**Tahir Siddique**

**Sohail Amjad**



DEPARTMENT OF COMPUTER SCIENCES

COMSATS UNIVERSITY ISLAMABAD, VEHARI CAMPUS

VEHARI – PAKISTAN

SESSION 2018-2022

*Undertaken By:*

**Tahir Siddique**

Reg. No. CIIT/FA18-bSE-036/Vhr

**Sohail Amjad**

Reg. No. CIIT/FA18-bSE-063/Vhr

*Supervised By:*

**Muhammad Jawad Rafeeq**



A DISSERTATION SUBMITTED AS A PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELORS IN COMPUTER SCIENCE / SOFTWARE ENGINEERING

DEPARTMENT OF COMPUTER SCIENCES

COMSATS UNIVESITY ISLAMABAD, VEHARI CAMPUS

VEHARI – PAKISTAN

SESSION 2018-2022

**CERTIFICATE OF APPROVAL**

It is to certify that the final year project of BS (SE) “Object Sound Recognition” was developed by **Tahir Siddique(CIIT/FA18-BSE/ 036)** and **Sohail Amjad(CIIT/FA18-BSE -063)**  under the supervision of “M.Jawad Rafeeq” and that in their opinion; it is fully adequate, in scope and quality for the degree of Bachelors of Science in Computer Sciences.

---------------------------------------

**Supervisor**

---------------------------------------

**External Examiner**

---------------------------------------

**Head of Department**

**(Department of Computer Science)**

**DEDICATION**

To my Loving Parents and respected Teachers

**ACKNOWLEDGEMENT**

All praise to almighty Allah alone, the omnipotent, the most compassionate. We would like to thank God for blessing us with the strength, intelligence, and patience to complete this project.

We feel great pleasure in expressing our heartiest gratitude to our honorable teacher “Muhammad Jawad Rafiq Lecturer COMSATS University Islamabad, Vehari campus, for kind behavior, valuable suggestion, worth and keen supervision, sympathetic attitude towards completion of project. We express our sincere thanks to all respectable teachers, friends and faculty members of computer science department of COMSATS University Islamabad, Vehari campus.

We would also like to thank our families for their love and support. Prayers of our families are treasure for our life.

**PROJECT BRIEF**

PROJECT NAME Object Sound Recognition

ORGANIZATION NAME COMSATS University Islamabad, Vehari campus

OBJECTIVE detect sound for critical alert

UNDERTAKEN BY Tahir Siddique

Sohail Amjad

SUPERVISED BY M Jawad Rafeeq

Lecturer

Computer science Department

Comsats University Islamabad Vehari Campus

Started On 25 October 2021

Completed On /\* END DATE \*/

COMPUTER USED DELL, HP

SOURCE LANGUAGE Ubuntu, PYTHON 3.10

OPERATING SYSTEM Ubuntu, Windows 10

TOOLS USED Spyder**ABSTRACT**

In each part of human existence, sound assumes a significant part. From individual security to basic reconnaissance, a sound is a critical component to foster computerized frameworks for these fields. Not many frameworks are now on the lookout yet their proficiency is a concerned point for their execution, in actuality, situations. The learning abilities of Deep learning structures can be utilized to foster sound grouping frameworks to defeat proficiency issues of customary frameworks. Our point is to utilize profound learning networks for grouping natural sounds in view of the produced spectrograms of these sounds. We will utilize spectrogram pictures of natural sounds to prepare Convolutional Neural Network. To prepare and test CNN we will utilize Urban8kSounds Dataset.

Table of Contents

[CHAPTER 1: INTRODUCTION 10](#_Toc104882651)

[1.1 System Introduction 10](#_Toc104882652)

[1.2 Background of the System 10](#_Toc104882653)

[1.3 Objectives of the System 11](#_Toc104882654)

[1.4 Significance of the System 11](#_Toc104882655)

[CHAPTER 2: REQUIREMENT SPECIFICATIONS 12](#_Toc104882656)

[2.1 Product Scope 12](#_Toc104882658)

[2.2 Product Description 12](#_Toc104882659)

[2.2.1 Product Perspective 12](#_Toc104882660)

[2.2.2 Product Functionality 12](#_Toc104882661)

[2.2.3 Users and Characteristics 12](#_Toc104882662)

[2.2.4 Operating Environment 12](#_Toc104882663)

[2.3 Specific Requirements 13](#_Toc104882664)

