sample input datasets was given to us. We applied testing on it using the “evals” function and we got a score of 1.0.



sequential model with hidden layer

Task 2

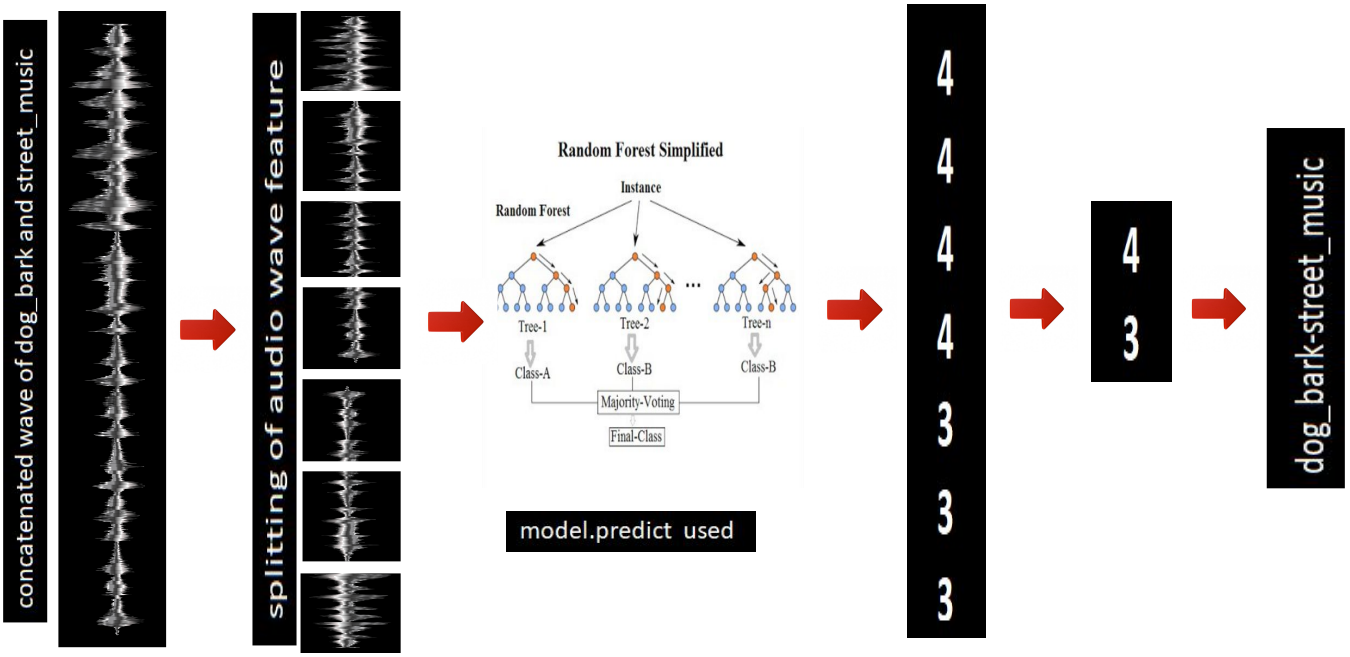
Audio dataset creation

We first created random concatenated datasets from audio files and did the extraction using “wavs2feat”. We were trying to implement “silence-detection” to do segmentation of our input but in our given dataset, there was no silent part.
So, we thought to train each small segment of an array corresponding to a single wave-file, so that when we give concatenated wave files array to our model, It will identify the label corresponding to that segment.
For the input (X) purpose we used the given feature extraction function “wav2feat” to convert our audio wav files to numpy arrays. Then we divided the given 2D numpy array into small segments with column size 20 and appended it into a list X.
The output of each segment of a numpy array, which was extracted from “wav2feat”, corresponds to the same class as the array does(for example, If an array with shape (513, 312) has label “dog-bark” then all the segments formed by this array with shape (513,20) will have the same label “dog-bark”).

Algorithm

We used Random Forest model to train our dataset. Although better performance applications of neural networks are reported in many papers for sound event detection, we presented a new random forest ideas for this. We used the random forest as classifier and trained with multiclass/multilabel strategy, we achieved better performance compared to the sequential model.
For our model

n_estimator = 30 Validation set accuracy = 94.73%



Testing

For testing our model if we take an input numpy array, first we do the segmentation part. By the model.predict function, we get a list of indices(0-9) corresponding to each segment, and apply a function on it which results in a final list of indices. We then can convert these indices to the corresponding labels.
A “feat” file having sample input datasets was given to us. We applied testing on it using the “evals” function and we got a score of 0.94167.