Urban Sound Classifications

Hady Sylla, Saurabh Swaroop, Uma M Kugan
Indiana University
Bloomington, Indiana

ABSTRACT

In real life, there is plenty of unstructured data available which largely remains unexploited. For example, while we communicate with people, we not only give attention to the words but to the emotions, body language, context, etc. At times processing these unstructured data can reap us easy rewards. While there is extensive research is conducted in areas such as speech and music, work on the urban acoustic environment is scarce. In this research work, processing unstructured audio containing a mixture of the air conditioner, car horn, children playing, dog barking, drilling, etc., for extracting necessary information. Most of the previous researches used audio set from carefully produced movies or audios recorded in a controlled environment. One of the major challenges will be lack of labeled audio data set. This work will help in mitigating the tremendous effort put on manually segregating the data set. While working with the urban environment it is not possible to confine or separate the audio sets obtained. This lack of common vocabulary while working with urban sounds is the other major challenge. The aim of this paper is to identify and extract features to create efficient machine learning models for urban sound classification in real-life noise conditions.

KEYWORDS

Urban Sound Classification, GMM-HMM, MFCC, Neural Network, Random Forest, Confusion Matrix

1 INTRODUCTION

In recent years, the automatic classification of these complex and dynamic urban sound is a growing research field, and it is an important aspect of various emerging applications, and therefore it has gained large focus in recent years. There are many interesting applications of the environmental sound classification. Automatic recognition of the surrounding environment allows hearing aid machines to switch between programs and work with minimum user interference [12]. There were many previous work in this area to classify environment sound which paved the way to a new set of features. Peltonen [9] used mel frequency cepstral coefficient (MFCC) as features and gaussian mixture models (GMM) and neural network as classifiers. Chu [3] have proposed a combination of MFCC and matching pursuit algorithm for feature extraction to classify a set of 14 natural sounds.They reported an accuracy was 83.9% using GMM classifier. In this project, we are extracting Mel Frequency Cepstral Coefficients algorithm (MFCC) and then applied classifiers such as Neural Network, GMM-HMM and Random Forest to ten different classes of urban sound.

2 URBANSOUND CLASSIFICATION - REAL WORLD APPLICATION

Communication is the important aspect of sound perception in all living organisms who constantly rely on all the surrounding sounds from the environment. Automating the classification of environmental sounds can be used in many day today applications such as remote surveillance, home automation, mobile devices, hearing aid, etc. An interesting application is the use of home surveillance equipment which identifies different sounds produced in an interior environment and alerts the user accordingly.

3 FEATURE EXTRACTION

The extraction of the best parametric representation of sound signals is an important task to produce a better recognition performance. The efficiency of feature extraction affects the accuracy of the model and so it is very important to choose the right feature for the better performance of the model. There are many algorithms such as LPC,LPCC,HFCC,MFCC to extract features. We used Mel Frequency Cepstral Coefficients algorithm (MFCC), MFCC Delta with first order and MFCC Delta with second order.We also tried other sound features like Tonnetz, Chroma, spectral centroid but none of them helped in improving accuracy. As we trained model by stacking features of all the sound files of each class example, it was not possible to include features which didn't add values. To get a good balance of training time and accuracy three features were used.

3.1 Mel Frequency Cepstral Coefficients - (MFCC)

Mel Frequency Cepstral Coefficents (MFCCs) is most common feature used for automatic sound recognition. They were introduced by Davis and Mermelstein in the 1980's, and have been state-of-the-art ever since [7].

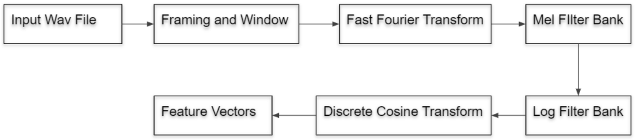

Figure 1: MFCC Steps

4 HMM

HMM is the sequence model that computes a probability distribution over possible sequences of labels and choose the best label sequence. It is defined as a variant of a finite state machine having a set of hidden states, Q, an output alphabet, O, transition probabilities, A, output probabilities, B, and initial state probabilities,$\prod$.

Each state produces an output with a certain probability (B). An HMM is said to be a triple, (A, B, $\prod$) [14].

Formal Definition:

Hidden states Q = $q_i, i = 1, ..., N$.

Transition probabilities A = a_{i_j} = $P(q_j\ at\ t+1\ |q_i\ at$ t), where P(a | b) is the conditional probability of a given b, t ≥ 1 is time, and $q_i \in Q$.

Informally, A is the probability that the next state is q_j given that the current state is q_i.