[2.3.1 Functional Requirements 13](#_Toc104882665)

[2.3.2 Behavioral Requirements 13](#_Toc104882666)

[2.3.3 External Interface Requirements 14](#_Toc104882667)

[2.4 Non-functional Requirements 14](#_Toc104882668)

[2.4.1 Performance Requirements 14](#_Toc104882669)

[2.4.2 Safety and Security Requirements 15](#_Toc104882670)

[2.4.3 Software Quality Attributes 15](#_Toc104882671)

[CHAPTER 3: DESIGN SPECIFICATIONS 16](#_Toc104882672)

[**3** 16](#_Toc104882673)

[3.1 System Design 16](#_Toc104882674)

[3.2 Logical Design 17](#_Toc104882675)

[3.3 System Architecture 18](#_Toc104882676)

[3.4 System Interaction and Use Cases 19](#_Toc104882677)

[3.5 Algorithmic Viewpoint 20](#_Toc104882678)

[CHAPTER 4: DEVELOPMENT AND TOOLS 21](#_Toc104882679)

[4.1 Introduction 21](#_Toc104882681)

[4.2 Development 21](#_Toc104882683)

[4.2.1 Tools and Technologies Used 21](#_Toc104882684)

[4.2.2 External APIs/ External Hardware 21](#_Toc104882685)

[4.3 System Implementation 21](#_Toc104882686)

[4.4 Data and Information flow in the system 22](#_Toc104882687)

[4.5 User Interfaces 22](#_Toc104882689)

[4.6 Additional Modules and API Provided 22](#_Toc104882690)

[CHAPTER 5: QUALITY ASSURANCE 23](#_Toc104882691)

[5.1 Introduction 23](#_Toc104882693)

[5.2 Test Plan 23](#_Toc104882695)

# CHAPTER 1: INTRODUCTION

This is an introductory chapter. After reading this chapter, you will understand the system, like why this system is important, why we are using it, how it will work, what are the objectives of the proposed system and in what kind of applications it can be implemented.

## System Introduction

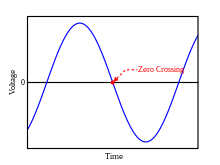
Living in a world encompassed by various types of sound from various sources, our mind, and hear-able framework is continually distinguishing each sound that it hears, in its direction. Characterizing sound or sound has been a significant field of examination for a long time now and there have been many attempted and tried techniques with various models and highlights which have shown to be valuable and precise. Order of sound can go from fields like media, bioacoustics observing, and gatecrasher identification in natural life regions to sound observation, and ecological sounds.

There are three different stages that are attached to the classification of sound; pre-processing of the audio signal, specific spectral feature extraction, and finally the classification of the audio signal. Signal pre-processing samples the input audio signal into various fragments which are utilized for extricating essential features. Zero-crossing rate, spectral flux, centroid, Chroma vector, MFCC (Mel-frequency Cepstral Coefficient), and poly features are among the most well-known handcrafted features for audio classification. For the sake of our approach to audio classification, we have chosen the following features: Chroma gram, Mel spectrogram, Spectral Contrast, Tonnetz, MFCC, etc. The idea of MFCC is to convert time-domain signals into frequency domain signals and use Mel filters to mimic cochlea that has more filters at low frequency and fewer filters at high frequency. Thus, it is safe to conclude that the feature MFCC and its characteristics are focused on the audibility of the human hearing system that can accommodate the dynamic nature of true-life sounds with the way that they are treated with feature vectors for classification.

1. **Zero-crossing rate:**

A zero-crossing is where the indication of a numerical capacity changes, addressed by a capture of the pivot (zero worth) in the diagram of the capacity. It is an ordinarily involved term in hardware, math, acoustics, and image processing.

The zero-crossing rate (ZCR) is the rate at which a signal changes from positive to zero to negative or negative to zero to positive. Its worth has been widely utilized in both speech recognition and music data recovery for ordering percussive sounds.

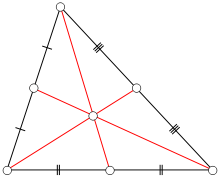


1. **Spectral flux:**

Spectral flux is a proportion of how rapidly the power range of a sign is changing, determined by comparing the power range for one frame against the power range from the past frame.

