Urban Sound Classification

Final Project Report

Reham Mohamed Aburas
Purdue University
raburas@purdue.edu

Habiba Farrukh
Purdue University
hfarrukh@purdue.edu

Pavani Guttula
Purdue University
pguttula@purdue.edu

Aly Shehata
Purdue University
ashehat@purdue.edu

1. INTRODUCTION

In today's urban environments, we experience many different kinds of sounds that we, as humans, can instantly recognize and interpret. These urban sounds can include the sound of construction work, vehicle horns, street music, gun-shots, etc.

The ability to classify these sounds using Data Mining / Machine Learning algorithms opens the door for many future work and applications. For instance, self-driving vehicles can use urban sound classification to augment the vehicles understanding of its surroundings. If a vehicle can accurately classify a vehicle horn, for instance, it can then react to that sound if needed. Apart from this, sound classification could help in assisting people with lower hearing ability by providing them with a context of their surroundings. Additionally, sound classification can be used in a magnitude of other applications such as security, law-enforcement, and context-aware maintenance of noise-levels in crowded areas.

In this project, we propose to apply a set of different data mining algorithms to an existing urban sound dataset: https://www.kaggle.com/pavansanagapati/urban-sound-classification, and then perform a survey of the performance across the different algorithms. We plan to make methodical comparisons among our different algorithms in terms of their assumptions, parameter tuning procedures, and performance. Finally, we hope to gain enough insight on the data and the domain of urban sounds, while also attempting to apply our best performing algorithms to newly recorded, real-world examples.

2. DATASET DESCRIPTION

2.1 Literature Survey

Environmental sounds classification has gained momentum over the years. Early attempts have used signal processing and machine learning techniques as wavelets filterbank [6, 3], rule-based classifiers [10], SVMs and KNN [9], etc. These techniques use traditional audio features as cepstral coefficients, their derivatives, pitch, timbre, among others. Recently, deep convolutional neural networks (CNN) were applied on audio spectrograms [7, 5, 1, 4]. These approaches claim to capture new patterns to distinguish different types of sound, in which traditional audio features fail. Deep neural networks, however, are highly dependent on the size of the dataset. Therefore, data augmentation techniques were used in [7] a to overcome the data scarcity. Other approaches proposed using transfer learning methods [2] where they pretrain a model using large unlabeled sound and video dataset to learn the low level features and then use the pretrained model for the target task.

In this project, we plan to compare some of the different learning techniques for environmental sound classification and verify the assumption of CNNs superiority on traditional techniques. We also plan to test the effect of increasing the dataset size by applying data augmentation techniques on the learning task.

2.2 Data Collection and Preprocessing

UrbanSounds dataset was introduced with a sound taxonomy in [8]. The dataset contains about 18.5 hours of real-world manually annotated sounds across 10 classes. To construct this data, real recordings were collected from FreeSounds (an online sound repository containing over 160K user-uploaded recordings) that correspond to urban sound classes.

2.2.1 Original Dataset

UrbanSouns8K is a subset of UrbanSounds dataset organized for classification tasks. This dataset is composed of short audio snippets with maximum duration of 4 seconds. The dataset assumes a single sound source, so each segment corresponds to a single class. To maintain uniform class distribution, the audio snippets for each class are limited to 1000 snippets. The total is 8732 labeled snippets (8.75 hours). The dataset provided in folders is divided into 10 folds.

2.2.2 Data Augmentation

We performed Data augmentation on the Original dataset for two reasons:

- To evaluate the algorithms on more realistic audio files rather than on the original dataset with less external

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

noise.

- To increase the size of our dataset.

We used MUDA library for augmenting audio data files to apply the deformers Pitch shift(to raise or lower the pitch), Time Stretch(to change the speed or duration of an audio signal without affecting its pitch.), Background Noise, Dynamic Range Compression(to reduce the volume of loud sounds or amplify quiet sounds) to each audio file. We wanted to analyze how adding these deformities to the audio file would effect the classification rate of algorithms. To achieve this we maintained same feature extraction procedure for both original and augmented dataset. Data augmentation increased the size of our data by 10 times. Due to resource limitation, we sampled only 10 percent of the augmented dataset and added it to the original data. So our augmented datset has double the size of the original dataset.

2.2.3 *Preprocessing*

For both Original and Augmented datasets, we used a 70-30 split of data for training and testing purposes respectively. For Cross Validation purposes,out of the training data, we generated 10 folds. From these 10 folds, we iteratively selected 9 different folds as training data and tested on the last remaining fold.

2.3 Feature extraction

To gain more insights on the data, we extracted different sound features from the data. We wanted to have both 1D and 2D features to test the data on various algorithms and gain insights about how having different types of features would effect the performance of our algorithms.

2.3.1 *1D Features*

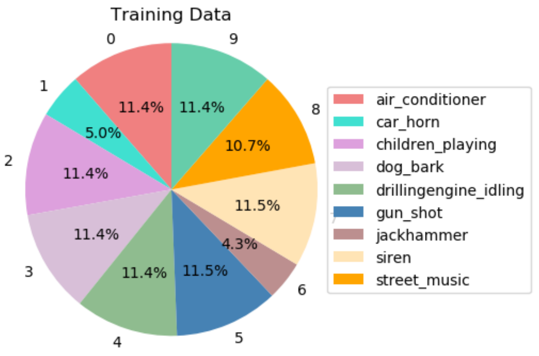
Based on our literature survey, we decided to extract those set of features which represent frequency patterns in the audio samples over time. Our goal was to identify the components of the audio signal that are good for identifying the linguistic content and discarding all the other like background noise, emotion etc. For this purpose, we selected four different commonly used sound features. A brief description of each of the 4 features is given below:

- **MEL-Frequency Cepstrum Coefficient**- A representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear MEL scale of frequency.