B = b_{i_k} = $P(o_k|q_i)$, where $o_k \in O$.

Informally, B is the probability that the output is o_k given that the current state is q_i.

$\prod = p_i = P(q_i\ at$ t=1).

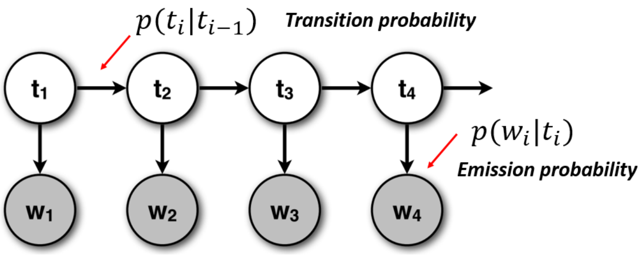

Figure 2: HMM (image adapted from CS6501 of the University of Virginia) [1]

5 NEURAL NETWORK

Neural Network is an essential tool for mapping complicated input-output relationships. It is Made up of layers of neurons, and each neuron is a transformation function. The most important step is training the model which involves minimization of a cost function. Once training is finished and validated, the application is cheap and fast. In 1958, Frank Rosenblatt introduced a training algorithm called perceptron. which is very easy, and simple of a neural network and it consists of a single neuron with adjustable weights and a threshold. A multilayer perceptron (MLP) is a class of feed forward artificial neural network which includes an input layer, a hidden layer, and an output layer. MLP uses back propagation for training [10].

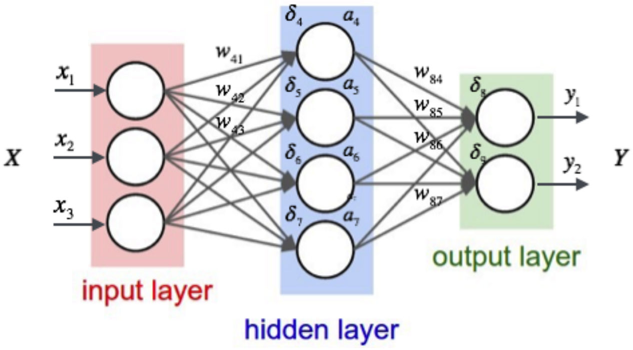

Figure 3: Neural Network [13]

6 RANDOM FOREST

Random forest classifier creates multiple decision trees from a randomly selected subset of the training set and then aggregates them to decide the final class of the test object. We need first to choose random samples from a given data set, construct a decision tree for each sample and get a prediction result from each decision tree. Then Perform a vote for each predicted outcome and prediction with the most votes as the final prediction [2].

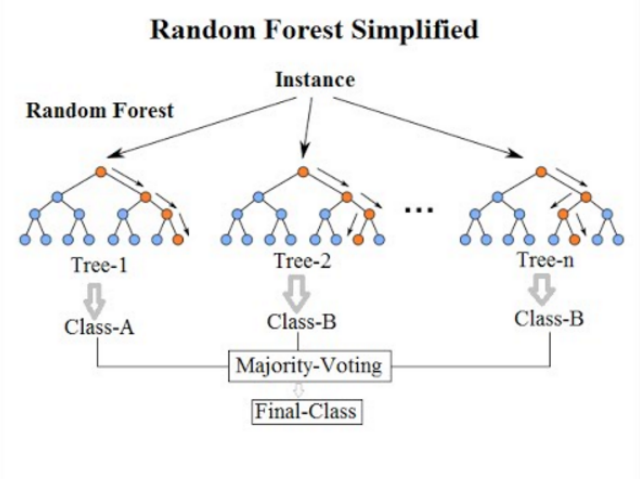

Figure 4: Random Forest [6]

7 EXPERIMENTS

7.1 System Design

The following diagram shows the high-level flow of the process we implemented in this paper.As we have used two different approach to solve the problem, there are some differences in hyperparamters. In HMM, three features are used but in neural network an extra feature tonnetz is used.

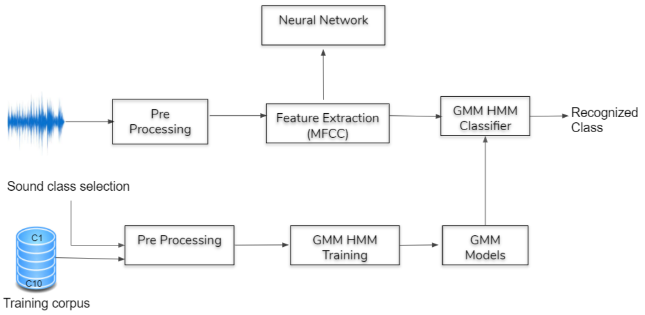

Figure 5: System Design

7.2 Dataset

We used the UrbanSound 8k Dataset by Justin Salamon, Christopher Jacoby, and Juan Pablo Bello [11]. The UrbanSound 8k dataset contains 8732 real-field recording samples from 10 classes of different sound sources: air conditioner, car horn, children playing, dog bark, drilling, engine idling, gunshot, jackhammer, siren and street music. Samples have a duration of fewer than four seconds. We used Librosa library for sound file processing and HMMLearn library for training.