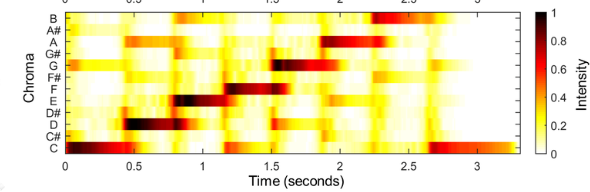
1. **Centroid:**

The centroid is the center of the circle. The point wherein the three medians of the triangle meet is known as the centroid of a triangle. It is likewise characterized as the mark of crossing point of the multitude of three medians. The middle is a line that joins the midpoint of a side and the contrary vertex of the triangle. The centroid of the triangle separates the middle in the proportion of 2:1. It tends to be found by taking the normal of x-coordinate places and y-coordinate marks of all the vertices of the triangle.



1. **Chroma vector:**

Chroma vector is also known as chromagram. Chroma-based highlights, which are likewise referred to as "pitch class profiles", are a useful asset for examining music whose pitches can be seriously sorted (frequently into twelve classes) and whose tuning approximates to the equivalent tempered scale. One principal property of chroma highlights is that they catch symphonious and melodic qualities of music, while being powerful to changes in tone and instrumentation.



1. **MFCC:**

In sound handling, the mel-frequency cepstrum (MFC) is a representation of the momentary 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 by and large make up a MFC. They are gotten from a kind of cepstral representation of the audio cut (a nonlinear "spectrum-of-a-spectrum"). The contrast 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 framework's response more intently than the linearly-spaced frequency bands utilized in the normal spectrum. This frequency warping can allow for better representation of sound, for example, in audio compression.

MFCCs are commonly derived as follows:

* Take the Fourier transform of a signal.
* Map the powers of the spectrum obtained above onto the mel scale, using triangular overlapping windows or alternatively, cosine overlapping windows.
* Take the logs of the powers at each of the mel frequencies.
* Take the discrete cosine transform of the list of mel log powers, as if it were a signal.
* The MFCCs are the amplitudes of the resulting spectrum.

1. **Poly features:**

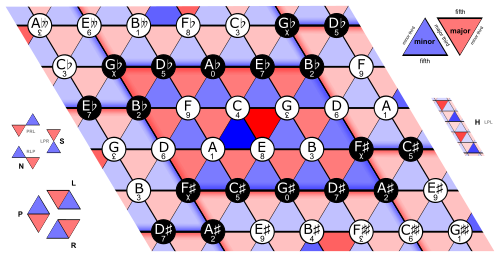
Polynomial features are those features created by raising existing features to an exponent. For example, on the off chance that a dataset had one information feature X, a polynomial feature would be the addition of another feature (section) where values were calculated by squaring the values in X, for example X^2. This cycle can be repeated for each information variable in the dataset, creating a transformed version of each.

1. **Spectral Contrast:**

Spectral contrast is characterized as the decibel distinction among peaks and valleys in the spectrum. There are two general motivations behind spectral contrast enhancement for hearing-impaired (HI) individuals. To begin with, in a sensorineural-impaired cochlea, auditory channels are generally broader than the normal and are much of the time abnormally asymmetrical.

1. **Tonnetz:**

In musical tuning and harmony, the Tonnetz (German for 'tone organization') is a conceptual lattice diagram addressing tonal space originally depicted by Leonhard Euler in 1739. Various visual representations of the Tonnetz can be utilized to show traditional harmonic relationships in European classical music.



## Background of the System

An early technique to handle sound recognition issues was proposed by Sawhney and Maesx, which displayed the temporal evolution of audio features and utilize intermittent neural organizations and a k-nearest neighbor criterion to demonstrate the mapping among features and categories. Notwithstanding, such a temporal feature could hardly obtain an accuracy larger than 70%. Then, at that point, Eronen m mn et al. found that the MFCC feature that based on human recognition of soundscape and how human ears perceive various frequencies. Some soundscape research extract MFCC features to depict the local spectral wrap of audio signals and classify them with SVM (radial basis function bit) and random woods.

The outcome shows an accuracy of approximately 70% which is a lot of lower than the deep learning technique. Another strategy utilizing HMM to translate audio into occasions performs inadequately within the sight of meddling acoustic noise. Since noise is unavoidable in a practical situation, we should find features and models more hearty. Thusly, researchers will quite often find powerful machine learning strategies like CNN, and RNN with high dimensional info features like spectrograms. These further develop power because of the discriminative capabilities of the back-end classifiers. Our work is particular in the accompanying ways. We will apply CNN with two distinct features: Mel and MFCC, to compare the performance of various features. And then, at that point, we will attempt to show to anticipate features.