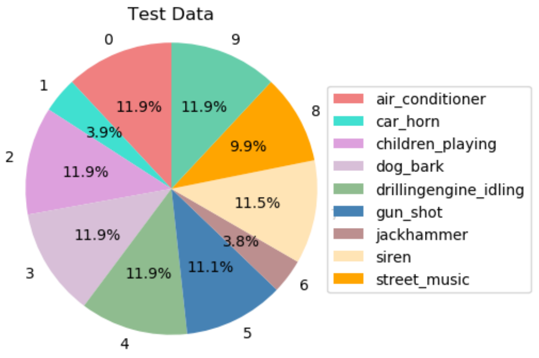
- **MEL-Spectogram**- An acoustic time-frequency representation of a sound similar to MFCC. This feature is directly extracted from MFCC with the only difference being that the spectogram is in MEL scale which a human ear can recognize better.

- **Chromagram**- Chromagram also referred to as ”pitch class profiles”, is a powerful tool for analyzing music whose pitches can be meaningfully categorized (often into twelve categories) and whose tuning approximates to the equal-tempered scale.

- **Spectral Contrast**- Deals with the strength of spectral peaks, valleys, and their difference separately in each sub-band, and represents the relative spectral characteristics.



(a) Training Dataset



(b) Test Dataset

Figure 1: Class distribution among the training and test datasets

We used a python library, Librosa, for extracting these features for each of the audio sample in our dataset. Librosa generated a one dimensional set for all the 4 features, where each feature was mapped as frequency vs time. To convert these features into 1 dimension, we extracted mean of each feature values over time. As a result of this step, we generated a total of 187 features extracted solely from the audio samples. For each audio sample, the first 40 features represented the MFCC values of the audio signal, the next 12 features represented the 12 pitch classes of the chromagram, the following 128 features represented the MEL-Spectrogram and finally the last 7 features represented the spectral contrast values. We used these 1D features for the implementation of our linear set of algorithms i.e. Logistic Regression (LR) and Support Vector Machine (SVM) (details in Section 3).

2.3.2 *2D Features*

Apart from the 1D features, we also extracted two dimensional features for our dataset. For the 2D features, we computed the 2D spectrogram of each audio file. A spectrogram is a visual representation of the spectrum of frequencies of a signal as it varies with time. Similar to 1D features, we used the Librosa library for generating the spectrogram images. The size of each spectrogram image was 128x300 pixels. We used these spectrogram images as input to our Convolutional Neural Network (CNN) (details in Section 3).