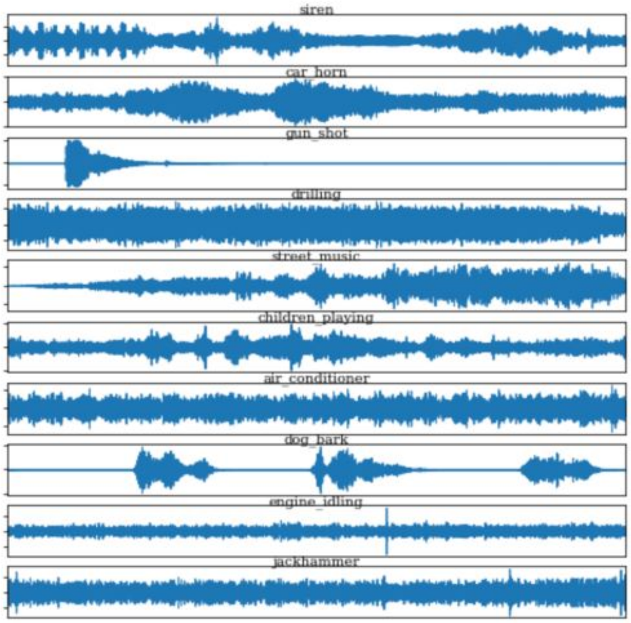

Figure 6: Wave plot of sample audio files from each class

7.3 Pre Processing

We segregated and split all the audio files into respective class folders. When we analyzed the files, we found out that some of the files were less than two seconds. When we tried to filter out the files which have duration's of more than 3.5 seconds, the files that belong to gun-shot classes were left very few files to train. Hence we did not do any filtering w.r.t. file duration and proceeded with default length. We used fifty files from each class for testing our HMM model.

7.4 Features Extraction

For each of the ten classes mentioned above: We are extracting following features for our GMM-HMM and Neural Network classification:

MFCC - It is the small set of feature which concisely describes the overall shape of a spectral envelope. We have passed the input signal to librosa.feature.mfcc to return a sequence of 24 mfcc.

MFCC Delta - This feature gives the local estimate of the derivative of the input data along the time axis, i.e., change in the coefficients.

MFCC Delta Delta - This feature gives the local estimate of the second order derivative of the input data along the time axis, i.e., change in delta values..

Tonnetz - This feature computes the tonal centroid features for each frame.We have used this feature while training a neural network.

We tried to extract other features like Chroma, Mel, Zero crossing rates. But it did not improve our results as these features are more related to musical notes. We used MFCC, MFCC Delta, MFCC Delta Delta in GMM-HMM model and MFCC and Tonnetz in Neural Network.

7.5 HMM Model Implementation

We implemented Hidden Markov Model with Gaussian mixture emissions on the data set using hmmlearn library. To prepare input, we first extracted features for all the files of each class and stacked them in one matrix where each row represented feature vector.The probility emissions of each hidden states are modeled on Gaussian Mixture Model emissions. The optimal hidden states transition probabilities are obtained by using the Viterbi algorithm which is the defualt algorithm with the GMMHMM.fit method in HMMlearn library. The Viterbi algorithm is a dynamical programming algorithm which is used to compute the most probable path [4, 5]. The training hyper-parameters which gave best results are as follows: n_iter = 1000 , Number of sequence = 12 , MFCC Frames = 20. We were able to get 86.6% accuracy.

7.6 Neural Network Model Implementation

After feature extraction, we have split the data sets into train and test and built the model out of it. The model uses a hyperbolic tangent function or 'tanh' function as the activation function. It is employed as t which has itś output zero centered hence having easier optimization. The solver/optimization algorithm used is Adam stochastic optimization algorithm which employs momentum for achieving faster decent than normal gradient decent algorithm. When we used MFCC, Tonnetz and Delta we got 81% score, but one of the class performance was reduced to zero. So we used only the mfcc and tonnetz, and the score was increased to 85% and, when we just used delta accuracy was dropped down to 74%. Similarly we tried with different activation function, and tanh was 85%, and it outperformed relu and logistic whose scores were 63% and 75%.

