NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTION, AFFILIATED TO VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM, APPROVED BY AICTE & GOVT.OF KARNATAKA

ML PROJECT REPORT
On
Urban Sound Classification

Submitted in partial fulfillment of the requirement for the award of Degree of

Bachelor of Engineering
in
Computer Science and Engineering

Submitted by:
Umang Kumar Harlalka (1NT18CS211)
Mukesh Goit (1NT18CS198)
Abhinash Prasad Sah (1NT18CS189)

Under the Guidance of
Dr. Vani V.
Professor, Dept. of CS&E, NMIT

NITTE EDUCATION TRUST

Department of Computer Science and Engineering

(Accredited by NBA Tier-1)

2020-21

NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTION, AFFILIATED TO VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM
, APPROVED BY AICTE & GOVT.OF KARNATAKA)

Department of Computer Science and Engineering

(Accredited by NBA Tier-1)

CERTIFICATE

This is to certify that Baby Cry Classification is an authentic work carried out by **Mukesh Goit (1NT18CS198), Umang Kumar Harlalka (1NT18CS211) , Abhinash Prasad Sah (1NT18CS189)** bonafide students of **Nitte Meenakshi Institute of Technology**, Bangalore in partial fulfillment for the award of the degree of ***Bachelor of Engineering*** in COMPUTER SCIENCE AND ENGINEERING of Visvesvaraya Technological University, Belagavi during the academic year ***2020-2021.*** It is certified that all corrections and suggestions indicated during the internal assessment have been incorporated in the report.

Internal Guide **Signature of H.O.D**

Dr. Vani V	Dr. Sarojadevi H.
Professor, Dept. CSE,	Professor, Head, Dept. CSE,
NMIT Bangalore	NMIT Bangalore

DECLARATION

We are hereby declare that;

The project work is our original work. This Project work has not been submitted for the award of any degree or examination at any other university/College/Institute. This Project Work does not contain other persons' data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons. This Project Work does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then: their words have been re-written but the general information attributed to them has been referenced;where their exact words have been used, their writing has been placed inside quotation marks, and referenced. This Project Work does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the thesis and in the References sections.

Name	USN
Mukesh Goit	1NT18CS198
Umang Kumar Harlalka	1NT18CS211
Abhinash Prasad Sah	1NT18CS189

Date: 17/01/2022

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our effort with success. I express my sincere gratitude to our Principal **Dr. H. C. Nagaraj**, Nitte Meenakshi Institute of Technology for providing facilities.

We wish to thank our HoD**, Dr.Sarojadevi H.** for the excellent environment created to further educational growth in our college. We also thank him for the invaluable guidance provided which has helped in the creation of a better project.

I hereby like to thank our **Dr. Vani V., Professor**, Department of Computer Science & Engineering on his periodic inspection, time to time evaluation of the project and help to bring the project to the present form.

Thanks to our Departmental Project coordinators. We also thank all our friends, teaching and non-teaching staff at NMIT, Bangalore, for all the direct and indirect help provided in the completion of the project.

Name	USN
Mukesh Goit	1NT18CS198
Umang Kumar Harlalka	1NT18CS211
Abhinash Prasad Sah	1NT18CS189

Date: 18/01/2022

ABSTRACT

This learning activity project is a simple audio classification model based on machine learning and deep learning. We address the problem of classifying the type of sound based on short audio signals and their generated spectrograms, from labeled sounds belonging to 10 different classes during model training. In order to meet this challenge, we use a model based on Convolutional Neural Network (CNN),Artificial neural network(ANN) or KNN. The audio was processed with Mel-frequency Cepstral Coefficients (MFCC) into what's commonly known as Mel spectrograms, and hence, was transformed into an image. Our final CNN model achieved 91% accuracy on the testing datasets.