2.4 Data Visualization

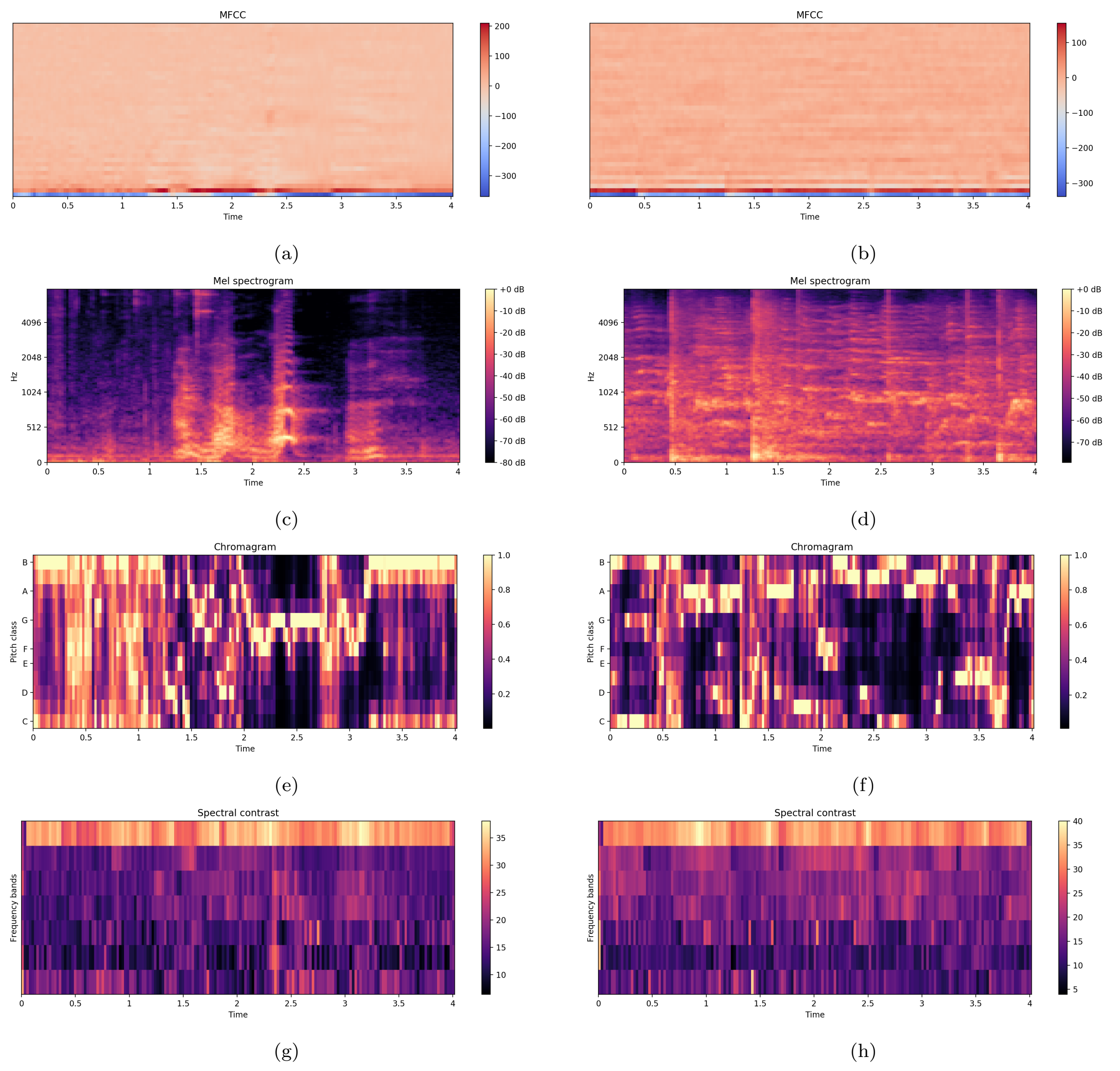


Figure 2: 2D representation of the different features extracted from audio files of two different sound classes namely 'dog bark' and 'children playing' (a) and (b) represent the MFCC of the sound signal for 'dog bark' and 'children playing' respectively. MEL-spectrogram for 'dog bark' and 'children playing' is shown in (c) and (d) respectively. (e) and (f) show the Chromagram of the two audio samples. Spectral contrast for the two samples is shown in (g) and (h).

To understand patterns in the data and the specially the audio files we used various data visualization techniques. Firstly, we wanted to make sure if there is a uniform distribution of all the classes in both training and test dataset. Figure 1a shows the representation of class label distribution in the training dataset, while figure 1b shows the distribution for test dataset. Distribution from both the datasets confirmed that we have more or less equal percentage of samples with respect to all classes.

We also wanted to understand how the audio file features look like and how different they are from each other in terms of their features for two different audio files. By plotting these 2D figures for our features we realized that the frequency of audio signals over time differ considerably for each of the two different class labels. Figure 2 shows the 2D representation of our 1D features for two different sound classes. We selected one audio sample labeled as 'dog bark' and another labeled as 'children playing' for comparing these features. We can clearly see that these two audio samples differ in the 1D features.

For 1D features, we also wanted to understand what is variation in the mean value of each feature for each different class. Figure 3 shows the mean MFCC values for the various classes. For spectral contrast, we plotted the average contrast value for the 10 classes in Figure 4. Apart from the mean feature values, we also observed the most commonly occurring i.e. mode of our features for each of the 10 classes. For MFCC and spectral contrast, we observed

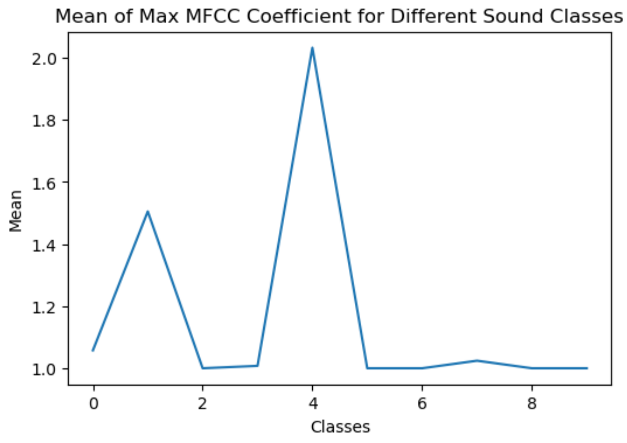


Figure 3: Mean value of MFCC for each of the 10 different classes

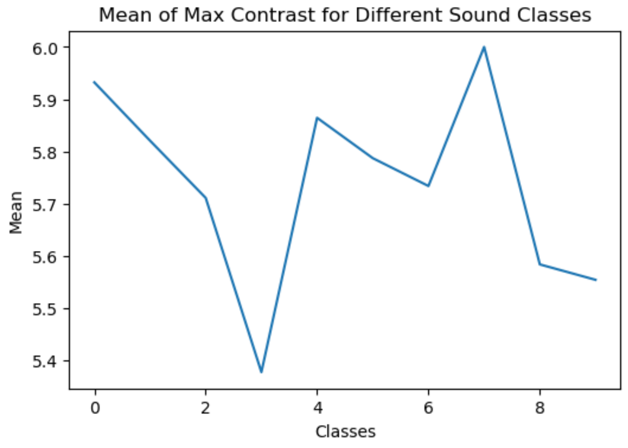


Figure 4: Mean contrast values for each of the 10 different classes

that the mode value is the same for all classes. This mode value was 1 for MFCC, while for spectral contrast, the consistent mode value was 6. For MEL Spectrogram, we saw a very diverse distribution over the 10 classes in the most commonly seen value. Figure **??** shows the mode value from the MEL-spectrogram for the 10 classes. As MEL spectogram closely aligns with human hearing, it probably only confirms that how differently we perceive the sounds of each of the classes. We observed a similar pattern in mode values for chroma feature. The mode value for chroma feature for different classes is shown in Figure 6.

