# INFANT CRY DETECTION SYSTEM

**A Project Report**

**Submitted in partial fulfilment of the**

**Requirements for the award of the Degree of**

**Bachelor of Technology**

**In**

**Computer Science and Engineering**

**Submitted By**

**Section-1**

**BATCH-100**


**KANTHETI MOHANA 180030330**

**ALAPATI YOMITHA 180030333**

**YEDAVALLI SRUTHI 180030930**


**Under the Supervision of**

**Dr. K. SWARNA**

**(Associate Professor)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**K L E F, Green Fields,**

**Vaddeswaram - 522502, Guntur(Dist), Andhra Pradesh, India Nov-2021**

# K L University

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING


## *DECLARATION*

The project based report entitled **"INFANT CRY DETECTION SYSTEM"** is a record of bonafide work of K.Mohana (180030330), A.Yomitha (180030333), Y.Sruthi (180030930) submitted in the partial fulfillment for the award of B.Tech in Computer Science and Engineering to the K L UNIVERSITY. The results embodied in this report have not been copied from any other departments/University/Institute..

# K L University

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING


## *CERTIFICATE*

This is to certify that this project based report entitled **"INFANT CRY DETECTION SYSTEM"** is being submitted by K.Mohana (180030330), A.Yomitha (180030333), Y.Sruthi (180030930) submitted in the partial fulfillment for the award of B.Tech in Computer Science and Engineering to the K L UNIVERSITY is a record of bonafied work carried out under guidance and supervision.

The results embodied in this report have not been copied from any other departments/University/Institute.

**Signature of the Supervisor**
Dr.K.Swarna mam
(Assoc Prof)

**Signature of the HOD**
Prof.V.HARI KIRAN
(Head of the Department)

# ACKNOWLEDGMENTS

My sincere thanks to **Dr.K.Swarna mam** for their outstanding support throughout the project for the successful completion of the work

We express our gratitude to **Dr.V.HARI KIRAN SIR,** Head of the Department for Computer Science and Engineering for providing us with adequate facilities, ways and means by which we are able to complete this project work.

We express our gratitude to **Dr. Siva nageswar rao,** Professor In-Charge for Computer Science and Engineering for providing us with adequate facilities, ways and means by which we are able to complete this minor project work.

We would like to place on record the deep sense of gratitude to the honourable Vice Chancellor, K L University for providing the necessary facilities to carry the concluded minor project work.

Last but not the least, we thank all Teaching and Non-Teaching Staff of our department and especially my classmates and my friends for their support in the completion of our minor project work

# ABSTRACT

Crying is the first sound a child makes when he/she enters the world. It is a way of expressing different circumstances including pain, hungry, lonely, discomfort etc. Parents try to recognize their sounds but they fail to find why the baby cries in time. Baby cry sound detection allows parents to be automatically alerted when their baby is crying. Automatic detection of a baby cry in audio signals is an essential step in applications such as remote baby monitoring. We introduce pre-processing approaches and describe a diversity of features such as MFC, LPCC, STE, etc. Together with traditional machine learning classifiers such as KNN, SVM, and GMM, newly developed neural network architectures such as CNN and RNN are applied in infant cry research. The features extracted by doing the pre-processing stpes are taken as input for the further classification of infant cry.

## CONTENTS

# 1.INTRODUCTION

The dtection of cry sounds is an important pre-processing step for various applications involving cry analysis. Cry signals has been object of analysis and research for many years. Many researchers and scientists have found that necessary evidence that cry signals provides relevant information about physical and psychological states of newborns babies. Audio classification is the process of listening to and analyzing audio recordings. Also known as sound classification, this process is at the heart of a variety of modern AI technology including virtual assistants, automatic speech recognition, and text to speech applications. Over the decades a number of works has been done in the field of infant cry. Infant cry sounds due to six causes i,e. pain, discomfort, ailment, environment factors, emotional need and hungry of infant cry. So first the infant cry is pre-processed to get the csv file using the python libraries. Then by using that csv file we can find out the cause for infant cry using algorithms regression, CNN, LDA .

## 2.LITERATURE SURVEY

**TITLE:** BABY CRY DETECTION: DEEP LEARNING AND CLASSICAL APPROACHES

**AUTHORS:** RAMI COHEN, DIMA RUINSKIY, JANIS ZICKFELD

In this paper they compare deep learning and classical approaches for detection of baby cry sounds in various domestic environments under challenging signal-to-noise ratio conditions. They used several CNN architectures for infant cry detection and to compare their performance with classical machine-learning approaches like regression, SVM etc. In this model to move ahead with CNN acoustic features are extracted using RNN and by designing CNN with more carefully they are able to obtain better results.

**TITLE:** BABY CRY DETECTION IN DOMESTIC ENVIRONMENT USING DEEP LEARNING

**AUTHORS:** YIZHAR LAVNER, RAMI COHEN, HANS IJZERMAN

In this paper they used two machine-learning algorithms low-complexity logistic regression and CNN. The first algorithm is used as a reference and they trained the model and extracted features like MFCC, pitch, and formants. In second algorithm CNN , each audio signal in divided into different segments and each segment is further divided into frames so that processing can be easily done. These frames are taken as input for CNN architecture to get best results. Compared to logistic-regression CNN yielded better results.

**TITLE:** A REVIEW OF INFANT CRY ANALYSIS AND CLASSIFICATION

**AUTHORS:** CHUNYAN JI, THOSINI BAMUNU, YUTANG GAO

In this paper all the recent works on infant cry analysis are researched. Many literatures are reviewed based on signal processing techniques and machine-learning algorithms. Pre-processing techniques are introduced so as to extract the acoustic features. Along with traditional approaches CNN and RNN are also used for in infant cry research.

**TITLE:** INFANT CRY DETECTION IN REAL WORLD ENVIRONMENTS

**AUTHORS:** Xuewen Yao, Megan Micheletti, Mckensey Johnson, Edison Thomaz, Kaya de Barbaro

This paper gives the problems of infant cry detection in real-world. Most of the existing cry detection systems were tested with the data collected with controlled settings i.e with in home environment and noisy sounds. In this paper they evaluated several machine-learning algorithms with deep spectrum and acoustic features. This model has accuracy very low which in real-world is not applicable because in real-world settings there may be several other noises and some other criteria with which the model should be able perform with more accuracy. With the findings in this paper they concluded that in-lab data does not perform well when compared with real-world data.

**TITLE:** A FULLY AUTOMATED APPROACH FOR BABY CRY SIGNALS SEGEMENTATION AND BOUNDARY DETECTION FOR EXPIRATORY AND INSPIRATORY EPISODES

**AUTHORS:** LINA ABOU-ABBAS, CHAKIB TADJI, HESAM ALAIE FERSAIE

In this paper, a framework for automatic cry sound segmentation for application in a cry-based diagnostic system has been proposed. The main focus of this paper is to extend the systems in the previous work to post-processing based on intensity, zero crossing rate etc. In this paper they used Gaussian mixture and Hidden markov model. Various other features are also considered in this models to increase the efficiency and accuracy.

