UrbanSound Classification P10

Jay Jagtap	Rohan Pillai	Sri Harsha Varma	Sai Shashank N
jjjagtap@ncsu.edu	rspillai@ncsu.edu	suppala@ncsu.edu	snayabu@ncsu.edu

1 Background

Classification of images is a very popular research area with several applications in information classification, indexing and retrieval. However, there isn't much research in a related area of audio classification particularly music, speech or sounds classification. Some of the current research in this area is only restricted to identification of auditory scene type. There isn't any research in reliable identification of the source of sound such as an idle engine, chirping of birds, etc [1]. Lack of reliable and labelled audio data could be attributed for this lack of any research in this domain. Lack of a common vocabulary makes it even more difficult for comparing results since the classification results vary from study to study.

This paper proposes a classification of urban sounds that is based on the taxonomy of sounds introduced in [2]. Any such taxonomy should satisfy three requirements in order to successfully address the issues stated above. Firstly, it should take into consideration any previous such taxonomy or research. Secondly, it should strive to contain as many low-level details as possible. Lastly, it should focus on sounds that contribute to urban noise pollution. To make sure this paper follows these requirements, we will be following the taxonomy introduced in [3] contributed to urban acoustics. We will be providing low-level details such as "horn", "engine", "brakes", etc.

2 Proposed Method

The audio files in the folders (fold1 - fold10) consists of a representative sample (i.e. files equally distributed from all the sources under consideration) and thus help in the 10-cross validation.

In addition to the audio files, a CSV file which consists of the meta data of the the audio files has been provided. The meta data has various attributes: slicefilename, fsID, start, end, salience, fold, classID, class

salience: A (subjective) salience rating of the sound. 1 = foreground, 2 = background.

slicefilename: The name of the audio file.

fsID: The Freesound ID of the recording from which this excerpt (slice) is taken.

fold: The fold number (1-10) to which this file has been allocated.

start: The start time of the slice in the original Freesound recording.

end: The end time of slice in the original Freesound recording.

class: The class name: airconditioner, carhorn, childrenplaying, dogbark, drilling, engineidling, gunshot, jackhammer, siren, street music.

classID: A numeric identifier of the sound class: 0 = airconditioner 1 = carhorn 2 = childrenplaying 3 = dogbark 4 = drilling 5 = engineidling 6 = gunshot 7 = jackhammer 8 = siren 9 = street music

To deal with audio files, we have used the publicly available Python library librosa which is a package for music and audio analysis. It has basic functionalities such as loading and pre-processing the audio files. When we load any audio file using librosa, it returns a tuple with two objects. The first item is an audio time series array for the audio file, and the second object is its corresponding sampling rate.

After loading an audio file, we now need to create a mel spectrogram for it. We cannot plot the frequencies at their present scale because humans cannot perceive frequency differences at a linear scale. For instance, humans can identify the difference 500 Hz and 700 Hz but will not be able to tell the difference between 10k Hz and 11k Hz. So for this purpose, we first need to change the scale accordingly. This scale is called the mel scale [4]. A mel spectrogram is a spectrogram where the frequencies are converted to the mel scale. Below is a mel spectrogram for the above audio file. Creation of the spectrogram or wave-plot is often the prerequisite for the sound classification neural