We also plotted 2D Spectograms, our crucial feature fed to the Convolutional Neural Network model for different classes i.e. 'street music' and 'drilling sound'. Figure 7 shows the 2D spectogram of street music and Figure 8 shows the 2D spectogram of drilling sound. We observed a clear visual difference in the spectograms of both the classes.

3. ALGORITHMS AND EVALUATION

We implemented Logistic Regression(LR) and Support Vector Machine(SVM) Algorithms primarily to classify the dataset using our 1d features. We implemented Convo-

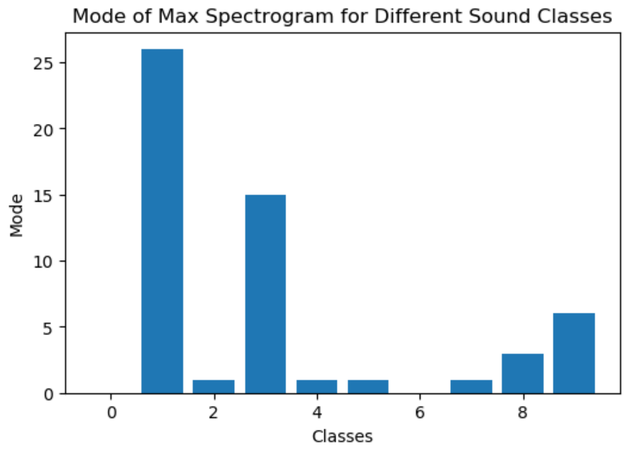


Figure 5: Mode of maximum MEL-spectogram value for the 10 classes

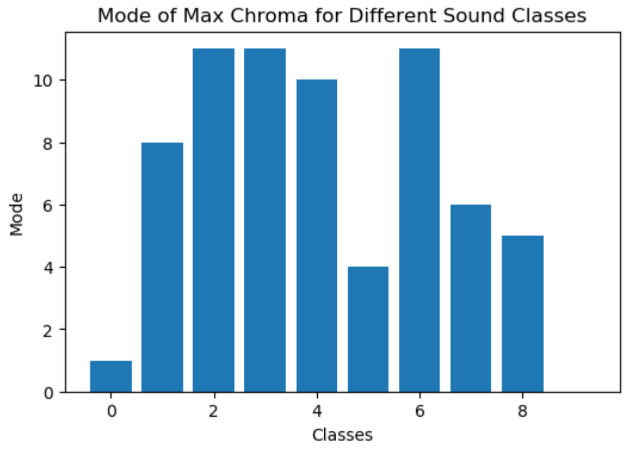


Figure 6: Mode of maximum Chroma feature value for the 10 classes

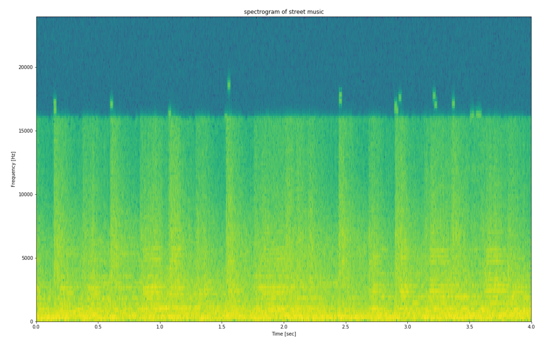


Figure 7: 2D spectrogram of street music audio sample

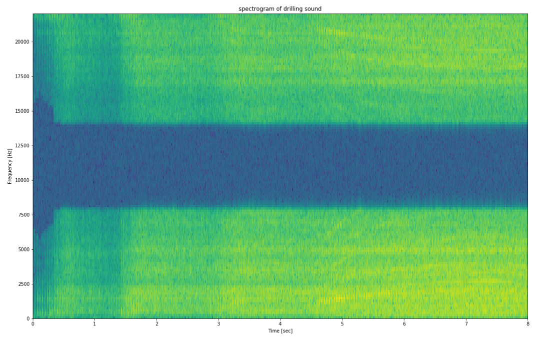


Figure 8: 2D spectrogram of drilling sound audio sample

lutional Neural Networks(CNN) to classify our dataset by feeding spectogram images directly(2D). We also implemented CNN by feeding it with 1 dimensional features obtained by taking mean of 2 Dimensional features over time. CNN with 1D features is implemented mainly to compare it's performance with the other linear models SVM and LR.

3.1 Logistic Regression

To implement this linear model as a multi-label classifer, we used the sklearn library's logistic regression. The stopping criteria set is a iteration limit of 10000. Using a penalty term of 1 and L2 regularization, sklearn's logistic regression gave an accuracy of 75 percent on original test dataset. We also ran logistic regression on augmented dataset which is double the size of our original dataset and achieved an accuracy of 71 percent, a little less than what we got with original dataset.

We also performed cross validation on the training data by dividing the training dataset into 10 folds. Iteratively using 9 of the 10 folds as training set and the other set as test dataset, we ran logistic regression algorithm on the original dataset using sklearn's library. This again gave us 75 percent accuracy on an average for all the 10 iterations. We can clearly see that we are not over-fitting the training data as we have similar accuracy on both training data and test data. Again, Cross validation gave us an average accuracy of 69 percent on augmented dataset, which is the same as the test accuracy on augmented test dataset confirming no over-fitting of training data.

Figure 9 shows the confusion matrix of our logistic regression model on the original dataset. Figure 10 shows the confusion matrix for logistic regression model with the augmented dataset.

