JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Urban Sound Classification using Deep Learning

[1]Ruturaj Ghodke, [2]Prajan Shukla, [3]Vedita Kharabe
Department of Computer Engineering,
AISSMS Institute of Information Technology, Pune, India

Abstract : In life, sound plays a very vital role. Just like sight, humans perceive sound and gain crucial information from it. Even without sight, sounds help us to understand what is happening around us. Today, many security systems like cameras or trackers are based on either sight or motion. Even for the visually disabled, visual aid is provided, and hence the importance of sound here is neglected. Environmental sounds, when analyzed and identified, give us an idea about the presence of things around us. Urban environmental sounds are so unstructured in nature that it is not easy for an intelligent system to classify them, but they contain patterns of temporal spectrum that assist in analysis and classification. Using these techniques, a machine can learn to identify sounds and accurately identify them. Hence, we can also use the trained system for surveillance in cameras where the sight is out of range, to aid people with difficulty in hearing and many other applications. The aim of this research is to identify and classify a given environmental sound by using a CNN (Convolutional Neural Network) for classification. We feed an environmental sound audio file to the system after converting it to a spectrogram (visual representation) of the sound and predict the output of the type of sound fed in. Since spectrograms are nothing but images, we use CNN as it works best with image classification and computer vision.

IndexTerms **- Deep Learning, Audio Processing, Spectrogram, CNN.**

I. INTRODUCTION

Today, there are many applications that use sight or visual representations as a tool to perceive the environment and act upon them as per the system it is made for. Surveillance systems [1] for example consist mainly of cameras for capturing live feed of the surroundings, but sometimes it is not possible to get knowledge of an event just by sight. Perceiving and understanding sound not only gives us a clue of the event that's happening in the environment without disturbing our line of sight but also takes remedial actions according to the event. A person trying to cross a road will look for vehicles on one side of the road but can perceive sounds coming from all sides. If accidentally a vehicle comes from the other side, the sound of its horn can alert the person quickly even though he was not looking at that side. This shows how important sound is and how crucial knowledge we can get from it. If a machine can have the ability to analyze and classify sounds [2] [3], then we can apply its results to various applications. A person with difficulty hearing can use the system which will accept sound data and can output what sound was fed to it [4]. Surveillance systems can add this functionality to their architecture which will help them to predict events in surroundings where their line of sight cannot reach. This functionality can also be added in home automation systems [5]. Research has already begun to use machine learning to analyze environmental sounds to use for robot navigation [6].

The proposed algorithm analyses and classifies urban environmental sounds using CNN (Convolutional Neural Network) and results the predicted sound to the output. There is not much advancement done in machine learning for sound. Some algorithms were used other than CNN, but their results were not promising enough. The sounds are not directly fed to the algorithm but are preprocessed and are converted into spectrograms (a visual representation of the sound i.e., a frequency vs time graph of the sound) and that image is fed to the CNN algorithm [7] [8]. The output is a predictive class of the sound that was fed to the input. The dataset used for training the CNN is Urbansound8k [9] containing over 8,000 environmental sounds (<=4sec) with 10 labeled classes. The rest of the paper is organized in the following sequence - Section II mentions the Related Work followed by Section III which discusses the Proposed Methodology. Further Section IV highlights the Results and Analysis and finally, we have Section V with the Conclusion, along with Acknowledgement and References.

II. RELATED WORK

J. Singh et.al [10], proposed background sounds from a video or sound clip with a foreground human speech to be identified using a type of CNN model called VGG network. A newly made dataset called YBSS-200 is used which provides them with a maximum accuracy of 80.2%. The VGG CNN model is known to have a very good accuracy rate but the new dataset they used had many noisy data which hindered accuracy, but augmenting this new dataset to an already clean one like Urbansound8k will surely show improvement in performance.

Zhang et.al [11], used CNNs and dilated filters along with a LeakyReLU activation function methodology to classify environmental sounds. The datasets used are Urbansound8k, ESC50, and CICESE. This model with LeakyReLU proved to be close to 2% more accurate than CNNs with only a ReLU function but at the cost of higher complexity power and bigger storage. 2% more accuracy

does not hold up for a higher complex model according to us, although the paper proposes to compress the model in further research.