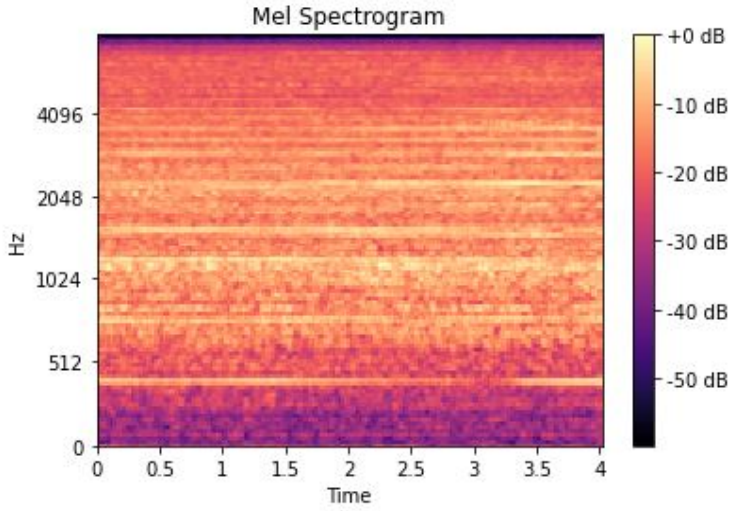

Figure 1: Mel Spectrogram

network. Spectrograms are a useful technique for visualising the spectrum of frequencies of a sound and how they vary during a very short period of time. A mel is a number that is indicative of the pitch of the audio file. Librosa is capable of extracting the power spectrogram for each mel over time as well as a function for easy display of the resulting mel spectrogram.

After extracting the data .wav files and stacking it in feature numpy arrays, we feed it to convolutional Neural Network for classification. Convolutional Neural Network perform better with image classification due to their feature extraction and classification parts. During this stage of our project, we have used Neural Network Architecture. Our neural network has two convolution layers, each convolution followed by a max pooling layer to extract dominant features. We also add a dropout layer to reduce reliance on a single feature. Each of the convolution layer has a tanh activation function and we have a softmax layer at the end to get the classification probability vector.

In Figure 3, the summary of the model is presented.