Table of Contents

INTRODUCTION

Sounds are the part of our daily lives, ranging from the conversations we have when interacting with people, the music we listen to, and all the other environmental sounds that we hear on a daily basis such as a car driving past, the patter of rain, or any other kind of background noise. Sound classification is a constantly developing area of research and is at the heart of a lot of advanced technologies including automatic speech recognition systems, security systems, and text-to-speech applications. There are numerous applications which are continuously improving like video indexing and content based retrieval, speaker and sound identification use cases and potential security applications. Moreover, we know convolutional neural networks (CNNs) are widely used in image classification and they achieve significantly high accuracy, so we try to use this technique in a seemingly different field of audio classification, where discrete sounds happen over time. This project aims to build a deep learning powered audio classifier. The basic underlying problem is to be able to manipulate audio data and build a model to classify sounds. The project aims to leverage progress achieved in the deep learning field for speaker identification and recognition problems in order to perform accurately and improve to effectively assist users. The input of the algorithm used in this project is taken from the database of the Urban Sound Classification Challenge, which are short audio samples commonly found in an urban environment like children playing, street music, a car engine etc. The samples must first be pre-processed in order to extract the MFCC features of each audio signal. The MFCC features vector is used as an input for the CNN model for classification and to generate predictions. The neural network outputs a vector with the probabilities of the sample belonging to each of the registered class. This vector is used to generate a prediction of the class of the sound, guessing for the one with the highest probability.

Data source and data quality

We will use the dataset from the Kaggle. The dataset is obtained from the following link – https://www.kaggle.com/chrisfilo/urbansound8k.
The dataset is a present wav file and contains 8732 entries of unique values.

Data Pre-processing

In our approach, the various urban sound signals are taken as input, and mel frequency cepstral coefficients (MFCCs) are deployed to concisely describe the overall shape of a spectral envelope. The features are extracted from the image using a deep convolutional neural network.

Machine Learning Models

Artificial Neural Network

An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal to other neurons. An artificial neuron receives a signal then processes it and can signal neurons connected to it. The "signal" at a connection is a real number, and the output of each neuron is computed by some non-linear function of the sum of its inputs. The connections are called *edges*. Neurons and edges typically have a *weight* that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Neurons may have a threshold such that a signal is sent only if the aggregate signal crosses that threshold. Typically, neurons are aggregated into layers. Different layers may perform different transformations on their inputs. Signals travel from the first layer (the input layer), to the last layer (the output layer), possibly after traversing the layers multiple times.

CNN

Convolutional neural networks are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs. They have three main types of layers, which are: Convolutional layer, Pooling layer and fully connected (FC) layer. The convolutional layer is the first layer of a convolutional network. While convolutional layers can be followed by additional convolutional layers or pooling layers, the fully connected layer is the final layer. With each layer, the CNN increases in its complexity, identifying greater portions of the data. Earlier layers focus on simple features.

Mel-frequency cepstral coefficients (MFCCs)