F. Vesperini et.al [12], proposed how the environmental sound can be detected and classified using Capsule Neural Networks (CapsNet) rather than the primary CNN. The datasets used were TUT Sound Events 2016 & 2017 and TUT Rare Sound Events 2017. The maximum average accuracy resulted in 78.0% during evaluation. The model can classify overlapped images (simultaneous sounds) and is particularly effective with small datasets but struggles with large ones. Since working on CapsNet has just started recently, more research should be done on its robustness to overlapping signals.

K. Zhang et.al [13], proposed how environmental sounds are identified and classified using Multi-scale Time-Frequency Convolutional Recurrent Neural Network (MTF-CRNN). The datasets used are TUT Rare Sound Events 2017 & TAU Spatial Sound Events 2019. The model produced an accuracy of 89.2%. This modified model is adopted to improve the performance of detection of events in sound and reduce the neural network parameter counts resulting in low computing power. But as parameter counts are reduced the same relationship does not hold true for running time.

Aswathy Madhu [14] presented a CNN for urban sound classification with lesser parameters. The proposed architecture reduces parameters by 24%. The dataset used is Urbansound8k and a mean accuracy of 90.85% is obtained. The architecture performs with lesser computational complexity without compromising classification accuracy. Further tweaking of the CNN layers can bring about interesting results.

N. Davis et.al [15], proposed a system to classify environmental sounds using deep CNNs and data augmentation. Here, confusion matrices are used to show the accuracy of the model. The testing accuracy obtained for the dataset before augmentation is 79% and 87% for the time stretch augmented dataset. Hence, data augmentation positively affected only half of the classes whereas for others accuracy improved by more than 5%. So, the detection accuracy is improved by using class-conditional data augmentation.

Q. Kong et.al [16], investigated detection of environmental sounds using a CNN-Transformer which performs similarly to a CRNN, and an optimal threshold optimization method is also used to overcome previous threshold setting problems. Datasets used are DCASE (2017/2018/2019) Task4. This method works best on weakly labeled data sets containing only audio tags for each sound clip. A larger data set should be used to extend the system and enhance its performance.

J. Sang et.al [17], used convolutional recurrent neural networks with raw waveforms enhance the accuracy of environmental sound classification also providing the efficacy of its structure considering the parameter count. Urbansound8k dataset is used, and the accuracy achieved was 79.06%. With a smaller number of parameters, they achieved the results indicating that CRNN results are better in computation than deep CNN compositions. Trying to make the network deeper here resulted in overfitting and dropped accuracy to 68.07%. To train on such a deep model, we feel a larger dataset needs to be sufficient.

The above papers achieved a slightly more accuracy rate but at the cost of higher processing power and time needed. We can compromise on such improvement but will focus on optimizing learning time and processing power. Also, some papers use data sets that are not correctly or weakly labeled, thus leading to a decrease in accuracy rate. We try to use a strictly labeled data set to achieve the maximum possible accuracy.

III. PROPOSED METHODOLOGY

Figure 1 depicts the proposed system. The system carries out steps right from giving an input audio file to its prediction as to what sound it is. The system has two main phases: preprocessing phase and the CNN (Convolutional Neural Network) phase. The preprocessing phase converts the audio files (.wav format) into their respective spectrograms (.jpg image) which are a visual representation (a frequency vs time representation) of that sound. For training the CNN we use the UrbanSound8k dataset. This dataset contains 8732 environmental labeled sounds containing 10 classes like: *street music, siren, jackhammer, gunshot, engine idling, drilling, dog barking, children playing, car horns, and air conditioners.*