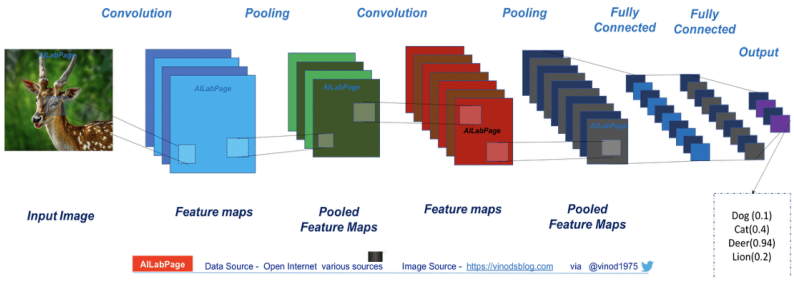

Figure 2: About CNN

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 60, 41, 24)        456
_________________________________________________________________
batch_normalization (BatchNo (None, 60, 41, 24)        96
_________________________________________________________________
activation (Activation)      (None, 60, 41, 24)        0
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 30, 20, 24)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 30, 20, 32)        6944
_________________________________________________________________
batch_normalization_1 (Batch (None, 30, 20, 32)        128
_________________________________________________________________
activation_1 (Activation)    (None, 30, 20, 32)        0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 15, 10, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 15, 10, 64)        18496
_________________________________________________________________
batch_normalization_2 (Batch (None, 15, 10, 64)        256
_________________________________________________________________
activation_2 (Activation)    (None, 15, 10, 64)        0
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 7, 5, 64)          0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 7, 5, 128)         73856
_________________________________________________________________
batch_normalization_3 (Batch (None, 7, 5, 128)         512
_________________________________________________________________
activation_3 (Activation)    (None, 7, 5, 128)         0
_________________________________________________________________
global_max_pooling2d (Global (None, 128)               0
_________________________________________________________________
dense (Dense)                (None, 128)               16512
_________________________________________________________________
dense_1 (Dense)              (None, 10)                1290
=================================================================
Total params: 118,546
Trainable params: 118,050
Non-trainable params: 496
```

Figure 3: Model Summary

3 Plan & Experiment

3.1 Dataset

The dataset contains 8,732 labelled sound samples (less than 4s) classified into 10 classes- Air Conditioner, Car Horn, Children Playing, Dog Bark, Drilling, Engine Idling, Gun Shot, Jackhammer, Siren, and Street Music. The dataset follows the taxonomy described above. These sound excerpts are digital audio files in .wav format. Sound waves are digitised by sampling them at discrete intervals known as the sampling rate. The excerpts are sorted into 10-folds and were taken from field recordings uploaded to `www.freesound.org`.