7.7 Random Forest Model Implementation

We used sklearn.ensemble to implement the random forest model for the data set. The most critical parameter for using this model is n_estimators which denote the number of trees in the forest. If the number of trees defined is on the higher side, the predictions are more stable, but it takes longer to compute [8]. We used the number of trees in the forest as 500. Random Forest model is one of the weakest model and we were able to get only 28.84% accuracy.

8 RESULT AND ANALYSIS

Our best prediction on training the three models:

GMM-HMM	86.6
Neural Network	81.04
Random Forest	28.84

There are many factors which could be more experimented with to achieve better accuracy:

White Noise - There are noise contamination in the files so proper de-noising before training could give better results.

Classes With Similar Spectrogram - As shown in spectrogram for all the classes, few have very similar spectrogram which created confusion. For example, the car horn was confused with Jackhammer 6 times out of 50.

Feature Extraction - Three features MFCC, Delta and Delta Delta were used which gave the best results. Several other features were experimented with like Chroma, Tonnetz, etc. but they did not improve the performance. More experiments need to be done with features in the future.

8.1 Confusion Matrix

Confusion matrix is the table that is used to display or describe the performance of the model. It contains the information about actual and predicted classification calculated by the machine learning model.

Looking at our confusion matrix for all three models, it indicates that the model performs the same way that human would respond or classify the sound.

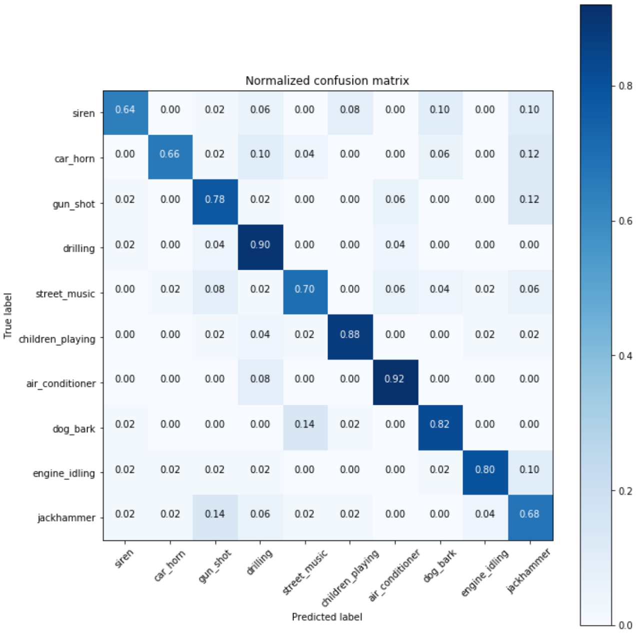

Figure 7: Normalized Confusion Matrix - HMM

9 CONCLUSION

In this paper, we were able to compare GMM-HMM Model with Neural network and Random Forest and infer that GMM-HMM is performing well whereas Neural Network fits more non-linearly. We have experimented the models with different feature extraction and went ahead with the features that performed well. We studied the challenges presented by the data set, and we would like to explore more about feature engineering for urban sound classification and implement various model such as CNN, RNN. We