**TITLE:** DEEP LEARNING ASSISTED NEONATAL CRY CLASSIFICATION VIA SUPPORT VECTOR MACHINE MODELS

**AUTHORS:** ASHWINI K, P.M.DURAI RAJ VINCENT, KATHIRAVAN SRINIVASAN, CHUAN-YU CHANG

In this paper, the audio signals are transformed into images using short time fourier transform (STFT) . The DCNN takes these images as input and the features obtained are taken as input for SVM model. This work combines machine learning and deep learning techniques which produces best results with moderate samples. By combining CNN based feature extraction and SVM model the accuracy of the model is increased and we get best results compared to other algorithms and methods.

## 3.EXISTING SYSTEM:

Many people proposed many algorithms related to infant cry detection system. Some of them include CNN, KNN, Hidden markov model, Gaussian mixture. Using CNN the audio files / tracks we have will be taken as input and these sound tracks are segmented into frames and the first two layers of CNN will be extracting the acoustic features and the remaining layers will take these features as input and classify the audio files regarding the reason behind the infant cry. The KNN model classifies whether the sound signals are either speech or music or cry etc which can be used in pre-processing of audio tracks. HMM model classifies the audio signals using time series signals so that based on timing we can classify the audio signals. It is also used in audio classification in infant cry detection and has high accuracy.

## 4.PROPOSED SYSTEM:

In our proposed system we use LDA algorithm for infant cry detection. LDA is mostly used to extract the features from the sound tracks we classify. It is very efficient in extracting and features and classifying the cry. In LDA we extract the features and to classify the audio signals by taking the input as the acoustic features we find out their frequency. To find the frequency we have many methods and to find frequency in LDA we have some formulas relating to eigen values. In LDA eigen vectors and eign values play an important role. When we calculate these eigen values these values take / consider all the values that are neglected which makes a vast difference in output and accuracy of system.

# 5.DATASET

## DATASET DESCRIPTION

The database used for this study consists a few seconds of audio recordings made by parents of babies when they are crying. Many infants cry is recorded and they are made together in a folder so as to use it conveniently while programming. The audio files consists of many other sounds other than infant cry which are to be removed while training the dataset. The dataset we took from github. This github dataset consists of different types sounds made by the children some of include crying baby, noise, silence, baby laugh. Among these we consider crying baby folder dataset. In crying baby folder consists of many babies crying sound tracks in various situations. There are minimum of five audio files of each baby crying.

# 6 PRE-PROCESSING FEATURES:

## 6.1 LOUDNESS:

The loudness of a sound relates the intensity of any given sound to the intensity at the threshold of hearing. It is measured in decibels (dB). The threshold of human hearing has an intensity of about. If an infant cries loudly it means that the baby is in pain or sorrow and this loudness depends on amplitude of the wave. When we listen to a sound signal, the amplitude controls how loud we receive the signal to be. Larger amplitude means louder the sound. Zero amplitude means no sound. Our ears can analyse how much louder the sound is but machines cant analyse so we calculate loudness by intensity.

## 6.2 ENERGY:

Infant cry signals differs from adults and the variations in the wave forms can be observed mostly in energy. since, infants don't have much energy we can identify those signals easily in wave forms. energy of a signal is defined as the integral of the signal squared during the time that you are calculating it, or in our case since this is digital signal, sum of squared samples of the signal. So, the short term energy has unit which is the square of the unit of the original signal.If an infant cry so loudly there it self it represents that how much energy an infant used. Energy is one factor that helps us to find out reason behind the cry, since if a baby got a small scratch on a their body they cannot take it and they cry with all the energy they have so it is important for finding the reason for cry.

## 6.3 PITCH:

Pitch, the relative highness or lowness of a tone as perceived by the ear, which depends on the number of vibrations per second produced by the vocal cords.The pitch of a sound is how high or low the sound is. A high sound has a high pitch and a low sound has a low pitch with respect to its intensity. Pitch is a perceptual property of sounds that allows their ordering on a frequency-related scale, or more commonly, pitch makes it possible to judge sounds as "higher" and "lower" in the sense associated with musical melodies.

### 6.4 MFCC:

The Mel-Frequency Cepstral Coefficients (MFCC) feature extraction method is one of the main approach in speech feature extraction and present research aims to identify performance enhancements. One of the most recent MFCC implementations is the Delta-Delta MFCC, which improves speaker verification.The MFCC is a feature set that describes the shape of the short-term speech spectrum using a small vector of weights.Mel frequency cepstral coefficients (MFCC) was originally suggested for identifying monosyllabic words in continuously spoken sentences but not for speaker identification. ... MFCC is used to identify numbers spoken into a telephone and voice recognition system for security purpose. It is one of the most important feature in audio classification because this values can give a vast difference in results.

# 7 THEORITICAL ANALYSIS (METHODOLOGIES)

## 7.1 CONVOLUTION NEURAL NETWORK (CNN)

Convolutional Neural Network or CNN is a type of artificial neural network, which is most widely used for image or object recognition and classification in deep learning. A Convolutional neural network (CNN) is a neural network that has one or more convolutional layers and is mostly used for image processing, classification, segmentation and also for other auto correlated data. However, Deep Learning techniques recognizes objects in an image by using a CNN. In CNN the input audio signal is divided into different fragments and these fragments are taken as input in the form of image where the image is further processed using different layers of CNN. The different layers of a CNN are: the convolutional layer, the pooling layer, the ReLU correction layer and the fully-connected layer.

### 7.1.1 CONVOLUTION LAYER:

Convolutional layers are the most important layers i.e main building blocks used in convolutional neural networks. The first layer acts as a different neural network which acts as a feature extractor. Convolution layer and Pooling layer are used for extracting features in audio signals. In this layer in order to convert the audio signals to image and then extract features from this layer the image is involved with a filter i.e in the matrix form. It is also known as convolution matrix or convolution mask. In this layer to convert the image audio signal to the required size we use kernel. The size of the kernel is fixed and the image we have may be of any size i.e 5x5 or 4x4 so we have to convert this image size to kernel matrix size so that further processing can be done easily. When we want to extract more features from an image we need to use multiple kernels instead of using a single kernel and the size of all the kernels must be same so that after extracting features from the image we can combine it to one single image.