Parameter Tuning: For our logistic regression model, we also tried to tune the parameters. We used a randomized search over different parameter values for maximum number of iterations, inverse of regularization strength i.e. 'C' and different optimization algorithms including 'liblinear', 'sag', 'saga', 'lbfgs' solvers from the 'sklearn' library. Despite using different parameter values, we observed that the performance of our model stayed consistent. Therefore, parameter tuning did not affect the accuracies of our model much. The reported accuracy values are when we tested our model with the default parameters for optimization algorithm and 'C' value with a maximum iteration limit of 10000.

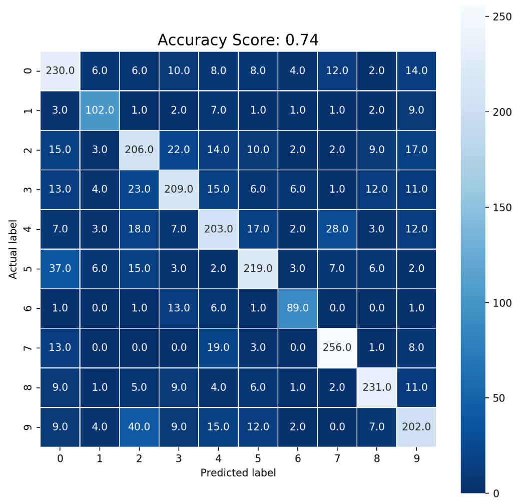


Figure 9: Confusion matrix of Logistic Regression model on Original Dataset

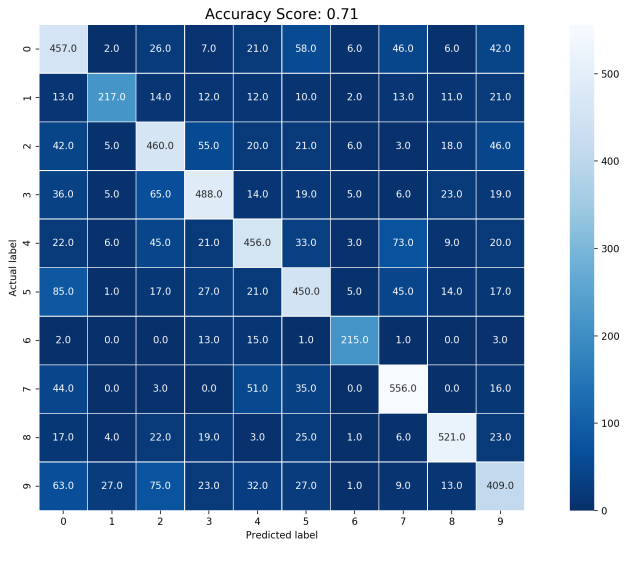


Figure 10: Confusion matrix of Logistic Regression model on Augmented Dataset

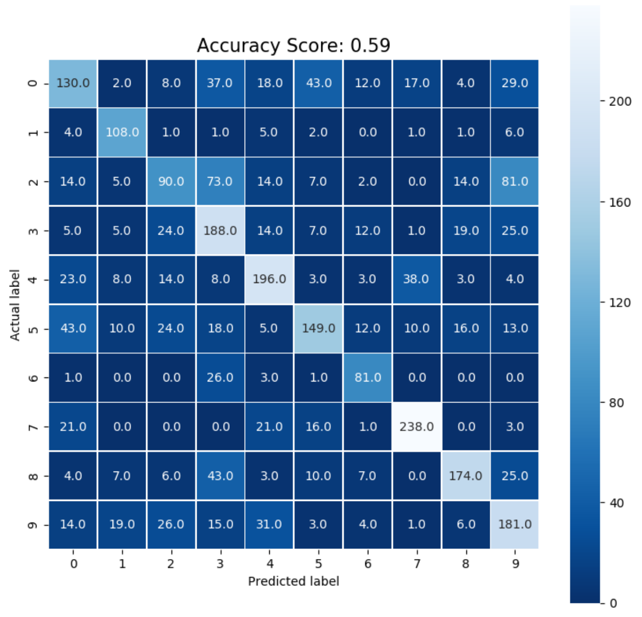


Figure 11: Confusion matrix of SVM model on Original Dataset

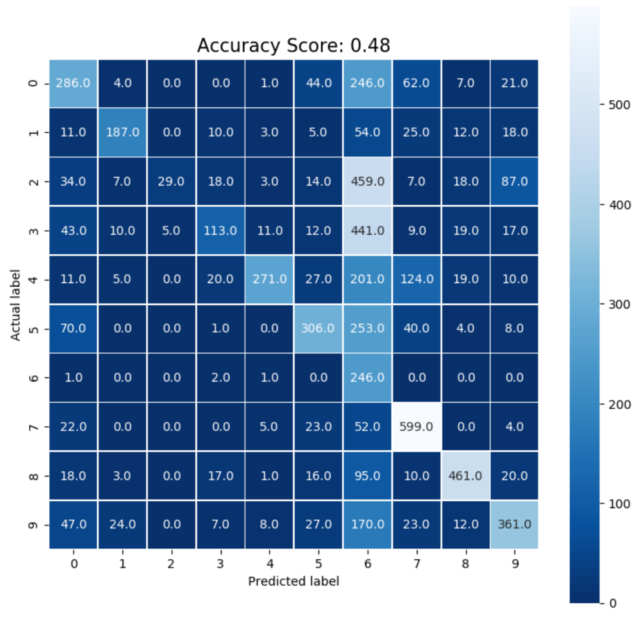


Figure 12: Confusion matrix of SVM model on Augmented Dataset

3.2 Support Vector Machine

To implement SVM as a multi-label classifer, we used the sklearn library's svc with kernel parameter set as linear. The stopping criteria is L2 normalization set as 0.001. Using a penalty term of 1 and no limit set on maximum iterations, sklearn's svm.svc gave us an accuracy of 81 percent on original test dataset. We also ran SVM on augmented dataset which is double the size of original dataset and achieved an accuracy of 78 percent, a little less than what we got with original dataset. We observed a drop in accuracy when augmented dataset is used which is similar to what we saw in LR model.