### SVM

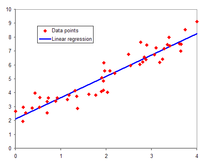
Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers detection.

#### Classification

#### Classification is a course of categorizing a given arrangement of data into classes, It can be performed on both organized or unstructured data. The cycle starts with anticipating the class of given data focuses. The classes are often alluded to as target, label or categories.

#### Regression

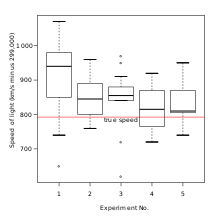
#### Regression is a method for investigating the relationship between free variables or features and a reliant variable or result. It's utilized as a technique for prescient demonstrating in machine learning, in which an algorithm is utilized to predict continuous results.



Regression line for 50 random points in a Gaussian distribution around the line y=1.5x+2 (not shown).

#### Outliers

#### Outliers can slant results, and anomalies in training data can impact overall model adequacy. Exception detection is a vital device in safeguarding data quality, as anomalous data and mistakes can be taken out and analyzed once recognized. Exception detection is an important part of each stage of the machine learning process.



Box plot of data from the Michelson–Morley experiment displaying four outliers in the middle column as well as one outlier in the first column.

### HMM

### A Hidden Markov Model (HMM) is a statistical model which is also utilized in machine learning. It tends to be utilized to portray the evolution of observable occasions that rely upon internal factors, which are not straightforwardly observable.

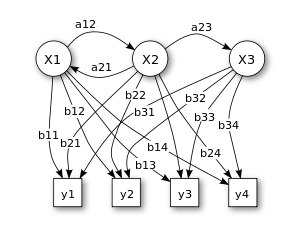


Figure 1. Probabilistic parameters of a hidden Markov model (example)

X — states

y — possible observations

a — state transition probabilities

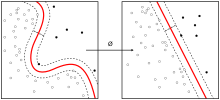
b — output probabilities

### Acoustic noise:

### Acoustic noise alludes to waves of strain traveling through air or different gases. Sound is acoustic energy in the audible range. acoustic energy capable of being heard.

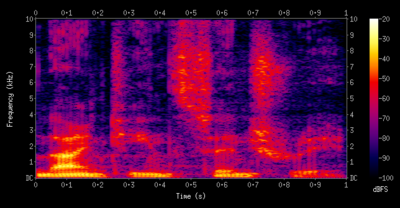
### RNN

Recurrent neural networks (RNN) are the state of the art algorithm for sequential data and are utilized by Apple's Siri and Google's voice search. The primary algorithm recalls its contribution, because of an internal memory, which makes it impeccably appropriate for machine learning issues that include sequential data. It is one of the algorithms in the background of the amazing achievements found in deep learning throughout the course of recent years.



#### Spectrogram

A spectrogram is a visual way of addressing the signal strength, or "commotion", of a signal after some time at various frequencies present in a particular waveform. Not exclusively can one see whether there is something else or less energy at, for example, 2 Hz versus 10 Hz, yet one can also perceive how energy levels vary after some time.



#### Classifier

A classifier in machine learning is an algorithm that automatically orders or categorizes data into one or more of a set of “classes.”

## Objectives of the System

* + To identify the object in sound.
  + It can be used in security systems.
  + It can help in controlling traffic signals by identifying the sound of traffic jams.
  + It can be used in vehicles to determine if the vehicle is in a running state or starting state.

## Significance of the System

This study manages the job of sounds in object acknowledgment in people. To be sure, a few items are effectively connected with a sound, i.e., a few items have either a run of the mill sound or a class of sounds. This is the situation, for instance, with items, for example, "ringer" or "motorbike." Other articles don't have common sounds or can be related with specific sounds just with difﬁculty. This is the situation, for instance, of items, for example, "table" or "cushion. "Considering that items can be classiﬁed as an element of regardless of whether they have an ordinary sound, a genuine inquiry is whether the normal sounds assume any part in the visual acknowledgment of the connected items. There are something like two went against situations to approach this inquiry.