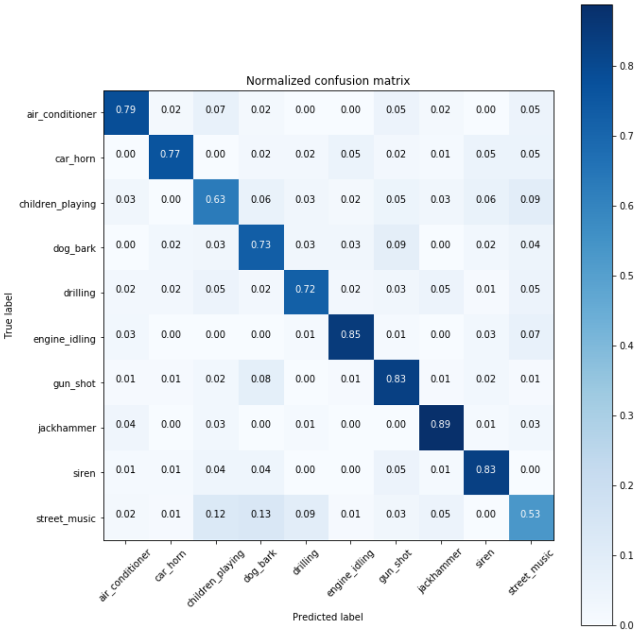

Figure 8: Normalized Confusion Matrix - Neural Network

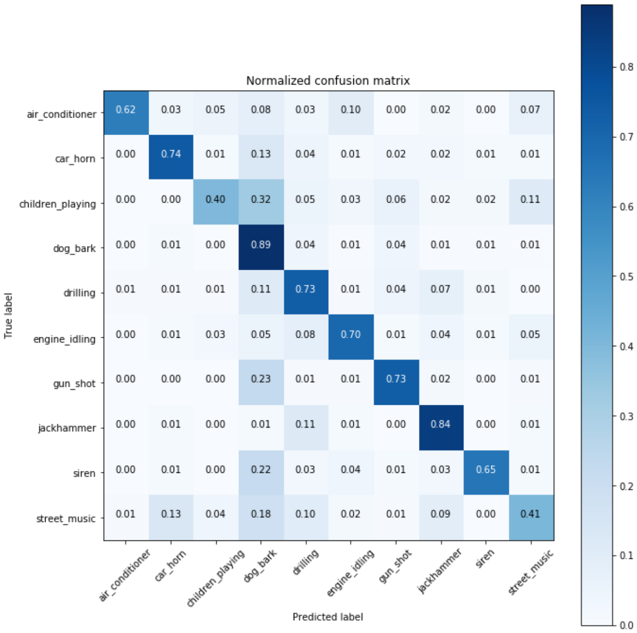

Figure 9: Normalized Confusion Matrix - Random Forest

believe that the data set will open the path to new and exciting research in sound and multimedia applications with a focus on urban environments and sound-space as an urban dwelling is at the exponential growth.

ACKNOWLEDGMENTS

We want to thank Professor Minje Kim for his continued support, guidance, and encouragement throughout the semester. We sincerely appreciate the learning opportunities provided by Professor Minje Kim and his teaching assistants who constantly motivated us to go beyond our limits. This project would not have been complete without their dedicated support and our sincere thanks to all our fellow students.

REFERENCES

[1] David S. Batista. 2017. Hidden Markov Model and Naive Bayes relationship. (2017). http://www.davidsbatista.net/blog/2017/11/11/HHM_and_Naive_Bayes/

[2] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[3] Selina Chu, Shrikanth Narayanan, and C-C Jay Kuo. 2009. Environmental sound recognition with time–frequency audio features. *IEEE Transactions on Audio, Speech, and Language Processing* 17, 6 (2009), 1142–1158.

[4] James H. Martin Daniel Jurafsky. 2018. Speech and Language Processing. (2018). "https://web.stanford.edu/~jurafsky/slp3/A.pdf"

[5] HMMlearn Developers. 2010. HMM Tutorial. (2010). "https://hmmlearn.readthedocs.io/en/latest/index.html"

[6] William Koehrsen. 2017. Random Forest in simple Explanation. (2017). "https://medium.com/@williamkoehrsen/random-forest-simple-explanation-377895a60d2d"

[7] James Lyons. 2013. Mel Frequency Cepstral Coefficient (MFCC) tutorial. (2013). "http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs"

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[9] Vesa Peltonen, Juha Tuomi, Anssi Klapuri, Jyri Huopaniemi, and Timo Sorsa. 2002. Computational auditory scene recognition. *Acoustics, speech, and signal processing (icassp), 2002 IEEE international conference on* 2 (2002), II–1941.

[10] Mingyue Qiu and Yu Song. 2016. Predicting the direction of stock market index movement using an optimized artificial neural network model. *PloS one* 11, 5 (2016), e0155133.

[11] J. Salamon, C. Jacoby, and J. P. Bello. 2014. A Dataset and Taxonomy for Urban Sound Research. In *22nd ACM International Conference on Multimedia (ACM-MM'14)*. ACM, Orlando, FL, USA, 1041–1044.

[12] Sunit Sivasankaran and KMM Prabhu. 2013. Robust features for environmental sound classification. In *Electronics, Computing and Communication Technologies (CONECCT), 2013 IEEE International Conference on*. IEEE, IEEE International Conference on Electronics, Computing and Communication Technologies, Bangalore, India, 1–6. https://hal.inria.fr/hal-01456201/document

[13] Venelin Valkov. 2017. Creating a Neural Network from Scratch - Tensorflow for Hackers. (2017). "https://medium.com/@curiousily/tensorflow-for-hackers-part-iv-neural-network-from-scratch-1a4f504dfa8"

[14] Paul E. Black Vreda Pieterse. 2008. "hidden Markov model", in Dictionary of Algorithms and Data Structures. (2008). "https://www.nist.gov/dads/HTML/hiddenMarkovModel.html"