Similar to LR, we also performed cross validation on the training data by dividing the training datset into 10 folds. Iteratively using 9 of the 10 folds as training set and the other set as test dataset, we ran SVM algorithm on the original dataset using sklearn's svc library. This again gave us 82 percent accuracy on an average for all the 10 iterations. We can clearly see that we are not over-fitting the training data as we have similar accuracy on both training data and test data. Again, Cross validation gave us an average accuracy of 76 percent on augmented dataset, which is the same as the test accuracy on augmented test dataset confirming no over-fitting of training data.

Figure 11 is the confusion matrix for SVM on original dataset. The confusion matrix on the augmented dataset is shown in figure 12.

Parameter Tuning We tried to set an iteration limit while running SVM algorithm on both original and augmented dataset and observed that there is indeed a heavy drop in accuracy. So we let it run until it converges instead of putting a stop limit as L2 normalizer.

3.3 Convolutional Neural Networks

We also implemented a 2D Convolutional Neural Network to test the classification accuracy on the 2D MEL spectrogram features. The basic idea is that several patterns exist in the 2D MEL spectrogram for different sound classes. These patterns might not be captured by the 1D features. A 2D CNN would be trained to recognize these patterns and extract the features that differentiate the class labels.

We implemented the 2D CNN classifier using Keras library with Tensorflow backend. We tuned the parameters and model structure on the training data. We used a sequential model with two 2D CNN layers, each with 32 filters of size 3×3. We used ReLU as the activation function. After each CNN layer, we used a 2D max pooling layer of size 2. The purpose of the max pooling layers is to reduce the dimensions and consequently the computational cost. It also helps with reducing the model overfitting.

On top of the CNN layers, we add two dense layers. The first has 128 dimensions, followed by a drop out layer with ratio 0.5. By dropping out some of the neurons during training, we again reduce the model overfitting on the training data. The last dense layers has 10 units which is equal to the number of classes. A Softmax activation function is used, which outputs a probability distribution for the class labels. A visualization for the model is provided in Figure 13.

For comparison, we also implemented a 1D CNN classifier and train it on the 1D features. The structure of the model is similar to the 2D model, except that we modify the dimensions of the layers to fit the 1D features.

The results of the two classifiers are shown in Figure 16. For augmentation in 2D, we used a 50% sample for resources constraints. The results show that the 2D model achieves 0.6 accuracy on the original dataset and 0.72 after adding the augmented data. The 1D model obtains higher accuracy of 0.74 on the original data. The accuracy of the 1D model was almost the same after adding the augmented data. We justify the lower accuracy of the 2D model by its slow convergence due to high dimensions. We also observe that the accuracy of the 2D model was boosted by adding the aug-