3.2 Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a Deep Learning algorithm which takes an image as the input and assigns significance to different aspects in the input image and be able to distinguish one from the other. When compared to other classification algorithms, Convolutional Neural Network requires less pre-processing.

For sounds, a successful application of CNN was first introduced by Piczak [5]. Our data pre-processing step was inspired from them. To pre-process the data, we have tried to get equal segments from different length audio clips and separate audio features are extracted from them so as to pass

them as separate channels just like RGB channels for a static image. After data pre-processing is complete, mel-spectrograms and their corresponding deltas were calculated for each sound clip in our dataset. We divide the clips into segments of 60 bands for 41 frames. This will transform the dataset into 60 rows and 41 columns. Mel-Spectrogram and their deltas are two channels that can be given to the CNN as input.

Our CNN model consists of four convolutional layers followed by a batch normalization layer, activation layer and a max pooling layer. Towards the end a global pooling layer was added followed by a fully connected layer. A Dense layer with softmax activation function was added to derive the class-wise probability of the different labels. For the entire model, we've used Adam optimiser with 10^{-4} learning rate and categorical cross-entropy loss function.

We've trained the neural network with 10-folds cross-validation in order to get a better measure of accuracy. To get the final accuracy of our model, we will average out the accuracy of the predictions over all folds.

3.3 Long Short Term Memory Networks

Long Short Term Memory Networks (LSTM) are a special type of Recurrent Neural Network, capable of learning long-term dependencies. Long Short Term Memory Networks are useful where we require the network to retain information over long period of time. Long Short Term Memory Networks are developed to avoid the long-term dependency problem.

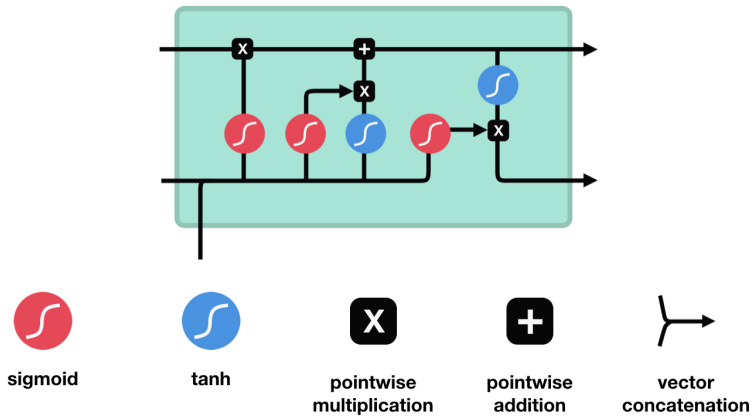

Figure 4: Long Short Term Memory Network

Long Short term Memory models are known to work well on time series data. RNNs understand the sequential data by making the hidden state at time t dependent on hidden state at time t-1. The mel-spectrograms have a time component as it is a feature of array of (128,1) steps. The RNNs can do much better in classifying temporal data like Mel-Spectrograms. Following is the architecture of RNN used:

Our first layer is of 100 units with return sequences set to True which transforms the input of (128,1) to (128,100) thus giving us data about sequences at each unit time. We add a dropout layer to counter over fitting and give this output of (128,100) to another LSTM layer which compresses the data back to 1 D array. We then add a fully connected layer and a final soft max layer for class predictions. The LSTM Model gives much better performance as compared to the CNN model.