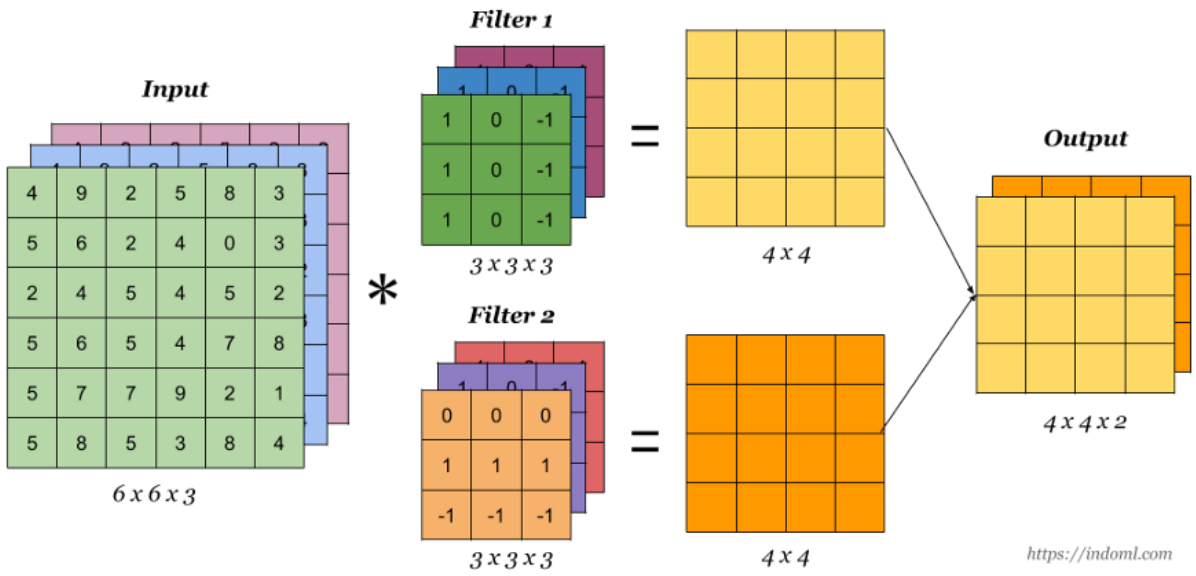



**Fig 7.a : Process followed in convolution layer**

### 7.1.2 POOLING LAYER:

Pooling layer is used to reduce the size of image taken as input so that there will be no more distractions when combining all the output images. Convolution layer is followed by pooling layer so this is process or flow to be followed in CNN while processing the audio signals. There are two types of pooling they are max pooling and average pooling.

#### 7.1.2.1 MAX POOLING:

In max pooling from each feature the maximum one is selected and the remaining images are reduced or maximised to this size to maintain equality when combining the images.

#### 7.1.2.2 AVERAGE POOLING:

In average pooling from each feature the average value feature is selected and all the remaining images or features rae reduced to this size.

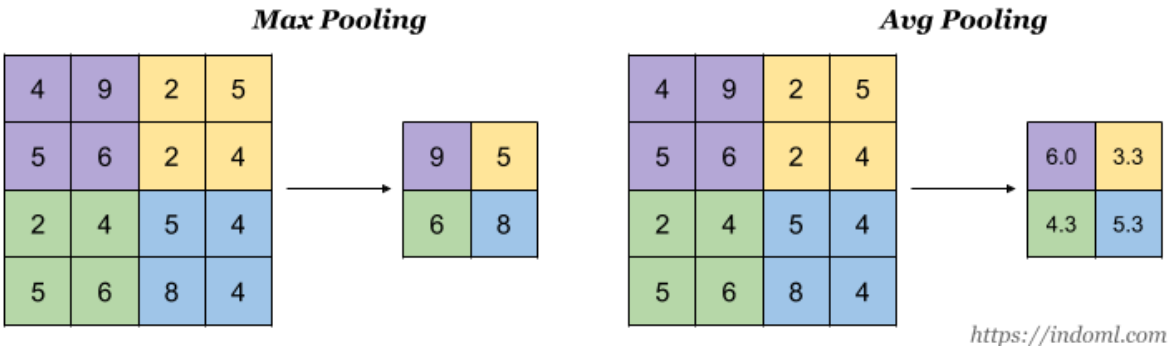



**Fig 7.b: Max pooling and Average pooling method followed in CNN pooling layer**

### 7.1.3: ReLU LAYER:

The ReLU layer applies function f(x) =max(0,x) to all the values. ReLU layer works more better because this network will be able to train the network fastly without making any significant difference to accuracy.

### 7.1.4: FULLY-CONNECTED LAYER:

Fully-Connected layer is the last layer in CNN which combines all the outputs of the previous layers to the single one. The output of this layer will be a single dimension feature vector.