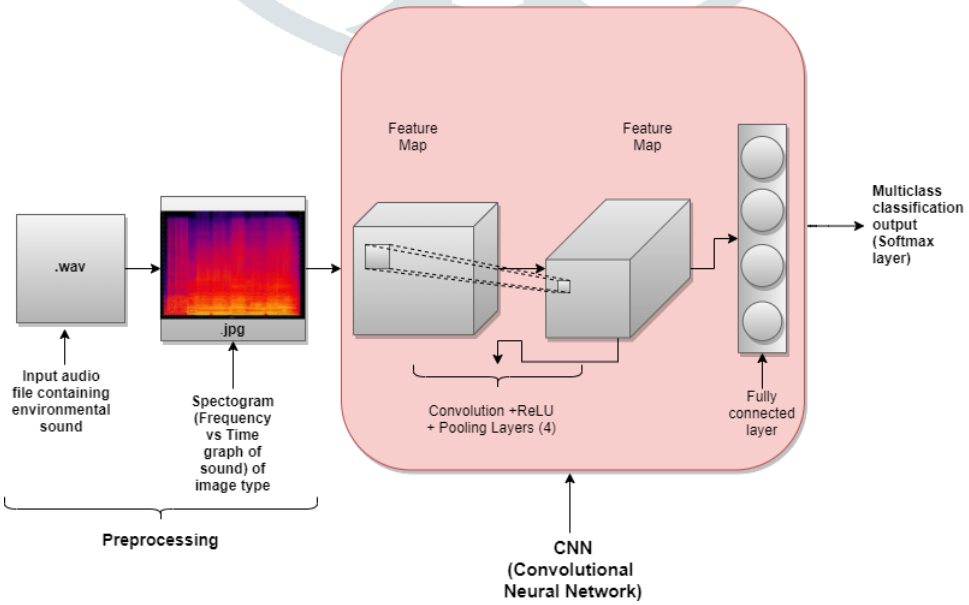

Figure 1 System Design

3.1 Input:

The system is first trained by feeding the audio files of the UrbanSound8k dataset and these files go through the same process as per the following steps of preprocessing and CNN. Only this time the CNN model is trained and tested. The system accepts an audio file containing environmental sound in .wav format. The sound file should be no more than 5 seconds for efficient processing. It will also be an added advantage if the audio file contains the urban sounds with minimum auditory 'noise'. This audio file is then fed to the system to be preprocessed. Figure 2.1 and Figure 2.2 display the amplitude vs time waveform signal of input sounds which are to be fed to the system to be preprocessed.

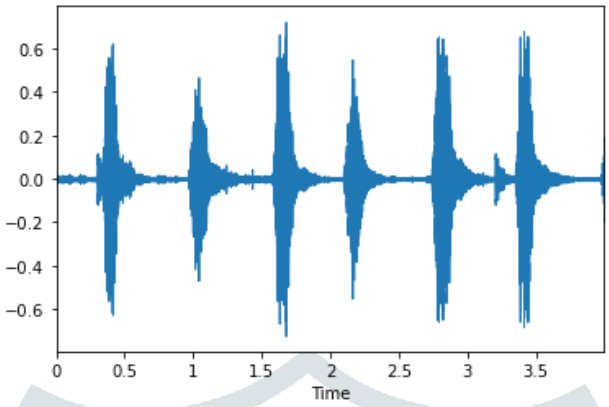

Figure 2.1 Input Sound Waveform Signal Example 1

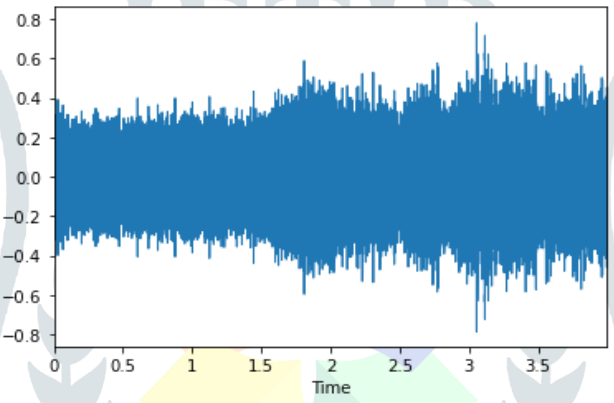

Figure 2.2 Input Sound Waveform Signal Example 2

3.2 Preprocessing:

The preprocessing steps involve converting the audio files into images. These images are a visual representation of the input sound itself. They are known as spectrograms or basically a graph containing frequency vs time data of that audio. For these conversions, we use a python library called Librosa which is mainly used for audio and music analysis. The library contains functions which accept a .wav audio file and outputs. Converting an audio file to image:

1. Pass audio .wav file to *librosa.feature.melspectrogram()* function.
2. The function returns a numpy n-dimensional array.
3. Use matplotlib functions to plot this array.