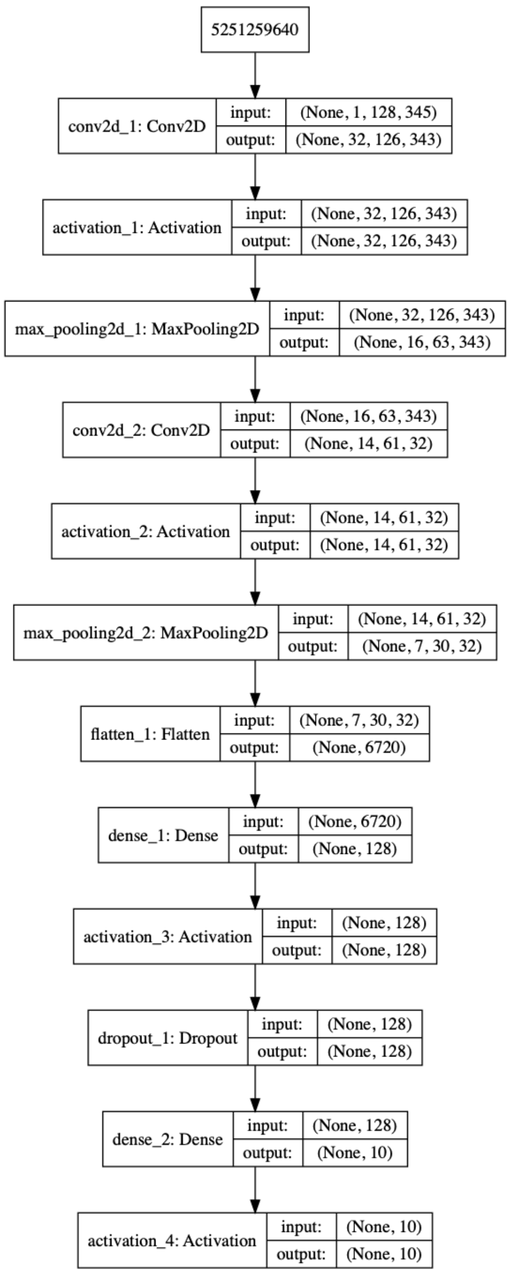


Figure 13: Visualization of the multi-layer 2D CNN model

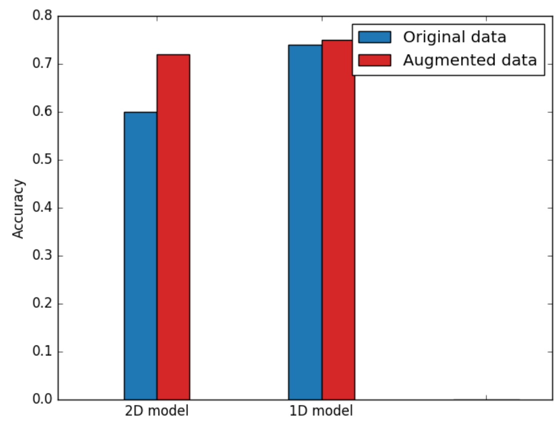


Figure 14: Accuracy of CNN 1D and 2D models on original and augmented dataset

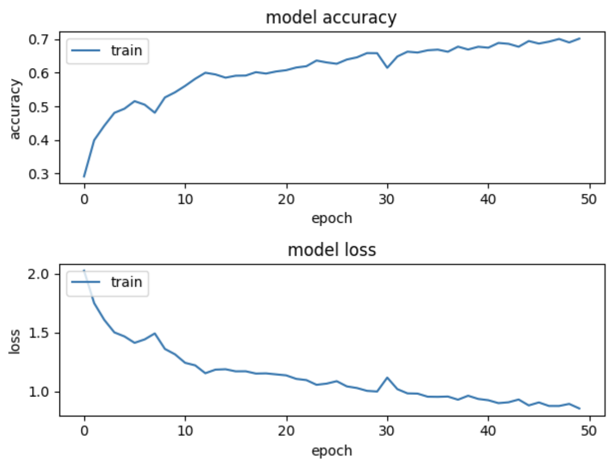


Figure 15: Training accuracy and loss metric at successive epochs for the 1D CNN model

mentation data, which shows that it is more capable of handling audio distortions, so it can better generalize on real world sound data.

4. INSIGHTS AND FUTURE WORK

4.1 Evaluation

The goal of our project was to classify urban sounds within our dataset using various data mining algorithms. In our proposal, we mentioned that our evaluation would be mainly based on determining the performance of the three algorithms i.e. logistic regression, support vector machine and convolutional neural network (CNN). For this purpose, our first step was to determine the performance of our models on the test set sampled from the original dataset as well as the original data combined with augmented data. We got an accuracy of 75%, 81%, 60% and 74.6% from logistic regression, SVM, CNN with 2D features and CNN with 1D features respectively. For the augmented data, the accuracy on the test set was 71%, 78%, 72% and 75% from logistic

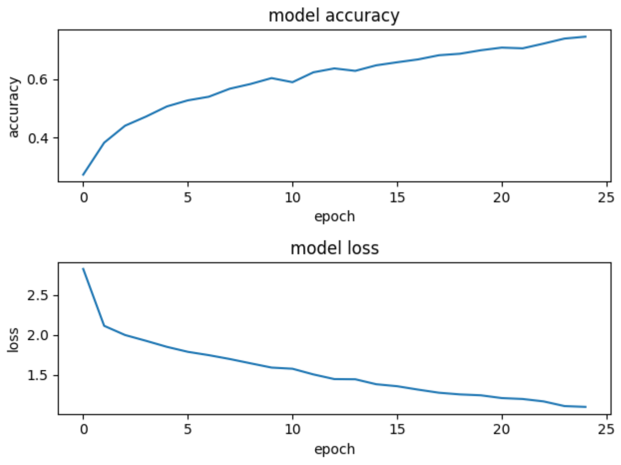


Figure 16: Training accuracy and loss metric at successive epochs for the 2D CNN model

regression, SVM, CNN with 2D features and CNN with 1D features respectively. Apart from this, we also performed 10-fold cross validation on our two datasets for the linear classifiers. The reason for performing cross validation was to determine if our models were over-fitting on the training data. For the 10 folds, we got an accuracy of 75%, 83% for logistic regression and SVM respectively on the original dataset. For the augmented data, the cross validation accuracies were 69% and 76% for the two classifiers. These results confirmed that our models were not overfitting.

4.2 Observations

Running multiple algorithms on 1-Dimensional and 2-Dimensional features gave us more understanding on how these algorithms behave with different sets of data. We observed how increasing the size of the dataset would impact the performance of each algorithm. Mentioned below are our observations and conclusions from each algorithm we implemented.

Logistic Regression

- Parameter tuning did not help much in case of Logistic Regression. The only parameter we set apart from the default parameter values is iteration limit. We in fact observed a drop in accuracy when we tried to modify parameters.

- We saw a 4 percent drop in accuracy of augmented dataset when compared to the original dataset. This made us realize linear models might not perform as well as we want them to be when the data is noisy.

Support Vector Machine

- Similar to the other linear model Logistic Regression, parameter tuning on scikit's SVM actually worsened the algorithm's performance. Setting a limit to 10000 on maximum iterations needed for the algorithm to stop, gave us an accuracy of 51 percent which is a major drop. So we proceeded with not setting any limit on the iterations and we just let it run until it converges, which definitely gave us a much better performance with an accuracy of 81 percent.