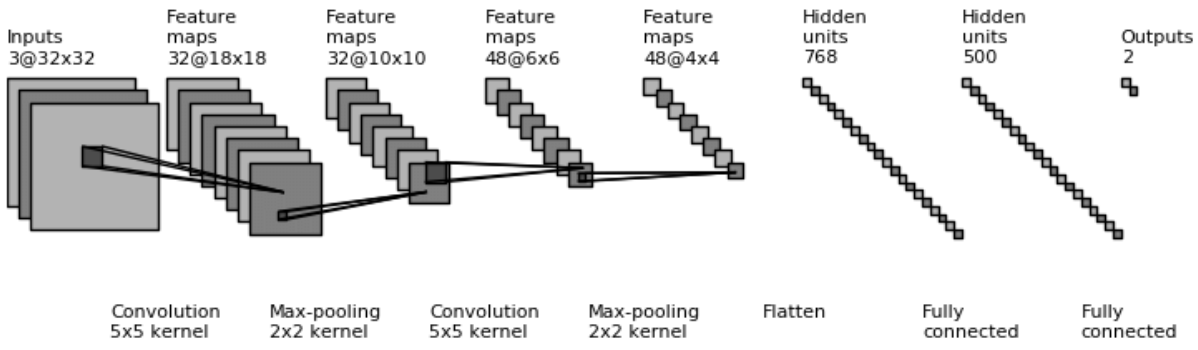



**Fig 7.c: All the features extracted from previous layers are connected / combined together in fully-connected layer**

**Total process followed in CNN:**

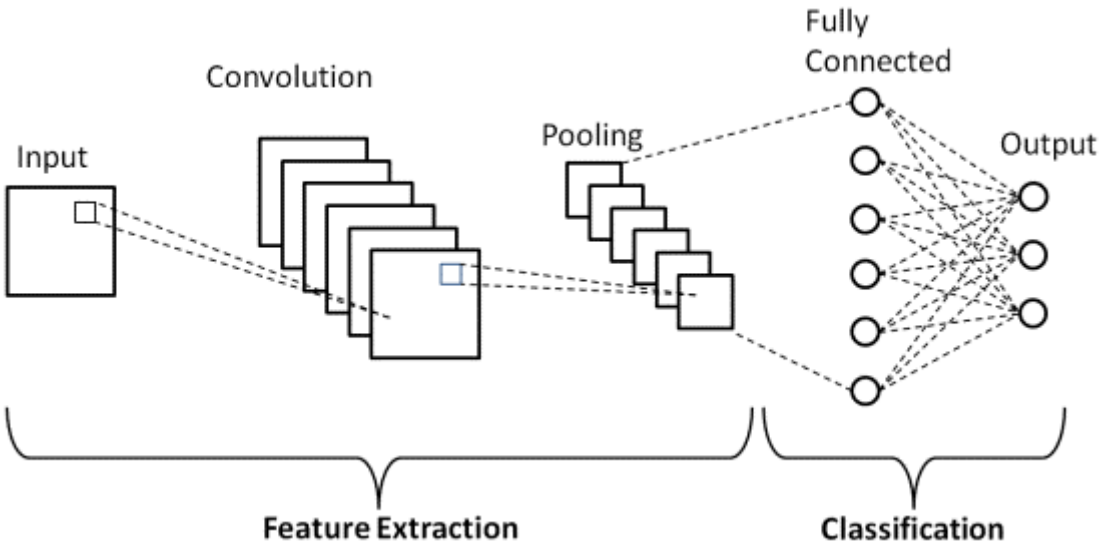



**Fig 7.d: Flow of execution followed in CNN algorithm**

## 7.2 LINEAR DISCRMINANT ANALYSIS (LDA)

Linear Discriminant Analysis or LDA is a dimensionality reduction technique in machine learning. This technique is used as a pre-processing step in applications of pattern classification to extract the features required to classify. LDA is a supervised classification technique that is considered as a part of crafting competitive machine learning models. Linear discriminant analysis(LDA) always discards the zero eigenvalues. But these zero eigenvalues ae important dimensions for discriminant analysis. In this algorithm, an objective function is proposed which utilizes both the principles and nullspace eigenvalues and simultaneously inherit the class separability information onto its latent space representation. To build the Convolutional neural network and perform the regularized discriminant analysis on top of it in an end-to-end fashion. Linear discriminant analysis will find out the axes and that maximize the between-class scatter matrix $S_b$, and minimizing the within-class scatter matrix $S_w$ in the projective subspace $A \in R^{lxd}$. A projective subspace is a lower dimensional subspace l=c-1. Where c is the no.of classes.

$$J(W) = \frac{\widetilde{S}_b}{\widetilde{S}_w} = \frac{|w^T S_b W|}{|w^T S_w W|}$$

Linear discriminant analysis(LDA) is mostly used to reduce the number of ascoetic features to a less number before classification. Each of these features are new dimensions are a linear combination of which form a template and pixel values. The linear Discriminant analysis estimates the probability that a new set of inputs belongs to every class. The output class is the one that has the highest probability. If the output class is (k) and the input is (x), here is how Bayes' theorem works to estimate the probability that the data belongs to each class. Linear Discriminant Analysis or LDA is a dimensionality reduction technique. It is used as a pre-processing step in Machine Learning and applications of pattern classification.

Listed below are the 5 general steps for performing a linear discriminant analysis; we will explore them in more detail in the following sections.

1. Calculate dd-dimensional mean vectors for the many classes from the dataset.

$$S_b = \sum_{i=1}^{g} N_i (\bar{x}_i - \bar{x})(\bar{x}_i - \bar{x})^T$$

2. Ccalculate the scatter matrices.

$$S_w = \sum_{i=1}^{g} (N_i - 1) S_i = \sum_{i=1}^{g} \sum_{j=1}^{N_i} (x_{i,j} - \bar{x}_i)(x_{i,j} - \bar{x}_i)^T$$

3. Compute the eigenvectors (ee1,ee2,...,eedee1,ee2,...,eed) and corresponding eigenvalues (λλ1,λλ2,...,λλdλλ1,λλ2,...,λλd) for the scatter matrices.

4. Sort the eigenvectors by decreasing eigenvalues and choose kk eigenvectors with the largest eigenvalues to form a d×kd×k dimensional matrix WWWW.

5. Use this d×kd×k eigenvector matrix to transform the samples onto the new subspace. This can be summarized by the matrix multiplication: YY=XX×WWYY=XX×WW (where XXXX is a n×dn×d-dimensional matrix representing the nn samples, and yyyy are the transformed n×kn×k-dimensional samples in the new subspace).

$$P_{lda} = \arg\max_{P} \frac{\left| P^T S_b P \right|}{\left| P^T S_w P \right|}$$

## 7.3 RDA:

Regularized Discriminant Analysis is to add a constant λ to the

diagonal elements in class scatter matrix

**Sw=Sw | + λ I**

where;

1.λ is the regularized parameter,

2.Sw is with in class covariance matrix,

3. I is Identity Matrix

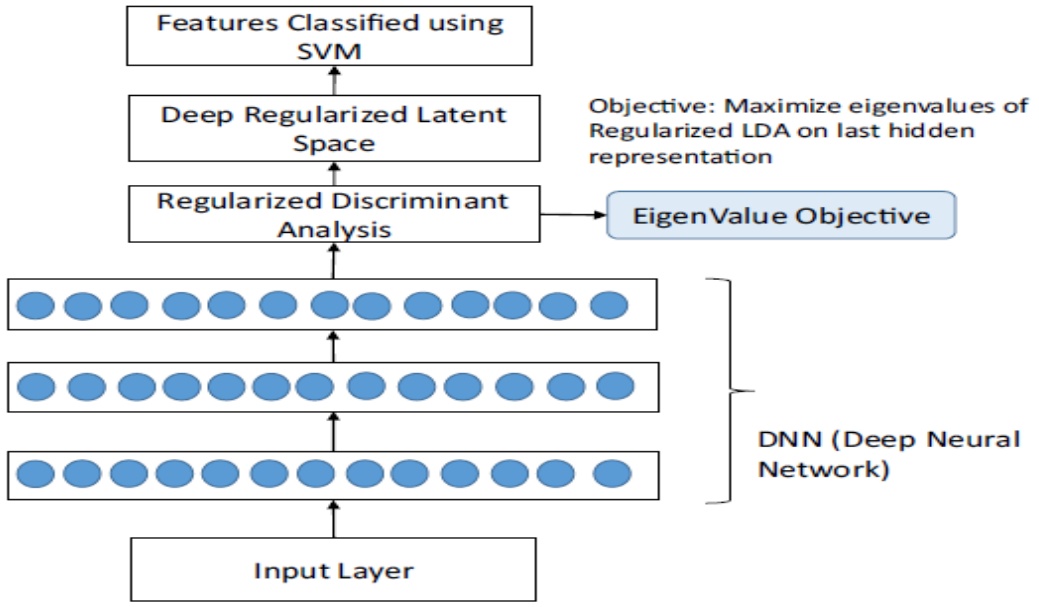
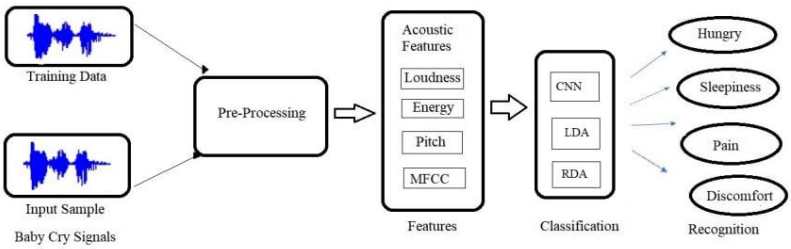



Fig. 1 Schematic sketch of deep regularized discriminative network

**Fig 7.e**

RDA is a regularization technique, which is particularly used when there are many features that are potentially correlated.

## 8 BLOCK DIAGRAM



## 9 SOFTWARE REQUIREMENTS

Operating system: WINDOWS 10

Tools: Python Jupyter Notebook

Language: Python

RAM: 2 GB

Hard-Disk: 6 GB

Processor: Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz

## 10 EXPERIMENTAL SETUP

### 10.1 Librosa:

Librosa is a python package used for audio analysis. It provides the basic necessary information required to create the music information retrieval systems. Librosa is mostly used when we are working with audio data sets or speech recognition. Librosa doesn't support all the extensions of sound files which may cause damage or fall back to audio library.