These n-dimensional arrays are pixeled representations of the image and in turn the audio itself, and thus this array is stored for further processing to CNN as a machine itself perceives the image as a n-dimensional array of pixel values. The arrays of pixels of these images are in the dimensions of *h * w * d (h = Height, w = Width, d = Dimension).* Figure 3.1 and Figure 3.2 show the converted or preprocessed spectrograms of Figure 2.1 and Figure 2.2 respectively, ready to be fed to the system.

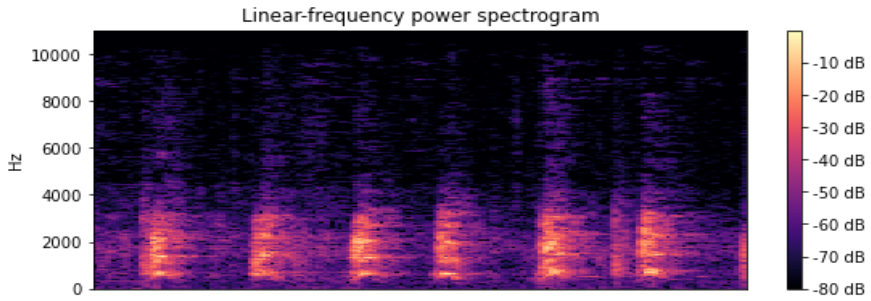

Figure 3.1 Spectrogram of Sound from Figure 2.1

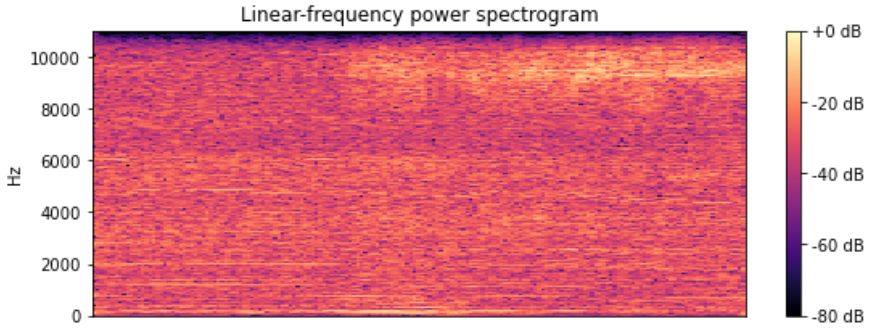

Figure 3.2 Spectrogram of Sound from Figure 2.2

3.3 Convolutional Neural Network:

A convolutional neural network is a type of deep neural network and is used majorly to analyze images. It is a deep neural network as it consists of multiple layers of nodes which compute the task of assigning weights or biases to various aspects of the image. Figure 4 shows a sample image visualized as a matrix of numeric values.

3	1	1	2	8	4
1	0	7	3	2	6
2	3	5	1	1	3
1	4	1	2	6	5
3	2	1	3	7	2
9	2	6	2	5	1

Figure 4 Input Image Matrix

\

1) Convolution Layer: The first layer of the CNN is used to extract features of the image. We pass the image to CNN. We also pass a Kernel or a Filter along with it. The Filter is a relatively smaller block with a specific pattern. We do matrix multiplication between the Filter and the portion of the image with which it fits on and then it slides to the next portion. The value is stored on another array called 'feature maps' as depending upon the filter pattern, edges and other features are detected here. Equation 1 shows the convolution operation that is performed in the convolutional layer of the CNN. Figure 5 shows convolutional layer matrix multiplication.

Input image dimensions (or feature maps) = *(m, m)*
Kernel/Filter dimensions = *(k, k)*
Output dimensions of new feature maps = *((m-k+1), (m-k+1))*

$$J_{(x,y)} = K \times I = \sum_{n,m} K_{(n,m)} \times I_{(x-n,\, y-m)} \quad (1)$$

where,
J is the convolved matrix,
K is the kernel,
I is the input image matrix,
x, y and *m, n* being indices of input matrix and kernel matrix.