3.4 RCNN

After studying the results given by both CNN and RNN Model, we propose a novel Neural Network Architecture. We propose to extract the best features from CNN and RNN and combine them. In this architecture, instead of a Sequential Model we deploy a Functional Model. Following is the architecture of our model.

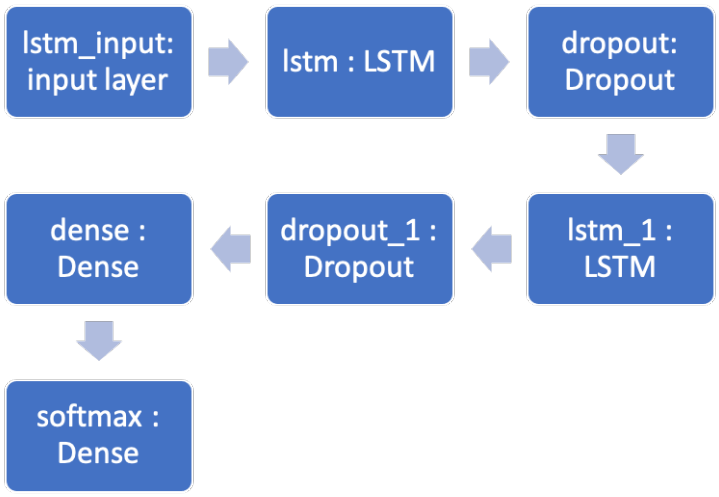

Figure 5: Long Short Term Memory Network

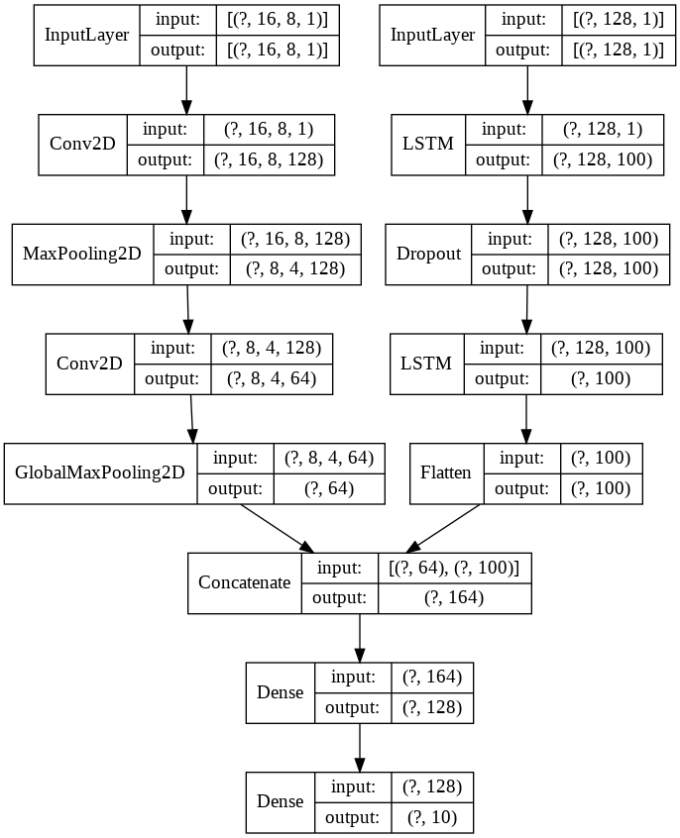

Figure 6: RCNN

We train a CNN and RNN in parallel and concatenate their results which is fed to a Fully Connected Network. The CNN architecture is 2 Convolutions with Max pooling whereas we use the same model architecture for RNN as described in the previous section. It is important to note that the input layer is represented as tuple i.e. Shape of inputs for CNN and LSTM is different. The input for CNN is of shape (16,8,1). The mel-spectrogram is reshaped in order to apply 2D convolutions on the data. The input shape for RNN is (128,1) which represents (time steps, features). This makes the input layer of shape = ((16,8,1), (128,1)). The output of both CNN and RNN branch are flattened and

concatenated to get the best of both models. Finally this flattened layer is given as input to a fully connected layer. This model performs with average accuracy of 85.98% which is better than the human accuracy measured of 82.3%.

3.5 Experiment

Our data-set is divided into 10 folds for classification. This enables us to do k-cross validation and have a better measure of our accuracy. The tools we have used are python, numpy, librosa library for audio processing and tensorflow for fitting data to Neural Networks. We have leveraged Google Cloud Platform with GPU support for training purposes.

4 Results

From Table 1, it can be observed that the validation accuracy is not same across all the folds, The validation accuracy for fold 1 is surprisingly low for all the models, which could be attributed to the fact that the validation dataset is skewed for the first fold. Figure 7 represents the same in the pictorial format.

Fold Number	CNN	LSTM	Proposed Architecture
1	54.36	38.9	46.22
2	62.18	61.7	78.04
3	56.98	71.35	86.05
4	68.44	79.69	88.4
5	48.22	92.6	92.22
6	66.24	93.5	91.00
7	72.34	93.91	93.67
8	46.76	93.54	96.52
9	51.03	96.44	93.87
10	48.97	94.62	93.66
Average	57.5	81.64	85.98

Table 1: Average Model Accuracy

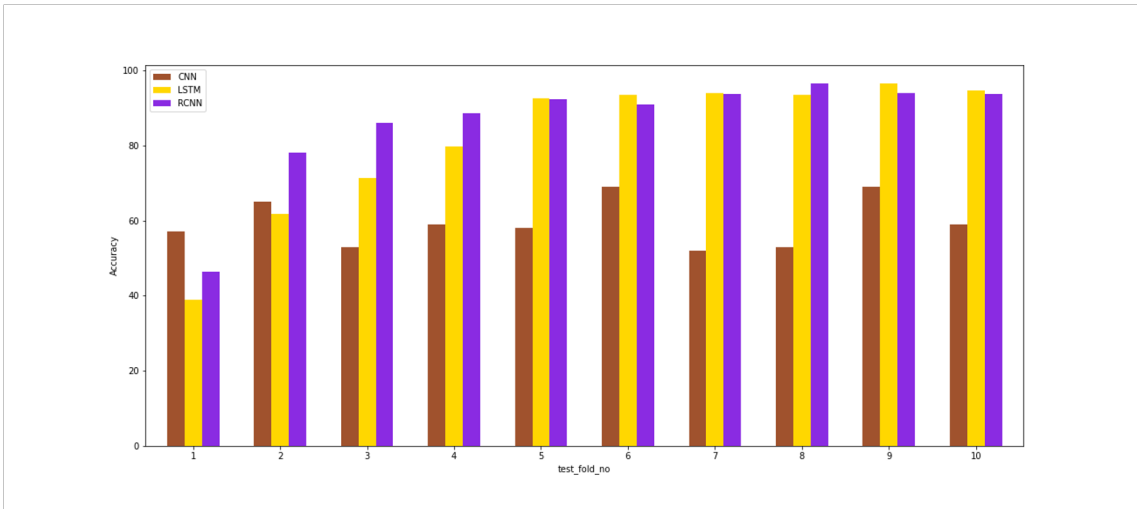

Figure 7: Model Accuracy comparison per fold