Command :

**!pip install librosa**

**import librosa**

### 10.2 Cython:

Cython is similar to python. cython combination of python and c. This cython contains c command but used in python. To use this cython module in python first we need to install in command prompt and then should import it.

Command :

**!pip install cython**

**import cython**

### 10.3 Numba:

Numba is a open source compiler used for python. It is used to convert the python code into machine learning code.

Command:

**!pip install numba==0.50**

**10.4 npm:**

npm is used to find the correct position of module in implementation. So that nodes easily find when needed. It mostly used for installing, discovering nodes programs. npm is used to helps in installing various packages in python. Solve the dependency problem in packages.

Command:

**!pip install npm**

**import npm**

## 11 IMPLEMENTATION

```
#required packages in this project

import librosa
import pandas as pd
import soundfile
import os, glob, pickle
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score

!pip install pandas
!pip install numpy

pip install npm
npm install npm@latest -g
pip install --upgrade pip

def extract_feature(file_name, mfcc, chroma, mel):
with soundfile.SoundFile(file_name) as sound_file:
X = sound_file.read(dtype="float64")
sample_rate=sound_file.samplerate
if chroma:
stft=np.abs(librosa.stft(X))
result=np.array([])
if mfcc:
mfccs=np.mean(librosa.feature.mfcc(y=X, sr=sample_rate,
n_mfcc=40).T, axis=0)
result=np.hstack((result, mfccs))
if chroma:
chroma=np.mean(librosa.feature.chroma_stft(S=stft,
sr=sample_rate).T,axis=0)
result=np.hstack((result, chroma))
```

```
if mel:
mel=np.mean(librosa.feature.melspectrogram(X,
sr=sample_rate).T,axis=0)
result=np.hstack((result,  mel))
return result
#DataFlair - Emotions in the RAVDESS dataset
emotions={
'01':'neutral',
'02':'calm',
'03':'happy',
'04':'sad',
'05':'angry',
'06':'fearful',
'07':'disgust',
'08':'surprised'
}

observed_emotions=['calm',  'happy', 'fearful', 'disgust']
def noise(data):
"""
Adding White Noise.
"""
# you can take any distribution from
https://docs.scipy.org/doc/numpy-1.13.0/reference/routines.random.html
noise_amp = 0.005*np.random.uniform()*np.amax(data)
data = data.astype('float64') + noise_amp *
np.random.normal(size=data.shape[0])
return data
#DataFlair - Load the data and extract features for each sound file
def load_data(test_size=0.15):
x,y=[],[]
for file  in glob.glob("Actors\\Actor_*\\*.wav"):
file_name=os.path.basename(file)
emotion=emotions[file_name.split("-")[2]]
```

```
if emotion not in observed_emotions:
continue
feature=extract_feature(file, mfcc=True, chroma=True,
mel=True)
data = noise(feature)
x.append(data)
y.append(emotion)
pd.DataFrame(x).to_excel("features.xlsx")
pd.DataFrame(y).to_excel("emotions.xlsx")
return train_test_split(np.array(x), y, test_size=test_size,
random_state=9)

pip install openpyxl

#DataFlair - Split the dataset
x_train,x_test,y_train,y_test=load_data(test_size=0.15)
#DataFlair - Get the shape of the training and testing datasets
print((x_train.shape[0], x_test.shape[0]))
(649, 115)
#DataFlair - Get the number of features extracted
print(f'Features extracted: {x_train.shape[1]}')
Features extracted: 180
#DataFlair - Initialize the Multi Layer Perceptron Classifier
model=MLPClassifier(alpha=0.01, batch_size=256, epsilon=1e-08,
hidden_layer_sizes=(256,), learning_rate='constant', max_iter=1024,
momentum = 0.9, activation='tanh')
#DataFlair - Train the model
model.fit(x_train,y_train)
MLPClassifier(activation='tanh', alpha=0.01, batch_size=256,
beta_1=0.9,
beta_2=0.999, early_stopping=False, epsilon=1e-08,
hidden_layer_sizes=(256,), learning_rate='constant',
learning_rate_init=0.001, max_iter=1024, momentum=0.9,
n_iter_no_change=10, nesterovs_momentum=True,
```

```
power_t=0.5,
random_state=None, shuffle=True, solver='adam',
tol=0.0001,
validation_fraction=0.1, verbose=False,
warm_start=False)
#DataFlair - Predict for the test set
y_pred=model.predict(x_test)
#DataFlair - Calculate the accuracy of our model
accuracy=accuracy_score(y_true=y_test, y_pred=y_pred)
#DataFlair - Print the accuracy
print("Accuracy: {:.2f}%".format(accuracy*100))
```

## 12 OUTPUT SCREENS

In [25]:

```
import librosa
import pandas as pd
import soundfile
import os, glob, pickle
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score
#from numba.core.decorators import jit as optional_jit
```

In [2]:

```
!pip install pandas
!pip install numpy
```

```
Requirement already satisfied: pandas in c:\users\dell\anaconda3\lib\site-packages (0.25.
1)
Requirement already satisfied: numpy>=1.13.3 in c:\users\dell\anaconda3\lib\site-packages
(from pandas) (1.16.5)
Requirement already satisfied: python-dateutil>=2.6.1 in c:\users\dell\anaconda3\lib\site
```

In [2]:

```
!pip install pandas
!pip install numpy
```

```
Requirement already satisfied: pandas in c:\users\dell\anaconda3\lib\site-packages (0.25.
1)
Requirement already satisfied: numpy>=1.13.3 in c:\users\dell\anaconda3\lib\site-packages
(from pandas) (1.16.5)
Requirement already satisfied: python-dateutil>=2.6.1 in c:\users\dell\anaconda3\lib\site
-packages (from pandas) (2.8.0)
Requirement already satisfied: pytz>=2017.2 in c:\users\dell\anaconda3\lib\site-packages
(from pandas) (2019.3)
Requirement already satisfied: six>=1.5 in c:\users\dell\anaconda3\lib\site-packages (fro
m python-dateutil>=2.6.1->pandas) (1.12.0)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
Requirement already satisfied: numpy in c:\users\dell\anaconda3\lib\site-packages (1.16.5
)
```

```
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
```

In [3]:

```
!pip install librosa
```

```
Requirement already satisfied: librosa in c:\users\dell\anaconda3\lib\site-packages (0.8.
1)
Requirement already satisfied: numba>=0.43.0 in c:\users\dell\anaconda3\lib\site-packages
(from librosa) (0.43.0)
Requirement already satisfied: scipy>=1.0.0 in c:\users\dell\anaconda3\lib\site-packages
(from librosa) (1.3.1)
Requirement already satisfied: packaging>=20.0 in c:\users\dell\anaconda3\lib\site-packag
es (from librosa) (21.0)
Requirement already satisfied: pooch>=1.0 in c:\users\dell\anaconda3\lib\site-packages (f
rom librosa) (1.5.1)
Requirement already satisfied: audioread>=2.0.0 in c:\users\dell\anaconda3\lib\site-packa
ges (from librosa) (2.1.9)
Requirement already satisfied: scikit-learn!=0.19.0,>=0.14.0 in c:\users\dell\anaconda3\l
```