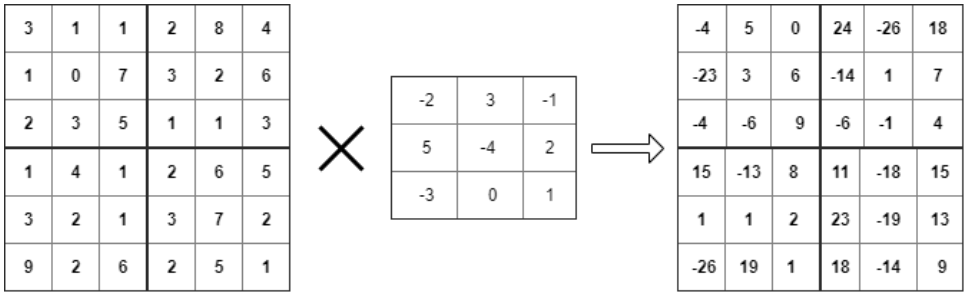

Figure 5 Convolutional Layer

2) ReLU Function: We use an activation function after the convolution layer to add non-linearity to our data. We use ReLU or Rectified Linear Unit to eliminate any negative values in our results. A particular neuron or node in the CNN is fired when we get a value above 0 when we use the ReLu function. It does not produce output if the value received by ReLU is below 0. Values after 0,

there exists a linear relationship between the dependent and the independent variable. Equation 2 defines the ReLU function. Figure 6 shows the graph of ReLU function.

$$f(x) = \max(0, x) \quad (2)$$

where *x* is the value parameter for the function.

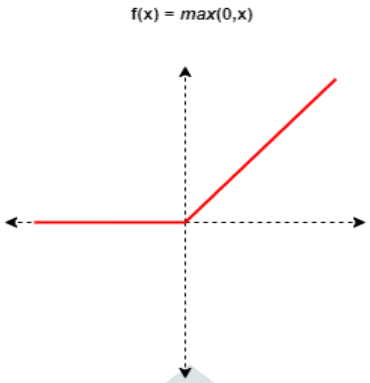

Figure 6 ReLU Function

3) Pooling Layer: We use a Pooling layer after every Convolution layer which reduces the magnitude of feature map generated by the convolution layer. This decreases computational power used for processing the data by reducing the dimensions of it whilst preserving important information. We use a MaxPool2D layer of size (2,2) for the Pooling. Max Pooling selects only the highest value (dominant features) from the portion the pool covers on the feature map. Figure 7 shows transformations of feature map after pooling. The dimensions after pooling are:

$$D_p = \frac{(n - f_s + 1)}{s_t} \times \frac{(w - f_s + 1)}{s_t \times c} \quad (3)$$

given,
h = feature map height,
w = feature map width,
c = count of channels in feature map,
f_s = filter size,
s_t = length of stride.

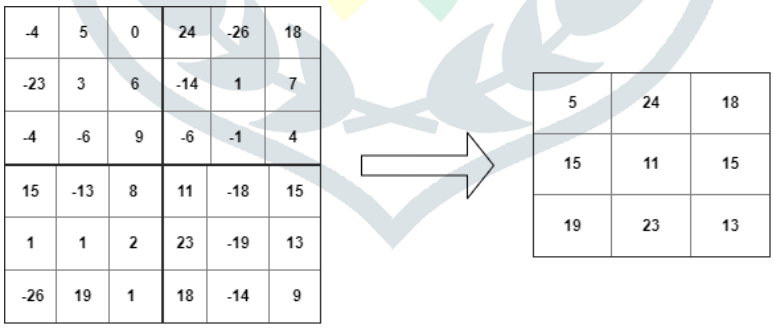

Figure 7 Pooling Layer

4) Fully Connected Layer: In this final layer called the Fully Connected layer, actual classification of sounds is carried out. The output of the Pooling layer, which is a small feature map, is flattened out as a column vector. Once flattened, we apply the softmax function, which in turn assigns weights or probabilities to the classes. The higher the probability, the higher chance the sound belonged to that class. Equation 4 defines the softmax function. Figure 8 shows matrix flattening of feature maps in fully connected layer.