In the ﬁrst scenario, upon the presentation of a visual object the system ﬁrst accesses an abstract representation of that object and then depending on the task at hand accesses the representations of information related to that object: among these representations is the representation of the typical sound. Thus, in this scenario, the access to the typical sound is post-categorical, in the sense that the object is ﬁrst recognized as an instance of a particular kind (e.g., a “dog”) and then the related information is retrieved. Here, the typical sound may be activated but, since its retrieval follows the identiﬁcation of the object, it does not play any role in the recognition of the object. In the second scenario, all stored representations associated with a given object are immediately and mandatorily activated upon the visual presentation of an instance of that kind of object. Here, the identiﬁcation of the object does not consist in the activation of an abstract semantic representation of these objects but instead corresponds to the activation of all stored representations. In other words, object identiﬁcation is the activation of object knowledge. So, this system is important in the classification and recognition of object sounds.

# CHAPTER 2: REQUIREMENT SPECIFICATIONS

In this chapter, you will be able to understand the requirements of the system. Understanding the requirements or requirement gathering is the most important part of building a system. In this chapter, you will understand both user and system-level requirements. It includes functional requirements, non-functional requirements, system requirements, and behavioral requirements.



## Product Scope

The scope of this work is focused on an urban sound classification. As an important processing step, feature extraction plays a critical role that will significantly affect the final classification performance. We try to break through the performance bottleneck, using novel feature sets extracted with image information retrieval techniques. It describes the experiments applying a convolutional neural network (CNN), a state-of-the-art image digit recognition algorithm, to the automatic extraction of sound pattern features. The system architecture, the characteristics of CNN, and the classification performance are explained. studies the invariance of the widely used MFCC feature set to Urban Sounds.

## Product Description

### Product Perspective

From the technology perspective, object sound recognition has a long history with several waves of major innovations. Most recently, the field has benefited from advances in deep learning and big data. The advances are evidenced not only by the surge of academic papers published in the field but more importantly by the worldwide industry adoption of a variety of deep learning methods in designing and deploying speech recognition systems.

### Product Functionality

* Detect Object by Voice File
* Detect Object by Recording
* Detect Object in Real-time

### Users and Characteristics

In this System user can be anything like if it is used in security system the user will the security system that will be connected to this system and generate an output by the response of this system.

### Operating Environment

This application will can be used in windows and the minimum GPU will be required 2.17, in case of security system input device is required to accept sounds from the environment. A microphone will be used for input to the system. Or the audio file can be uploaded to get the result.

## Specific Requirements

### Functional Requirements

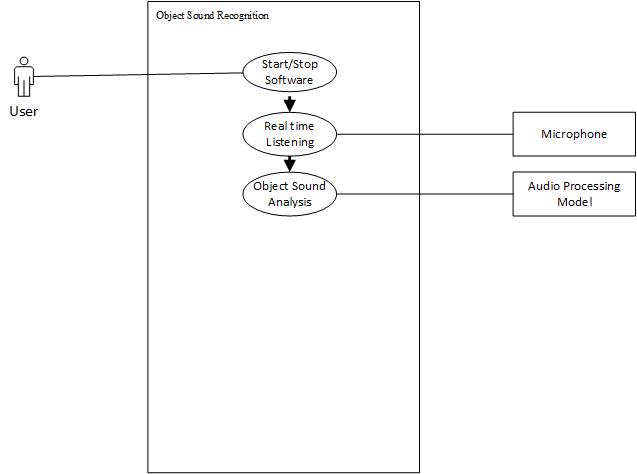
#### Take voice input in Real-time

#### The system shall be able to take input in Real-time and shows the output on interface.

#### Take voice input by audio file

The system should be able to process audio file and shows output to the interface.

#### Use Case Diagram

..

#### Use Case Descriptions

* User start or stop the software
* System will interact with the microphone while real-time listening
* System will interact with audio processing model while performing object sound analysis

### Behavioral Requirements

* User will click on start button
* Application will receive sound from the environment by using microphone
* This sound will be sent to the Audio Processing Model
* Model returns the matching result if any found.

### External Interface Requirements

#### User Interface

Chart

Description automatically generatedThere is only one screen for the user to start and stop the system and view the system’s performance.

#### Other Interfaces

Cutting-edge GPU is required if it will be developed in a local lab. Preferably, 5800+ Quad-core GPU would be fine. 8000+ better. Since the dataset is more than 7 GB, preferably, 4 GB or more amount of RAM is needed. 100GB of the mass storage device is needed. Related to software interface, Application has the button on the main page to start and stop the system and a graph to see the sound’s performance.