In [3]:

```
!pip install librosa
```

```
Requirement already satisfied: librosa in c:\users\dell\anaconda3\lib\site-packages (0.8.
1)
Requirement already satisfied: numba>=0.43.0 in c:\users\dell\anaconda3\lib\site-packages
(from librosa) (0.43.0)
Requirement already satisfied: scipy>=1.0.0 in c:\users\dell\anaconda3\lib\site-packages
(from librosa) (1.3.1)
Requirement already satisfied: packaging>=20.0 in c:\users\dell\anaconda3\lib\site-packag
es (from librosa) (21.0)
Requirement already satisfied: pooch>=1.0 in c:\users\dell\anaconda3\lib\site-packages (f
rom librosa) (1.5.1)
Requirement already satisfied: audioread>=2.0.0 in c:\users\dell\anaconda3\lib\site-packa
ges (from librosa) (2.1.9)
Requirement already satisfied: scikit-learn!=0.19.0,>=0.14.0 in c:\users\dell\anaconda3\l
ib\site-packages (from librosa) (0.21.3)
Requirement already satisfied: decorator>=3.0.0 in c:\users\dell\anaconda3\lib\site-packa
ges (from librosa) (4.4.0)
Requirement already satisfied: numpy>=1.15.0 in c:\users\dell\anaconda3\lib\site-packages
(from librosa) (1.16.5)
Requirement already satisfied: joblib>=0.14 in c:\users\dell\anaconda3\lib\site-packages
(from librosa) (1.0.1)
Requirement already satisfied: resampy>=0.2.2 in c:\users\dell\anaconda3\lib\site-package
```

```
Requirement already satisfied: joblib>=0.14 in c:\users\dell\anaconda3\lib\site-packages
(from librosa) (1.0.1)
Requirement already satisfied: resampy>=0.2.2 in c:\users\dell\anaconda3\lib\site-package
s (from librosa) (0.2.2)
Requirement already satisfied: soundfile>=0.10.2 in c:\users\dell\anaconda3\lib\site-pack
ages (from librosa) (0.10.3.post1)
Requirement already satisfied: llvmlite>=0.28.0dev0 in c:\users\dell\anaconda3\lib\site-p
ackages (from numba>=0.43.0->librosa) (0.29.0)
Requirement already satisfied: pyparsing>=2.0.2 in c:\users\dell\anaconda3\lib\site-packa
ges (from packaging>=20.0->librosa) (2.4.2)
Requirement already satisfied: requests in c:\users\dell\anaconda3\lib\site-packages (fro
m pooch>=1.0->librosa) (2.22.0)
Requirement already satisfied: appdirs in c:\users\dell\anaconda3\lib\site-packages (from
pooch>=1.0->librosa) (1.4.4)
Requirement already satisfied: six>=1.3 in c:\users\dell\anaconda3\lib\site-packages (fro
m resampy>=0.2.2->librosa) (1.12.0)
Requirement already satisfied: cffi>=1.0 in c:\users\dell\anaconda3\lib\site-packages (fr
om soundfile>=0.10.2->librosa) (1.12.3)
Requirement already satisfied: pycparser in c:\users\dell\anaconda3\lib\site-packages (fr
om cffi>=1.0->soundfile>=0.10.2->librosa) (2.19)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in c:\users\dell\a
naconda3\lib\site-packages (from requests->pooch>=1.0->librosa) (1.24.2)
Requirement already satisfied: idna<2.9,>=2.5 in c:\users\dell\anaconda3\lib\site-package
s (from requests->pooch>=1.0->librosa) (2.8)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in c:\users\dell\anaconda3\lib\site-
packages (from requests->pooch>=1.0->librosa) (3.0.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\dell\anaconda3\lib\site-pac
kages (from requests->pooch>=1.0->librosa) (2019.9.11)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
```

In [23]:

```
!pip install numba==0.50
```

```
Collecting numba==0.50
  Using cached numba-0.50.0-cp37-cp37m-win_amd64.whl (2.2 MB)
Collecting llvmlite<0.34,>=0.33.0.dev0
  Using cached llvmlite-0.33.0-cp37-cp37m-win_amd64.whl (15.0 MB)
Requirement already satisfied: numpy>=1.15 in c:\users\dell\anaconda3\lib\site-packages (
from numba==0.50) (1.16.5)
Requirement already satisfied: setuptools in c:\users\dell\anaconda3\lib\site-packages (f
rom numba==0.50) (41.4.0)
Installing collected packages: llvmlite, numba
  Attempting uninstall: llvmlite
    Found existing installation: llvmlite 0.29.0
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
    WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packag
es)
ERROR: Cannot uninstall 'llvmlite'. It is a distutils installed project and thus we canno
t accurately determine which files belong to it which would lead to only a partial uninst
all.
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
```

```
    Found existing installation: llvmlite 0.29.0
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
    WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packag
es)
ERROR: Cannot uninstall 'llvmlite'. It is a distutils installed project and thus we canno
t accurately determine which files belong to it which would lead to only a partial uninst
all.
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: You are using pip version 21.2.4; however, version 21.3 is available.
You should consider upgrading via the 'c:\users\dell\anaconda3\python.exe -m pip install
--upgrade pip' command.
```

In [ ]:

```
!pip install llvmpy
```

In [ ]:

```
pip install numba
```

```
Requirement already satisfied: numba in /usr/local/lib/python3.7/dist-packages (0.50.0)
```

```
    Found existing installation: llvmlite 0.29.0
```

```
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
    WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packag
es)
ERROR: Cannot uninstall 'llvmlite'. It is a distutils installed project and thus we canno
t accurately determine which files belong to it which would lead to only a partial uninst
all.
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -umba (c:\users\dell\anaconda3\lib\site-packages)
WARNING: You are using pip version 21.2.4; however, version 21.3 is available.
You should consider upgrading via the 'c:\users\dell\anaconda3\python.exe -m pip install
--upgrade pip' command.
```

In [ ]:

```
!pip install llvmpy
```

In [ ]:

```
pip install numba
```

```
Requirement already satisfied: numba in /usr/local/lib/python3.7/dist-packages (0.50.0)
```

In [ ]:

```
!pip install llvmpy
```

In [ ]:

```
pip install numba
```

```
Requirement already satisfied: numba in /usr/local/lib/python3.7/dist-packages (0.50.0)
Requirement already satisfied: llvmlite<0.34,>=0.33.0.dev0 in /usr/local/lib/python3.7/di
st-packages (from numba) (0.33.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-packages (from
numba) (57.2.0)
Requirement already satisfied: numpy>=1.15 in /usr/local/lib/python3.7/dist-packages (fro
m numba) (1.19.5)
```

In [ ]:

In [ ]:

```
pip install cython
```

```
Requirement already satisfied: cython in /usr/local/lib/python3.7/dist-packages (0.29.23)
```