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad (4)$$

where,
σ = softmax value
z = input vector
e^{Zi} = exponential function on input
K = count of classes to predict
e^{Zj} = exponential function on output

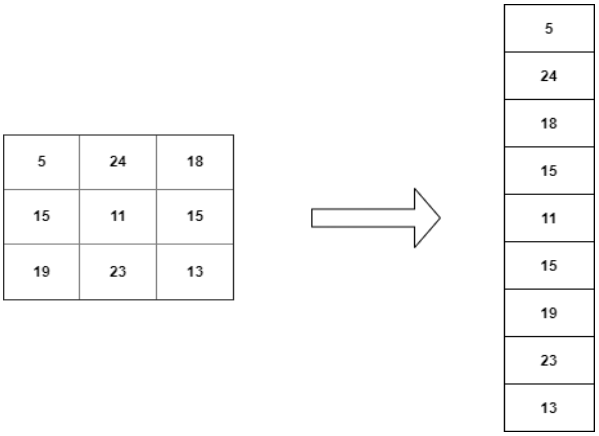

Figure 8 Fully Connected Layer

IV. Results and Analysis

Based on the input audio file, the class of the audio or sound is predicted by the model. The softmax function gives a percentage distribution for the classes for which the sound belongs to: *(for e.g., Car Horn, Jackhammer, Children Playing, etc.)*. We use 4 convolution layers with MaxPool2D layer. We also use a Dropout of 0.1 (10% of the nodes will shut off randomly). Introducing Dropout here we tend to prevent overfitting. Dropout regularizes the nodes and forces them to learn robust features that are useful in conjunction. The model is trained for 150 epochs. For getting better prediction results, we also use the Adam optimizer to obtain global minima. The output shows the input audio spectrogram image along with the class of the sound predicted. The model gave a training accuracy of 98.73% and an average test accuracy of 89.38%. Training and testing the model without Dropout gave us a test accuracy of 87.13%. Hence, it is observed that Dropout did regularize the model and bumped up accuracy by at least 2%. Any other Dropout values greater than 0.1, the model's accuracy drops. While predicting the classes of sounds, we first encode the classes into numeric identifiers so the model will be able to classify them. Table 1 shows the encoded labels along with the label names. This makes Figure 9 easier to understand.

Table 1 Sound Classes

Numeric Identifier	Sound Class
0	air_conditioner
1	car_horn
2	children_playing
3	dog_bark
4	drilling
5	engine_idling
6	gun_shot
7	jackhammer
8	siren
9	street_music

Figure 9 shows the confusion matrix of the predicted test results. The labels of numeric identifiers are nothing, but the corresponding label names as given above. The dataset comes with presorted 10 folds. We perform 10-fold cross validation as suggested by the dataset documentation. We train the model on all 10 folds and the result is the average of all the fold results. We avoided reshuffling the data in the dataset as it may lead to incorrectly placed samples in both the training set and test set which will lead to incorrect results.