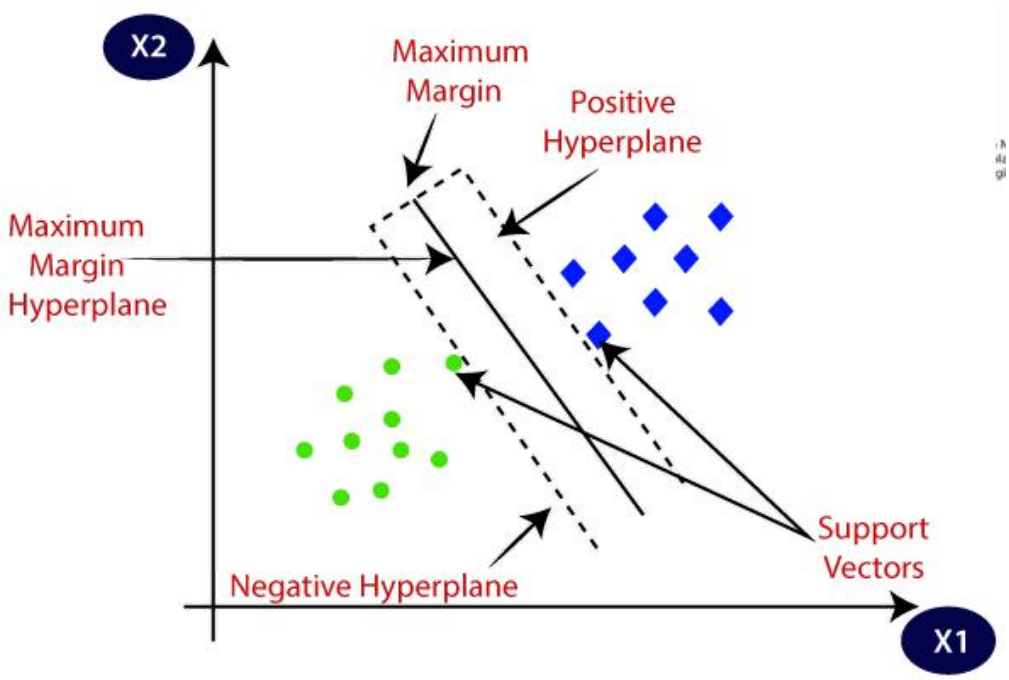
In sound processing, the **mel-frequency cepstrum** (**MFC**) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency.

Mel-frequency cepstral coefficients (**MFCCs**) are coefficients that collectively make up an MFC.They are derived from a type of cepstral representation of the audio clip (a nonlinear "spectrum-of-a-spectrum"). The difference between the cepstrum and the mel-frequency cepstrum is that in the MFC, the frequency bands are equally spaced on the mel scale, which approximates the human auditory system's response more closely than the linearly-spaced frequency bands used in the normal spectrum. This frequency warping can allow for better representation of sound, for example, in audio compression.

Support Vector Machine

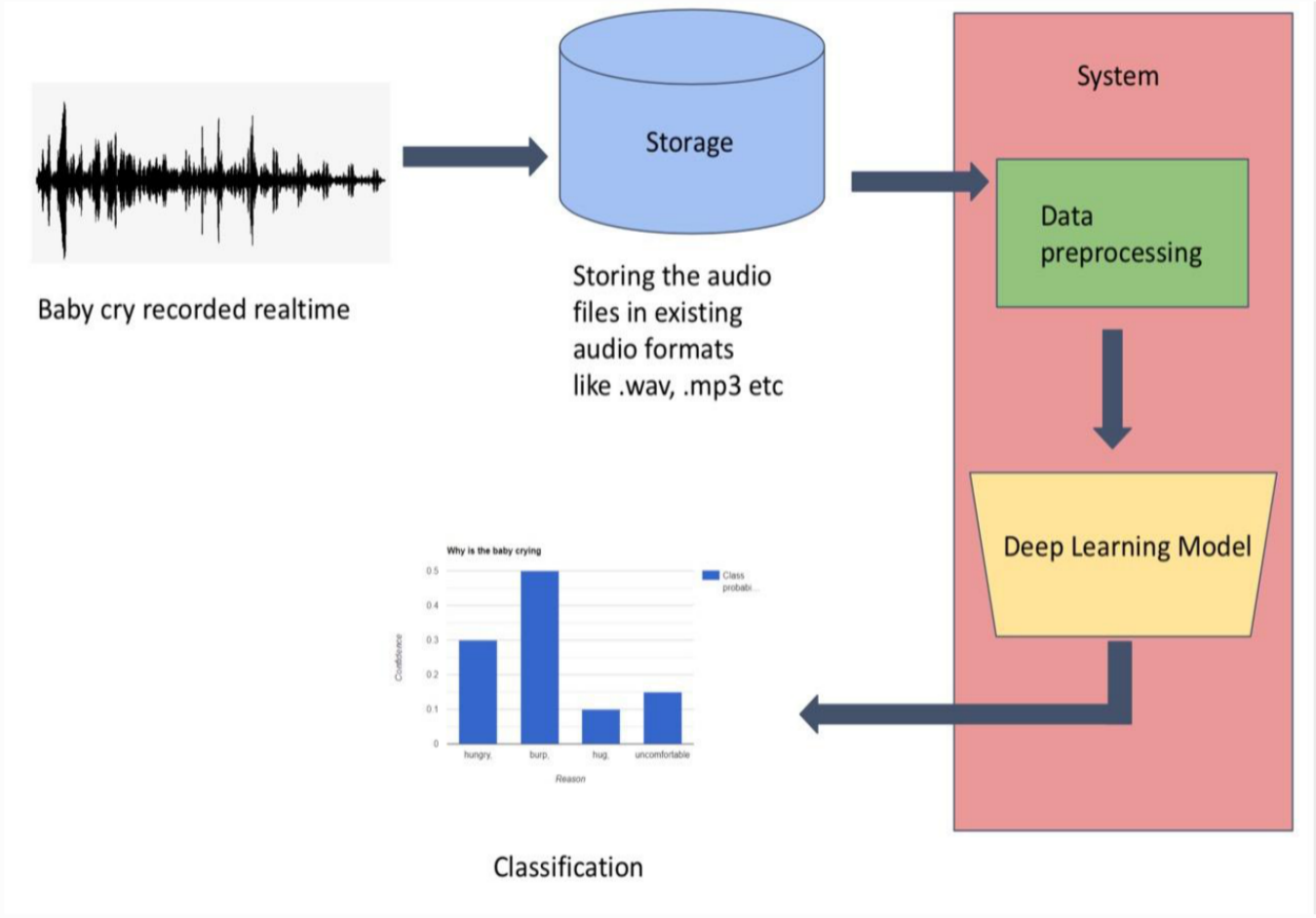
SVM is a supervised machine learning technique that may be used for both classification and regression. Though we might also argue regression difficulties, categorization is the best fit. The goal of the SVM method is to discover a hyperplane in an N-dimensional space that categorises