In [ ]:

```
npm install --global --production windows-build-tools
```

```
  File "<ipython-input-12-adf7faf36550>", line 1
    npm install --global --production windows-build-tools
               ^
SyntaxError: invalid syntax
```

In [ ]:

```
pip install npm
```

In [ ]:

```
npm install npm@latest -g
```

In [ ]:

```
pip install npm
```

In [ ]:

```
npm install npm@latest -g
```

In [ ]:

```
pip install --upgrade pip
```

In [26]:

```
#DataFlair - Extract features (mfcc, chroma, mel) from a sound file
def extract_feature(file_name, mfcc, chroma, mel):
    with soundfile.SoundFile(file_name) as sound_file:
        X = sound_file.read(dtype="float64")
        sample_rate=sound_file.samplerate
        if chroma:
            stft=np.abs(librosa.stft(X))
        result=np.array([])
        if mfcc:
```

In [26]:

```
#DataFlair - Extract features (mfcc, chroma, mel) from a sound file
def extract_feature(file_name, mfcc, chroma, mel):
    with soundfile.SoundFile(file_name) as sound_file:
        X = sound_file.read(dtype="float64")
        sample_rate=sound_file.samplerate
        if chroma:
            stft=np.abs(librosa.stft(X))
        result=np.array([])
        if mfcc:
            mfccs=np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=40).T, axis=0
)
            result=np.hstack((result, mfccs))
        if chroma:
            chroma=np.mean(librosa.feature.chroma_stft(S=stft, sr=sample_rate).T,axis=0)
            result=np.hstack((result, chroma))
        if mel:
            mel=np.mean(librosa.feature.melspectrogram(X, sr=sample_rate).T,axis=0)
            result=np.hstack((result, mel))
    return result
```

In [27]:

```
#DataFlair - Emotions in the RAVDESS dataset
emotions={
  '01':'crying',
  '02':'silent',
    '03':'noise',
    '04':'laugh'
}
#DataFlair - Emotions to observe
observed_emotions=['crying', 'silent','noise','laugh']
```

In [28]:

```
def noise(data):
    """
    Adding White Noise.
    """
    # you can take any distribution from https://docs.scipy.org/doc/numpy-1.13.0/referenc
e/routines.random.html
    noise_amp = 0.005*np.random.uniform()*np.amax(data)
    data = data.astype('float64') + noise_amp * np.random.normal(size=data.shape[0])
```

In [28]:

```
def noise(data):
    """
    Adding White Noise.
    """
    # you can take any distribution from https://docs.scipy.org/doc/numpy-1.13.0/reference/routines.random.html
    noise_amp = 0.005*np.random.uniform()*np.amax(data)
    data = data.astype('float64') + noise_amp * np.random.normal(size=data.shape[0])
    return data
```

In [29]:

```
#DataFlair - Load the data and extract features for each sound file
def load_data(test_size=0.15):
    x,y=[],[]
    for file in glob.glob("Actors\\Actor_*\\*.wav"):
        file_name=os.path.basename(file)
        emotion=emotions[file_name.split("-")[2]]
        if emotion not in observed_emotions:
            continue
```

In [29]:

```
#DataFlair - Load the data and extract features for each sound file
def load_data(test_size=0.15):
    x,y=[],[]
    for file in glob.glob("Actors\\Actor_*\\*.wav"):
        file_name=os.path.basename(file)
        emotion=emotions[file_name.split("-")[2]]
        if emotion not in observed_emotions:
            continue
        feature=extract_feature(file, mfcc=True, chroma=True, mel=True)
        data = noise(feature)
        x.append(data)
        y.append(emotion)
    pd.DataFrame(x).to_excel("features.xlsx")
    pd.DataFrame(y).to_excel("emotions.xlsx")
    return train_test_split(np.array(x), y, test_size=test_size, random_state=9)
```

In [27]:

```
pip install openpyxl
```

```
Requirement already satisfied: openpyxl in c:\users\dell\anaconda3\lib\site-packages (3.0
.0)
Requirement already satisfied: et-xmlfile in c:\users\dell\anaconda3\lib\site-packages (f
```

In [29]:

```
#DataFlair - Load the data and extract features for each sound file
def load_data(test_size=0.15):
    x,y=[],[]
    for file in glob.glob("Actors\\Actor_*\\*.wav"):
        file_name=os.path.basename(file)
        emotion=emotions[file_name.split("-")[2]]
        if emotion not in observed_emotions:
            continue
        feature=extract_feature(file, mfcc=True, chroma=True, mel=True)
        data = noise(feature)
        x.append(data)
        y.append(emotion)
    pd.DataFrame(x).to_excel("features.xlsx")
    pd.DataFrame(y).to_excel("emotions.xlsx")
    return train_test_split(np.array(x), y, test_size=test_size, random_state=9)
```

In [27]:

```
pip install openpyxl
```

```
Requirement already satisfied: openpyxl in c:\users\dell\anaconda3\lib\site-packages (3.0
.0)
Requirement already satisfied: et-xmlfile in c:\users\dell\anaconda3\lib\site-packages (f
```

In [30]:

```
#DataFlair - Split the dataset
x_train,x_test,y_train,y_test=load_data(test_size=0.15)
```

In [31]:

```
#DataFlair - Get the shape of the training and testing datasets
print((x_train.shape[0], x_test.shape[0]))
```

```
(240, 43)
```

In [32]:

```
#DataFlair - Get the number of features extracted
print(f'Features extracted: {x_train.shape[1]}')
```

```
Features extracted: 180
```

In [33]:

```
#DataFlair - Initialize the Multi Layer Perceptron Classifier
```

In [33]:

```
#DataFlair - Initialize the Multi Layer Perceptron Classifier
model=MLPClassifier(alpha=0.01, batch_size=256, epsilon=1e-08, hidden_layer_sizes=(256,),
learning_rate='constant', max_iter=1024, momentum = 0.9, activation='tanh')
```

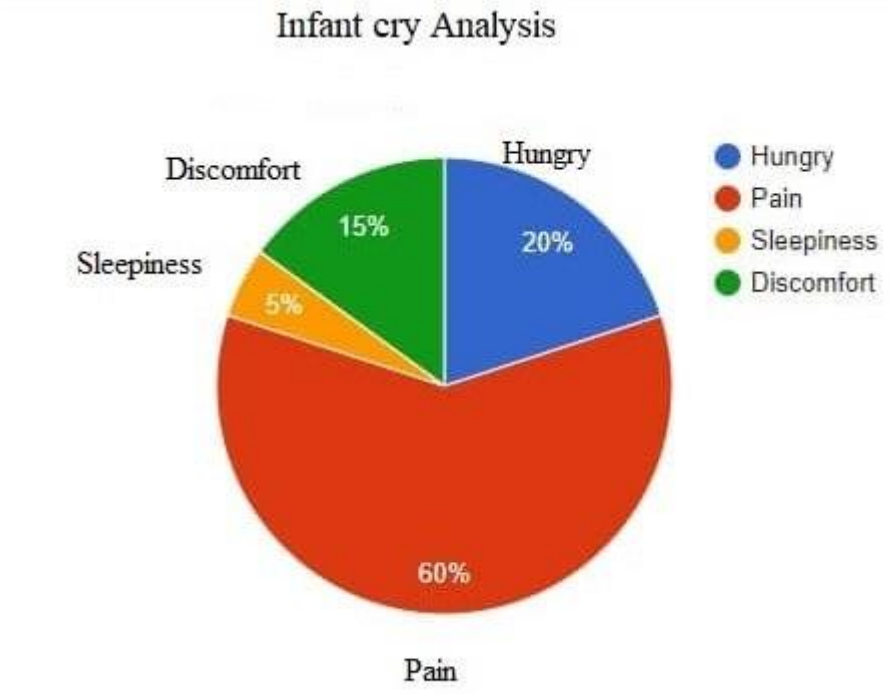
In [34]:

```
#DataFlair - Train the model
model.fit(x_train,y_train)
```

```
C:\Users\dell\Anaconda3\lib\site-packages\sklearn\neural_network\multilayer_perceptron.py
:350: UserWarning: Got `batch_size` less than 1 or larger than sample size. It is going t
o be clipped
  warnings.warn("Got `batch_size` less than 1 or larger than "
```