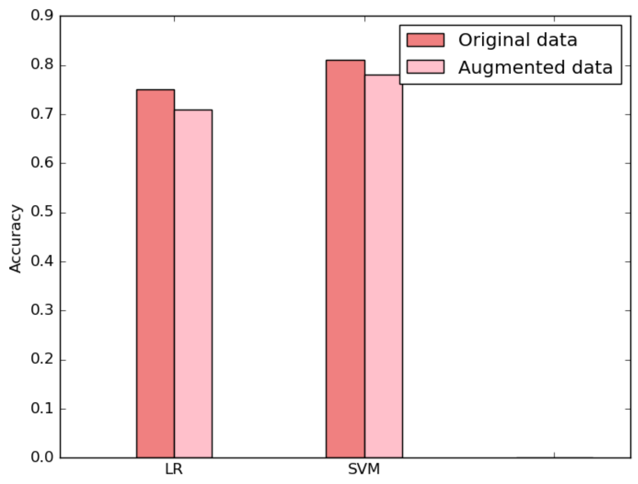


Figure 17: Accuracy of LR and SVM models for Original and AUgmented Datasets

- For SVM as well, we observed that accuracy dropped for augmented data when compared to the original dataset. From this observation, we concluded that data augmentation does not really help linear models much. Linear models are better off with classifying clean audio files.

Comparison between LR and SVM models

- On comparing accuracies obtained from running Logistic Regression(LR) and Support Vector Machine(SVM) models post parameter tuning, we observed that overall our Support Vector Machine model performed better than Logistic Regression model.

- We observed that both models performed better on original data when compared to the augmented data which shows the how sensitive both these linear models are to noise in the sound waves.

Figure 17 depicts both the above observed behaviours.

Convolutional Neural Networks with 2D features

- The 2D CNN model gives different accuracy with different parameters and different model structure. Due to the high dimensionality, the convergence time is slow. Also due to resources constraints 2 CNN layers were added. The model accuracy could be improved by adding more layers or retraining more complex models such as VGG for environmental sounds.

- We also observe that adding augmented data boosted the accuracy of 2D CNN model, which shows the capability of this model to handle distortions and noise in data. It also shows that due to high dimensionality, this model requires more data to give the desired accuracy, so there is a good chance to further improve the accuracy by adding more data.

- We believe that this model would better generalize in real world scenarios.

Convolutional Neural Networks with 1D features

- The 1D CNN model shows better accuracy than the 2D model on the original data since it converges faster than the 2D model. For augmentation data, the two models have close accuracy values.

- Unlike the 2D model adding more data did not improve the accuracy, which shows that the 1D model accuracy is unlikely to improve above 75%.

4.3 Future work

Our observations show that the classification of Urban Sounds could be improved by the following future directions:

- Adding more augmented data or adding other sound datasets to get better generalization and improve the deep learning accuracy.

- Adding more convolution layers to the 2D model and training with more powerful resources for convergence and tuning.

- Retraining or fine tuning existing large models such as VGG for environmental sounds.

5. CONTRIBUTIONS

Table 1 details the contributions of each individual team member. Apart from the below listed details, all of our team members put equal efforts in documenting project proposal, Final project report and preparing slides for project pitch and Final project presentation.

Team Member	Contribution
Reham Aburas	Feature extraction from the audio files obtained from kaggle. CNN implementation, Cross Validation, Parameter tuning along with Aly Shehata.
Aly Shehata	Augmented dataset generation from Original Dataset. CNN implementation, Parameter-tuning along with Reham. Project pitch presentation. Final project presentation.
Habiba Farrukh	Data visualization along with Pavani. LR implementation,Parameter-Tuning, Cross Validation. Final project presentation.
Pavani Guttula	Data visualization along with Habiba. SVM implementation,Parameter-Tuning, Cross Validation.

Table 1: Contribution of Tasks

6. REFERENCES

[1] D. M. Agrawal, H. B. Sailor, M. H. Soni, and H. A. Patil. Novel teo-based gammatone features for environmental sound classification. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 1809–1813. IEEE, 2017.

[2] Y. Aytar, C. Vondrick, and A. Torralba. Soundnet: Learning sound representations from unlabeled video. In *Advances in neural information processing systems*, pages 892–900, 2016.

[3] J. T. Geiger and K. Helwani. Improving event detection for audio surveillance using gabor filterbank features. *2015 23rd European Signal Processing Conference (EUSIPCO)*, pages 714–718, 2015.

[4] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, et al. Cnn architectures for large-scale audio classification. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 131–135. IEEE, 2017.

[5] K. J. Piczak. Environmental sound classification with convolutional neural networks. *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, 2015.

[6] J. Salamon and J. P. Bello. Feature learning with deep scattering for urban sound analysis. *2015 23rd European Signal Processing Conference (EUSIPCO)*, pages 724–728, 2015.

[7] J. Salamon and J. P. Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, 24:279–283, 2017.

[8] J. Salamon, C. Jacoby, and J. P. Bello. A dataset and taxonomy for urban sound research. In *ACM Multimedia*, 2014.

[9] J.-C. Wang, J.-F. Wang, K. W. He, and C.-S. Hsu. Environmental sound classification using hybrid svm/knn classifier and mpeg-7 audio low-level descriptor. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 1731–1735. IEEE, 2006.

[10] H. Wu and J. M. Mendel. Classification of battlefield ground vehicles using acoustic features and fuzzy logic rule-based classifiers. *IEEE Transactions on Fuzzy Systems*, 15:56–72, 2007.