data points clearly. In comparison to other non-linear classifiers such as neural networks, SVMs are intended to operate well with few samples and high-dimensional data.


X2
Maximum
Margin
Positive
Hyperplane
Maximum
Margin
Hyperplane
Support
Vectors
Negative Hyperplane
X1

DESIGN

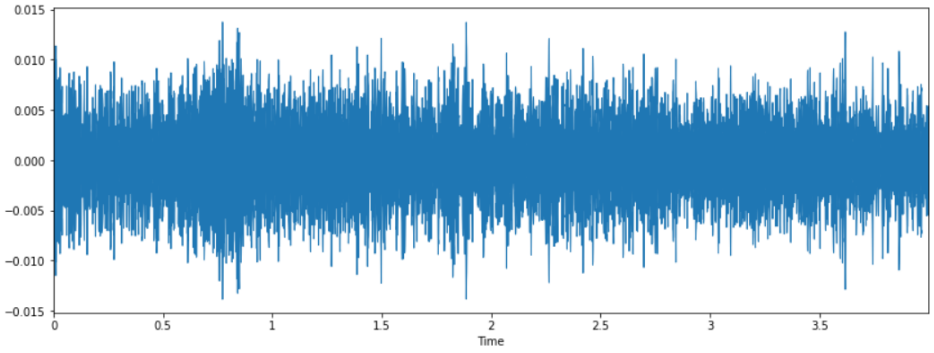
R


Baby cry recorded realtime
Storage
Storing the audio files in existing audio formats like .wav, .mp3 etc
System
Data preprocessing
Deep Learning Model
Why is the baby crying
Confidence
Reason
hungry
burp
hug
uncomfortable
Classification

SNAPSHOTS OF IMPLEMENTATION

In [21]:
```
### Sound
filename='UrbanSound8K/100263-2-0-3.wav'
plt.figure(figsize=(14,5))
data,sample_rate=librosa.load(filename)
librosa.display.waveplot(data,sr=sample_rate)
ipd.Audio(filename)
```