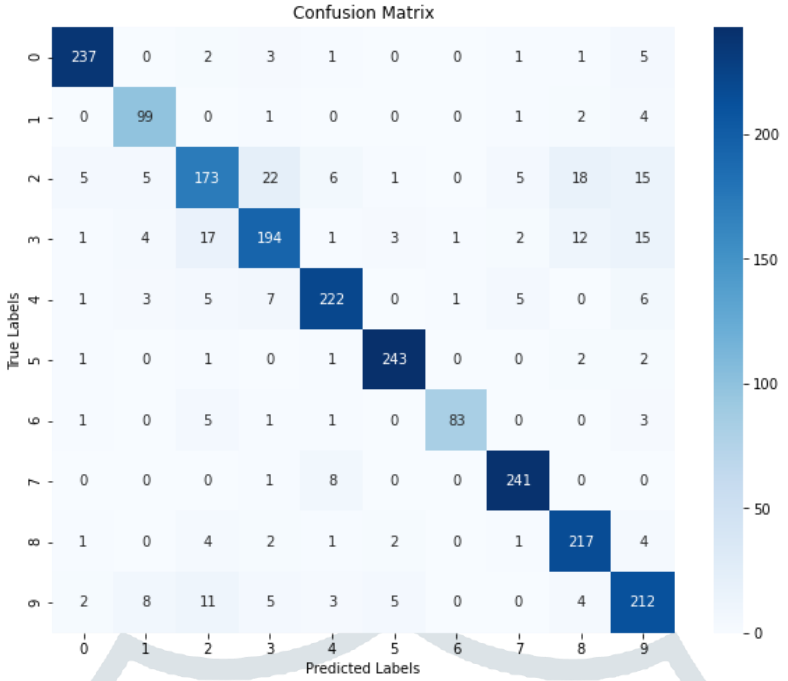

Figure 9 Confusion Matrix

We also compared our model to a pre-existing CNN model known as VGG16 [18]. It gave an accuracy of 92.7% on ImageNet (a dataset with over 14 million images belonging to 1000 classes). Since we could not find a pre trained VGG16 as ImageNet does not include spectrograms of sounds, we had to train VGG16 from scratch. VGG16 is a very deep CNN consisting of 13 convolutional layers and 5 pooling layers. We trained the VGG using the same dataset we used (Urbansound8k) and observed the following results. VGG16 gave a training accuracy of 96.26% and an average test accuracy of 86.73%. The values are finely close as compared to our model: (Train- 98.73%, Test- 89.38%). We think the results of VGG were slightly lower than ours because it may have unneeded layers for our dataset as compared to ImageNet which it was trained on initially and had 14 million images. Hence for such a huge dataset, 13 layers may have worked. If further our dataset is augmented, the results of VGG16 could improve in our testing. Figure 10 shows spectrograms with labels predicted by our model.

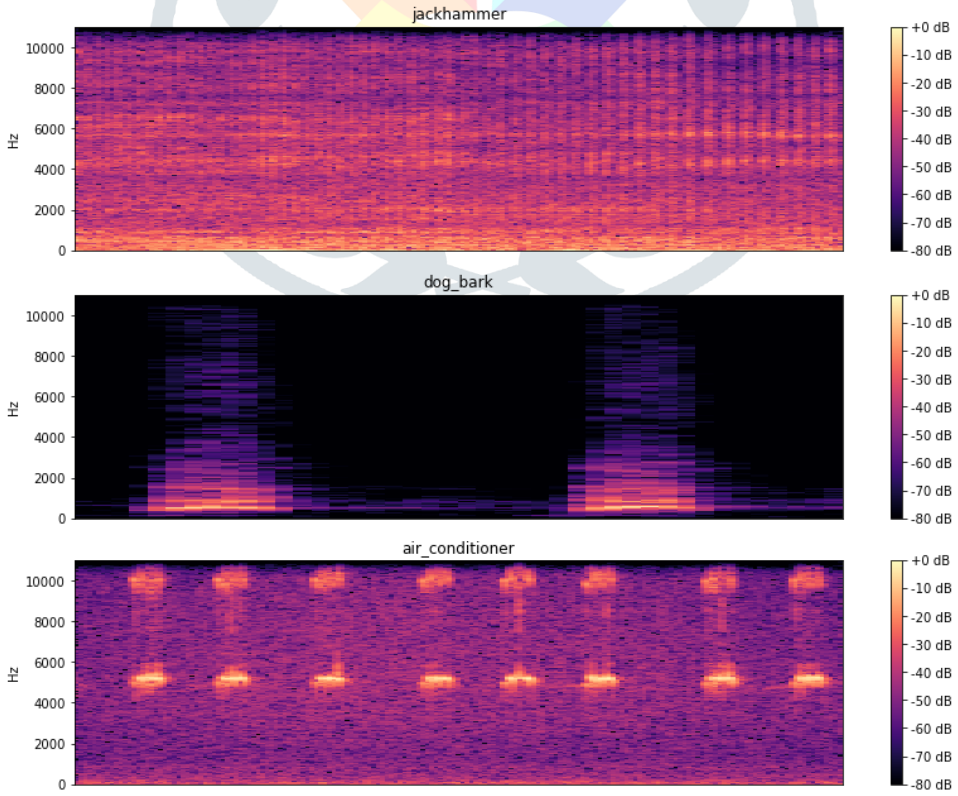

Figure 10 Sounds with their predicted labels

V. CONCLUSION

Identification and classification of environmental sounds is difficult as they are not structured in nature, but they contain strong spectro-temporal patterns which makes classification possible. We use CNN as it works best on classifying images and hence, we

classify sounds with a CNN by converting them into images of spectrograms. Hence, we trained and tested a CNN model for identifying and classifying urban environmental sounds using the Urbansound8k dataset. We achieved an average testing accuracy of 89.38% with our current model. Introducing Dropout regularized the model's performance and gave our results a boost of nearly 2%. Future enhancements like data augmentation will be made to improve the performance of the model.