Further, figure 8 represents the average model accuracy when 10 fold cross validation is employed. It can be clearly observed that CNN is the least accurate model on our dataset with nearly 58% accuracy. LSTM and our proposed architecture have outperformed CNN by huge margins with approximately 82% and 86% accuracy. In addition, it has is observed that the human accuracy on this dataset is nearly 82% which is lower than the accuracy of our proposed model.

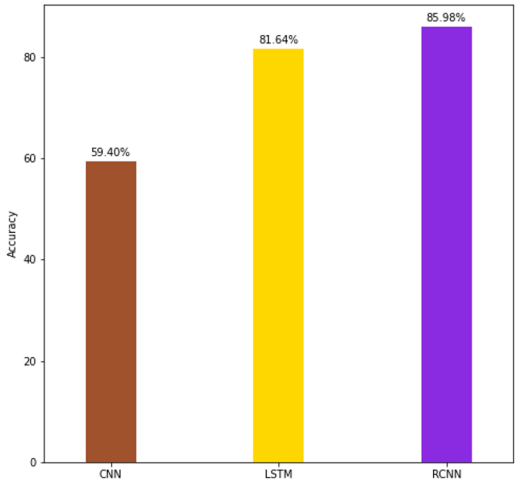

Figure 8: Average Model Accuracy

In Fig.9, the confusion matrix for the proposed architecture is presented, the Y-axis corresponds to the actual class and X-axis represents the predicted class. It is observed that, class 8 can be classified as class 5 sometimes, which increases false positives for class 5 and true negatives for class 8. This could be attributed to the fact that some sound frequencies could be too close to draw a separation and an addition of little noise can distort them and result in classification error. Excluding this anomaly, it is clear that the predicted classes are highly accurate.

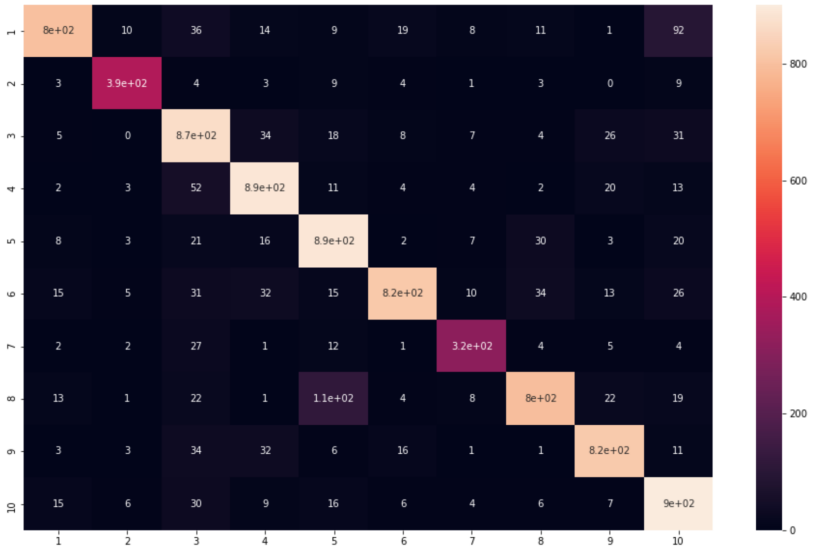

Figure 9: Confusion Matrix

5 Conclusion

The average accuracy for our proposed model after 10 cross-validation is 85.98%, which is better than the accuracy that could be obtained from LSTM and CNN models. This accuracy is in fact, higher than how accurately humans can distinguish these sounds (83.2%).

We believe this project can be helpful in aiding deaf people in their day to do activities, for indexing and retrieval of content-based multimedia, and also to improve the safety and security alerts from smart home assistants.

In addition, we believe by extensive hyper parameter search to improve the accuracy of the model. Also, we can explore ensemble/combination of different models and see if more features can be extracted from sound data in addition to the mel spectrogram.

References

[1] Chu, S., Narayanan, S. & Kuo C.C. (1995) Environmental sound recognition with time-frequency audio features., *IEEE TASLP*, 17(6):1142–1158, 2009.

[2] Salamon, J., Jacoby, C., & Bello, J.B. (2014) A Dataset and Taxonomy for Urban Sound Research. *In Proceedings of the 22nd ACM international conference on Multimedia (MM '14). Association for Computing Machinery New York, NY, USA, 1041–1044.* DOI:https://doi.org/10.1145/2647868.2655045

[3] Brown, A.L., Kang, J., & Gjestland, T. (2011) Towards standardization in soundscape preference assessment. Applied Acoustics, 72(6):387–392.

[4] Stevens, S., Volkmann, J., Newman, E.B. (1937). A Scale for the Measurement of the Psychological Magnitude Pitch.

[5] Piczak, Karol. (2015). Environmental sound classification with convolutional neural networks. 1-6. 10.1109/MLSP.2015.7324337.

Image Source:
Figure 2: https://vinodsblog.com/2018/10/15/everything-you-need-to-know-about-convolutional-neural-networks/
Figure 4: https://commons.wikimedia.org/wiki/File:The_LSTM_cell.png

GitHub: https://github.ncsu.edu/engr-csc522-fall2020/P10