Out[21]:



In [14]:
```
sample_rate
```

Out[14]:
```
22050
```

In [74]:
```
mfccs = librosa.feature.mfcc(y=librosa_audio_data, sr=librosa_sample_rate, n_mfcc=40)
print(mfccs.shape)
```

```
(40, 173)
```

In [75]:
```
mfccs
```

Out[75]:
```
array([[-4.45197296e+02, -4.47219299e+02, -4.49755127e+02, ...,
        -4.77412781e+02, -4.74241730e+02, -4.82704987e+02],
       [ 1.12513969e+02,  1.11970383e+02,  1.12244164e+02, ...,
         1.12045395e+02,  1.12248581e+02,  1.05560913e+02],
       [-1.58260956e+01, -2.30021858e+01, -3.12500191e+01, ...,
        -9.15441895e+00, -1.03232269e+01, -7.39410734e+00],
       ...,
       [-7.82766485e+00, -5.03880405e+00, -4.48165369e+00, ...,
        -1.90692782e-01,  4.34143972e+00,  1.00339069e+01],
       [-1.91763425e+00, -8.02737713e-01, -1.20930457e+00, ...,
        -1.23640239e-01,  2.90504694e-02,  9.22017097e-01],
       [-3.88130605e-01,  3.09317827e-01,  6.72155714e+00, ...,
        -2.33736587e+00, -4.25179911e+00, -2.31322765e+00]], dtype=float32)
```

In [76]:
```
#### Extracting MFCC's For every audio file
import pandas as pd
import os
import librosa

audio_dataset_path='UrbanSound8K/audio/'
metadata=pd.read_csv('UrbanSound8K/metadata/UrbanSound8K.csv')
metadata.head()
```

Out[76]:

	slice_file_name	fsID	start	end	salience	fold	classID	class
0	100032-3-0-0.wav	100032	0.0	0.317551	1	5	3	dog_bark
1	100263-2-0-117.wav	100263	58.5	62.500000	1	5	2	children_playing
2	100263-2-0-121.wav	100263	60.5	64.500000	1	5	2	children_playing
3	100263-2-0-126.wav	100263	63.0	67.000000	1	5	2	children_playing
4	100263-2-0-137.wav	100263	68.5	72.500000	1	5	2	children_playing

Fig. Feature Extraction

```python
def get_features(path):
    sample, sr = librosa.load(path, sr=16000)
    # magnitude =
    avg = np.mean(sample, keepdims=True)[0]

    rate, signal = scipy.io.wavfile.read(path)

    # lfcc = c(signal)
    lfccs = lfcc(signal)

    lfccs = lfccs.flatten()
    return [
        # magnitude,
        avg,
        # variance,
        # bandwidth,
        # centroid,
        # rolloff,
        # rms,
        # zero_crossing_rate,
        # peak,
        # valley,
        # contrast,
        # pitch,
        # formant,
        # lfcc,
        lfccs
    ]
```

```
[ ]  avg,lfccs = get_features('/content/artifacts/baby_cry_split_with_ambient:v4/hug/hug_0_1.wav')
```

In [132…
```
import tensorflow as tf
print(tf.__version__)
```

```
2.3.1
```

In [149…
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Dropout,Activation,Flatten
from tensorflow.keras.optimizers import Adam
from sklearn import metrics
```

In [150…
```
### No of classes
num_labels=y.shape[1]
```

In [ ]:
```
```

In [151…
```
model=Sequential()
###first Layer
model.add(Dense(100,input_shape=(40,)))
model.add(Activation('relu'))
model.add(Dropout(0.5))
###second Layer
model.add(Dense(200))
model.add(Activation('relu'))
model.add(Dropout(0.5))
###third Layer
model.add(Dense(100))
model.add(Activation('relu'))
model.add(Dropout(0.5))