VI. Acknowledgment

The authors would like to thank Dr. S. N. Zaware, HOD of Computer Department, AISSMS Institute of Information Technology for having informative discussions and providing many suggestions about the methods shown in the paper.

References

[1] S. U. Hassan, M. Zeeshan Khan, M. U. Ghani Khan, and S. Saleem, "Robust sound classification for surveillance using time frequency audio features," in 2019 International Conference on Communication Technologies (ComTech), 2019, pp. 13–18.
[2] K. Jaiswal and D. Kalpeshbhai Patel, "Sound classification using convolutional neural networks," in 2018 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), 2018, pp. 81–84.
[3] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," IEEE Signal Processing Letters, vol. 24, no. 3, pp. 279–283, 2017.
[4] I. Lezhenin, N. Bogach, and E. Pyshkin, "Urban sound classification using long short-term memory neural network," in 2019 Federated Conference on Computer Science and Information Systems (FedCSIS), 2019, pp. 57–60.
[5] M. J. Baucas and P. Spachos, "A scalable iot-fog framework for urban sound sensing," Computer Communications, vol. 153, pp. 302–310, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366419312502
[6] E. Baum, M. Harper, R. Alicea, and C. Ordonez, "Sound identification for fire-fighting mobile robots," in 2018 Second IEEE International Conference on Robotic Computing (IRC), 2018, pp. 79–86.
[7] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP), 2015, pp. 1–6.
[8] N. Davis and K. Suresh, "Environmental sound classification using deep convolutional neural networks and data augmentation," in 2018 IEEE Recent Advances in Intelligent Computational Systems (RAICS), 2018, pp. 41–45.
[9] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in 22nd ACM International Conference on Multimedia (ACM-MM'14), Orlando, FL, USA, Nov. 2014, pp. 1041–1044.
[10] J. Singh and R. Joshi, "Background sound classification in speech audio segments," in 2019 International Conference on Speech Technology and Human-Computer Dialogue (SpeD), 2019, pp. 1–6.
[11] X. Zhang, Y. Zou, and W. Shi, "Dilated convolution neural network with leakyrelu for environmental sound classification," in 2017 22nd International Conference on Digital Signal Processing (DSP), 2017, pp. 1–5.
[12] F. Vesperini, L. Gabrielli, E. Principi, and S. Squartini, "Polyphonic sound event detection by using capsule neural networks," IEEE Journal of Selected Topics in Signal Processing, vol. 13, no. 2, pp. 310–322, 2019.
[13] K. Zhang, Y. Cai, Y. Ren, R. Ye, and L. He, "Mtf-crnn: Multiscale time-frequency convolutional recurrent neural network for sound event detection," IEEE Access, vol. 8, pp. 147 337–147 348, 2020.
[14] S. K. Aswathy Madhu, "Improved deep cnn with reduced parameters for automatic identification of environmental sounds," INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH TECHNOLOGY (IJERT) Techsynod – 2019 (Volume 7 – Issue 13), 2019.
[15] N. Davis and K. Suresh, "Environmental sound classification using deep convolutional neural networks and data augmentation," in 2018 IEEE Recent Advances in Intelligent Computational Systems (RAICS), 2018, pp. 41–45.
[16] Q. Kong, Y. Xu, W. Wang, and M. D. Plumbley, "Sound event detection of weakly labelled data with cnn-transformer and automatic threshold optimization," IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 28, pp. 2450–2460, 2020.
[17] J. Sang, S. Park, and J. Lee, "Convolutional recurrent neural networks for urban sound classification using raw waveforms," in 2018 26th European Signal Processing Conference (EUSIPCO), 2018, pp. 2444–2448.
[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1409.1556