## Non-functional Requirements

### Performance Requirements

#### Response Time

The system will work with real-life data. Response time to identifying shouldn't exceed 300 msec.

#### Error Handling

When the system fails to predict in a given response time, it should give an alert message to the interface.

#### Workload

The system should be able to handle identifying given input while the microphone is taking input.

### Safety and Security Requirements

* Make sure system will not crashed due to electricity issue
* System will be safe from electricity shocked.
* System will be safe from theft.
* There is no hurdles between the environment and system input place.

### Software Quality Attributes

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Functionality | System designed to identify the objects. |
| Reliability | System should operate 99.9 % in runtime. |
| Robustness | System should successfully identify urban sounds objects which will be trained with training data. |
| Portability | System should work Ubuntu 16.04/Windows or later OS version. |
| Efficiency | Source codes of the system should be clear and operate efficiently. |

# CHAPTER 3: DESIGN SPECIFICATIONS

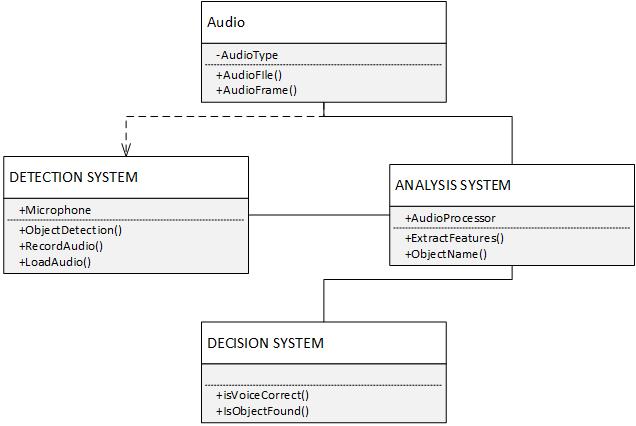
In this chapter, you will understand the design specifications of object sound detection and recognition. You will find out different diagrams like class diagrams for logical design purposes, sequence diagrams for system interaction purposes, and so on.



## System Design

## C:\Users\Azeem Amjad\Pictures\image.png

## Logical Design



## System Architecture

Diagram, engineering drawing

Description automatically generated

Model: "sequential"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

conv2d (Conv2D) (None, 16, 8, 64) 640

max\_pooling2d (MaxPooling2D (None, 8, 4, 64) 0

)

conv2d\_1 (Conv2D) (None, 8, 4, 128) 73856

max\_pooling2d\_1 (MaxPooling (None, 4, 2, 128) 0

2D)

dropout (Dropout) (None, 4, 2, 128) 0

flatten (Flatten) (None, 1024) 0

dense (Dense) (None, 1024) 1049600

dense\_1 (Dense) (None, 10) 10250

=================================================================

Total params: 1,134,346

Trainable params: 1,134,346

Non-trainable params: 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Confusion Matrix Details:**

Expected: [3, 2, 2, 2]

Predicted: [3, 2, 2, 2]

TN = 3, FP: 0, FN = 0, TP = 1

## System Interaction and Use Cases

* When user clicks on the start button, it will start the **Audio Handler** that record the audio file.
* Then it sends this file to the **Audio Processor** that Extract features from the file.
* Then it sends these features to the **Model**.
* **Model** generates prediction that sends back to the user as output.

Diagram

Description automatically generated

## Algorithmic Viewpoint

A Pseudo-code of the Proposed Approach

Input: Dataset: Xmsi=1, Bayesian learners

Output: ensemble meta-classifier

For i=1 to maxiterations Do:

Split dataset into k-folds

For Each fold in l-folds Do:

For Each predictor in ensemble Do:

Train learner on train set in fold

Validate class probabilities from learner in fold

Create prediction matrix of classes probabilities

End For

End For

Calculate probabilities across learners

Get loss value of loss function with probabilities

If loss<previous Then:

Stack learner

Else:

Save model checkpoint

Break For

End For

# CHAPTER 4: DEVELOPMENT AND TOOLS

This chapter describes the tools and techniques used in this project. The tool used is Spyder with the language python. The different libraries of python are used in this project.



## Introduction

## While deploying the application, we need a system that has a 1 TB+ mass storage device and contains a minimum of 4 GB of RAM. We need python 3.9 or more to deploy the system. We will import some libraries to manage data and build the model. Then we write functions to convert our audio data into images. We use LIBROSA to extract audio time series and the sampling rate of audio files. Then we apply to process.