###final Layer
model.add(Dense(num_labels))
model.add(Activation('softmax'))
```

In [152…
```
model.summary()
```

```
Model: "sequential_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
Epoch 00097: val_loss did not improve from 0.63103
219/219 [==============================] - 0s 2ms/step - loss: 0.8288 - accuracy: 0.7288 - val_loss: 0.6529 - val_accuracy: 0.7985
Epoch 98/100
191/219 [=========================>....] - ETA: 0s - loss: 0.8121 - accuracy: 0.7281
Epoch 00098: val_loss did not improve from 0.63103
219/219 [==============================] - 0s 2ms/step - loss: 0.8153 - accuracy: 0.7273 - val_loss: 0.6450 - val_accuracy: 0.7934
Epoch 99/100
191/219 [=========================>....] - ETA: 0s - loss: 0.8151 - accuracy: 0.7307 ETA: 0s - loss: 0.8208 - accura
Epoch 00099: val_loss did not improve from 0.63103
219/219 [==============================] - 0s 2ms/step - loss: 0.8212 - accuracy: 0.7274 - val_loss: 0.6649 - val_accuracy: 0.7997
Epoch 100/100
202/219 [==========================>...] - ETA: 0s - loss: 0.8116 - accuracy: 0.7293
Epoch 00100: val_loss did not improve from 0.63103
219/219 [==============================] - 0s 2ms/step - loss: 0.8071 - accuracy: 0.7303 - val_loss: 0.6363 - val_accuracy: 0.8002
Training completed in time:  0:00:45.200242
```

In [182…
```
test_accuracy=model.evaluate(X_test,y_test,verbose=0)
print(test_accuracy[1])
```

```
0.8002289533615112
```

In [156…
```
prediction_feature.shape
```

Out[156…
```
(1, 40)
```

In [190…
```
X_test[1]
```

Out[190…
```
array([-466.1843    ,    1.5388278 ,  -34.397358  ,   35.715336  ,
       -15.166929  ,  -18.850813  ,   -0.7415805 ,  -15.99989   ,
       -21.354332  ,    7.6506834 ,  -29.031452  ,  -19.142824  ,
        -2.6798913 ,   -8.466884  ,  -14.7660475 ,   -7.004778  ,
        -7.103754  ,    8.887754  ,   14.911873  ,   21.47102   ,
        21.336624  ,    0.9169518 ,  -18.795404  ,   -5.001721  ,
        -0.70152664,    2.91399   ,   -6.7105994 ,  -16.638536  ,
        -9.821647  ,   12.8619585 ,    0.6552978 ,  -23.953394  ,
       -15.200551  ,    9.21079   ,   10.419799  ,   -0.57916117,
        -1.2440346 ,   17.722294  ,   13.837573  ,   -5.164349  ],
      dtype=float32)
```

In [185…
```
model.predict_classes(X_test)
```

```
array([5, 3, 4, ..., 1, 2, 2], dtype=int64)
```

Fig. ANN Implementation

```
from sklearn import svm
clf = svm.SVC()

#Will take some time
clf.fit(X_train, y_train)
confidence = clf.score(X_test, y_test)
print(confidence)
```

```
0.5094142259414226
```

```
pred = clf.predict(X_test)
```

```
from sklearn.metrics import classification_report, confusion_matrix
conf_matrix = confusion_matrix(y_test, pred)
print(conf_matrix)
```

```
[[477   0   0   0   0   0]
 [ 99   0   0   0   0   0]
 [105   0   5   1   0   0]
 [102   0   0   1   0   0]
 [ 90   0   0   0   0   0]
 [ 72   0   0   0   0   4]]
```

```
print(classification_report(y_test,pred))
```

```
              precision    recall  f1-score   support

     ambient       0.50      1.00      0.67       477
       awake       0.00      0.00      0.00        99
         hug       1.00      0.05      0.09       111
      hungry       0.50      0.01      0.02       103
      sleepy       0.00      0.00      0.00        90
uncomfortable       1.00      0.05      0.10        76
```

Fig. SVM Implementation

Results

ANN – Accuracy 80%