## Development

### Tools and Technologies Used

Python 3.9 or above version needs to use the application, Cutting-edge GPU is required if it will be developed in a local lab. Preferably, 5800+ Quad-core GPU would be fine. 8000+ better. Since the dataset is more than 7 GB, preferably, 4 GB or more amount of RAM is needed. 100GB of the mass storage device is needed.

CONVOLUTIONAL NEURAL NETWORK

### External APIs/ External Hardware

There is no API’s used in the application. Application works standalone.

## System Implementation

1. Imported the Pandas and Numpy packages.
2. We installed librosa to extract audio features using pip. If you’re doing this locally, you should do the same via console.
3. We import the necessary keras to build our network as well as other necessary packages. Finally, we also build working directories in our store our checkpoint files.
4. Next, we begin our data conversion process by defining the functions that will convert our .wav files into feature like images. Briefly, we extract the audio time-series and sampling rate of each .wav file using librosa.
5. With that defined, we converted our training data. We did this once to extract all features.

## Data and Information flow in the system

## The application will get the audio file for a few seconds and convert it into features like image had by extracting the time series and sampling rate using LIBROSA Library. Then application gave these features to the algorithm to get the process the features and it checks if it contains any target sound from urban sounds.

## User Interfaces

Chart

Description automatically generated

## Additional Modules and API Provided

There is no API’s used

# CHAPTER 5: QUALITY ASSURANCE

In this chapter, we will understand why quality assurance is important and how we make different test cases to test the quality of the system. The quality assurance phase is mainly based on a Test plans including test strategies and types of tests applied to ensure the reliability and accuracy of the application to give the user a great and error-free learning experience.



## Introduction

Quality assurance is one of the main periods of a framework's improvement life cycle. In this stage, we check our framework to test whether it is working as per the necessities of the client or is it working appropriately as per the useful and non-utilitarian prerequisites.

## Test Plan

Table 5.1: Test case for system start up

|  |  |
| --- | --- |
|  | TC-01 |
| Test name | System’s start up |
| Date of test | 22/05/2022 |
| Name of application | Object sound detection and recognition |
| Description | Microphone starts recording while user click on run button. |
| Input | Click on start button. |
| Expected output | Start recording. |
| Actual output | Start Recording. |
| Test Role (Actor) | Team Member |
| Test verified by | Team Member/Supervisor |

Table 5.2: Test case for system’s Microphone

|  |  |
| --- | --- |
|  | TC-02 |
| Test name | System’s microphone |
| Date of test | 22/05/2022 |
| Name of application | Object sound detection and recognition. |
| Description | When driver clicks on start button, microphone enabled for real time to capture sound. |
| Input | None |
| Expected output | Capture sound in Real time |
| Actual output | Capture sound in real time |
| Test Role (Actor) | Team Member |
| Test verified by | Team Member/Supervisor |

# Reference

## Zero-crossing rate <https://en.wikipedia.org/wiki/Zero-crossing_rate>

## Spectral flux <https://en.wikipedia.org/wiki/Spectral_flux>

## Centroid <https://byjus.com/maths/centroid>

* Chroma vector <https://en.wikipedia.org/wiki/Chroma_feature>
* MFCC <https://en.wikipedia.org/wiki/Mel-frequency_cepstrum>
* Poly features <https://machinelearningmastery.com/polynomial-features-transforms-for-machine>
* Tonnetz <https://en.wikipedia.org/wiki/Tonnetz>
* SVM <https://scikit-learn.org/stable/modules/svm.html>
* Classification <https://www.edureka.co/blog/classification-in-machine-learning>
* Regression <https://www.seldon.io/machine-learning-regression-explained>
* Outliers <https://www.seldon.io/outlier-detection-and-analysis-methods>
* HMM <https://analyticsindiamag.com/a-guide-to-hidden-markov-model-and-its-applications-in-nlp>
* Acoustic noise <https://www.herzan.com/resources/applications/noise-source/acoustic-noise.html>
* RNN <https://builtin.com/data-science/recurrent-neural-networks-and-lstm>
* Spectrogram <https://pnsn.org/spectrograms/what-is-a-spectrogram>
* Classifier <https://monkeylearn.com/blog/what-is-a-classifier>