```
Epoch 00097: val_loss did not improve from 0.63103
219/219 [==============================] - 0s 2ms/step - loss: 0.8288 - accuracy: 0.7288 - val_loss: 0.6529 - val_accuracy: 0.7985
Epoch 98/100
191/219 [=========================>....] - ETA: 0s - loss: 0.8121 - accuracy: 0.7281
Epoch 00098: val_loss did not improve from 0.63103
219/219 [==============================] - 0s 2ms/step - loss: 0.8153 - accuracy: 0.7273 - val_loss: 0.6450 - val_accuracy: 0.7934
Epoch 99/100
191/219 [=========================>....] - ETA: 0s - loss: 0.8151 - accuracy: 0.7307 ETA: 0s - loss: 0.8208 - accura
Epoch 00099: val_loss did not improve from 0.63103
219/219 [==============================] - 0s 2ms/step - loss: 0.8212 - accuracy: 0.7274 - val_loss: 0.6649 - val_accuracy: 0.7997
Epoch 100/100
202/219 [==========================>...] - ETA: 0s - loss: 0.8116 - accuracy: 0.7293
Epoch 00100: val_loss did not improve from 0.63103
219/219 [==============================] - 0s 2ms/step - loss: 0.8071 - accuracy: 0.7303 - val_loss: 0.6363 - val_accuracy: 0.8002
Training completed in time:  0:00:45.200242
```

In [182…

```
test_accuracy=model.evaluate(X_test,y_test,verbose=0)
print(test_accuracy[1])
```

```
0.8002289533615112
```

In [156…

```
prediction_feature.shape
```

Out[156…

```
(1, 40)
```

In [190…

```
X_test[1]
```

Out[190…

```
array([-466.1843    ,    1.5388278 ,  -34.397358  ,   35.715336  ,
        -15.166929  ,  -18.850813  ,   -0.7415805 ,  -15.99989   ,
        -21.354332  ,    7.6506834 ,  -29.031452  ,  -19.142824  ,
         -2.6798913 ,   -8.466884  ,  -14.7660475 ,   -7.004778  ,
         -7.103754  ,    8.887754  ,   14.911873  ,   21.47102   ,
         21.336624  ,    0.9169518 ,  -18.795404  ,   -5.001721  ,
         -0.70152664,    2.91399   ,   -6.7105994 ,  -16.638536  ,
         -9.821647  ,   12.8619585 ,    0.6552978 ,  -23.953394  ,
        -15.200551  ,    9.21079   ,   10.419799  ,   -0.57916117,
         -1.2440346 ,   17.722294  ,   13.837573  ,   -5.164349  ],
      dtype=float32)
```

In [185…

```
model.predict_classes(X_test)
```

```
array([5, 3, 4, ..., 1, 2, 2], dtype=int64)
```

CONCLUSION

This report presents important studies in Urban sound analysis and categorization, giving details and resources for both researchers and medical professionals working in this field. It is demonstrated that the sound classification study is hampered by a lack of database resources. Large datasets with a variety of samples that meet the needs of deep neural networks are essential. To gain superior discriminating ability, the current trend in feature extraction is to build a mixed feature set that takes use of diverse domains. The relevant study findings suggest that combining characteristics can result in a significant improvement. Furthermore, novel neural network-based designs are becoming prevalent.

REFERENCES

1. https://ieeexplore.ieee.org/document/9358621
2. https://www.ibm.com/cloud/learn/convolutional-neural-networks/
3. https://www.kaggle.com/chrisfilo/urbansound8k
4. https://www.researchgate.net/publication/346659500_Urban_Sound_Classification_Using_Convolutional_Neural_Network_and_Long_Short_Term_Memory_Based_on_Multiple_Features