Out[34]:

```
MLPClassifier(activation='tanh', alpha=0.01, batch_size=256, beta_1=0.9,
              beta_2=0.999, early_stopping=False, epsilon=1e-08,
              hidden_layer_sizes=(256,), learning_rate='constant',
```



```
C:\Users\dell\Anaconda3\lib\site-packages\sklearn\neural_network\multilayer_perceptron.py
:350: UserWarning: Got `batch_size` less than 1 or larger than sample size. It is going t
o be clipped
  warnings.warn("Got `batch_size` less than 1 or larger than "
```

Out[34]:

```
MLPClassifier(activation='tanh', alpha=0.01, batch_size=256, beta_1=0.9,
              beta_2=0.999, early_stopping=False, epsilon=1e-08,
              hidden_layer_sizes=(256,), learning_rate='constant',
              learning_rate_init=0.001, max_iter=1024, momentum=0.9,
              n_iter_no_change=10, nesterovs_momentum=True, power_t=0.5,
              random_state=None, shuffle=True, solver='adam', tol=0.0001,
              validation_fraction=0.1, verbose=False, warm_start=False)
```

In [35]:

```
#DataFlair - Predict for the test set
y_pred=model.predict(x_test)
```

In [36]:

```
#DataFlair - Calculate the accuracy of our model
accuracy=accuracy_score(y_true=y_test, y_pred=y_pred)
#DataFlair - Print the accuracy
```

In [35]:

```
#DataFlair - Predict for the test set
y_pred=model.predict(x_test)
```

In [36]:

```
#DataFlair - Calculate the accuracy of our model
accuracy=accuracy_score(y_true=y_test, y_pred=y_pred)
#DataFlair - Print the accuracy
print("Accuracy: {:.2f}%".format(accuracy*100))
```

```
Accuracy: 95.35%
```

In [ ]:

```

```

## 13.CONCLUSION:

In this project, we explored how the Linear Discrimination Analysis algorithm can be worked which can be used to detect the crying sounds of infants using dataset which is retrieved from audio sample of subjects. First, we used the dataset that is available on gitHub, then we created a summarized version of the dataset to decrease the variations between the samples collected from the same subject in which we have done data preprocessing. We observed that we obtained higher accuracy with the algorithm that we have applied with the summarized dataset as presented in the paper.

The classification process within machine learning is automated not a standalone problem with a static training data and here we use testing data to classify and to know how it works. Rather, it is a complex dynamic process integrated with a screening tool in the presence of appropriate medical staff inside a clinical environment so that machine learning tools work much better and provide more accuracy. Our recent studies on different infant cry detection papers let us know that many machine learning algorithm are used and different set of features are extracted which help us know us there are different kind of diseases are also responsible for Infant cry apart from crying for normal situations like hungry, pain, fuzzy. In this project, we focused on recent machine learning studies that tackled infant cry detection as a classification problem and analysed their advantages and disadvantages.

The conclusions of our results are:-

- When we run the model for next line the measured or errors and accuracy might show a change.
- This is due to the model enhancements
- In this project we have used machine learning algorithm LDA which classified different features of infant cry with more accuracy.
- Future work can be extended by classifying the infant cry dataset with LDA algorithm which produced pain as reason the behind infant cry. This pain feature can be further

classified to know reason behind the pain like diseases etc. Accurate Classification may ease the drug discovery process.

## 14 References


1. Abou-Abbas, Lina and Tadj, Chakib and Fersaie, Hesam Alaie, "A fully automated approach for baby cry signal segmentation and boundary detection of expiratory and inspiratory episodes" , *The Journal of the Acoustical Society of America*, vol. 142, no.3, pp.1318- 1331,2017.

2. Cohen, Rami and Ruinskiy, Dima and Zickfeld, Janis and IJzerman, Hans and Lavner, Yizhar, "Baby cry detection: deep learning and classical approaches", *Development and Analysis of Deep Learning Architectures* pp. 171-196 Springer 2020.

3. Dewi, Sita Purnama and Prasasti, Anggunmeka Luhur and Irawan, Budhi, "The study of baby crying analysis using MFCC and LFCC in different classification methods", *2019 IEEE International Conference on Signals and Systems (ICSigSys)*, pp. 18-23, 2019.

4. Hariharan, M and Saraswathy, J and Sindhu, R and Khairunizam, Wan and Yaacob, Sazali, " Infant cry classification to identify asphyxia using time-frequency analysis and radial basis neural networks ", *Expert Systems with Applications*, vol. 39, no.10, pp.9515-9523, 2012.

5. Hariharan, Muthusamy and Sindhu, R and Yaacob, Sazali, "Normal and hypoacoustic infant cry signal classification using time--frequency analysis and general regression neural network", *Computer methods and programs in biomedicine*, vol. 108, no.2, pp.559-569 ,2012.

6. Ji, Chunyan and Mudiyanselage, Thosini Bamunu and Gao, Yutong and Pan, Yi, " A review of infant cry analysis and classification" *Eurasip Journal on Audio, Speech, and Music Processing*, vol. 202, no.1, pp 1-7, 2021.

7. Lavner, Yizhar and Cohen, Rami and Ruinskiy, Dima and IJzerman, Hans, "Baby cry detection in domestic environment using deep learning", *2016 IEEE international conference on the science of electrical engineering (ICSEE)*, pp 1-5, 2016

8. Severini, Marco and Ferretti, Daniele and Principi, Emanuele and Squartini, Stefano, " Automatic detection of cry sounds in neonatal intensive care units by using deep learning and acoustic scene simulation", *IEEE Access*, vol. 7, pp51982-51993, 2019.

9. Sharma, Shivam and Asthana, Shubham and Mittal, Vinay Kumar, "A database of infant cry sounds to study the likely cause of cry", *Proceedings of the 12th International Conference on Natural Language Processing*, pp.112-117, 2015.

DrillBit
Anti Plagiarism Software

The Report is Generated by DrillBit Plagiarism Detection Software

### *Submission Information*

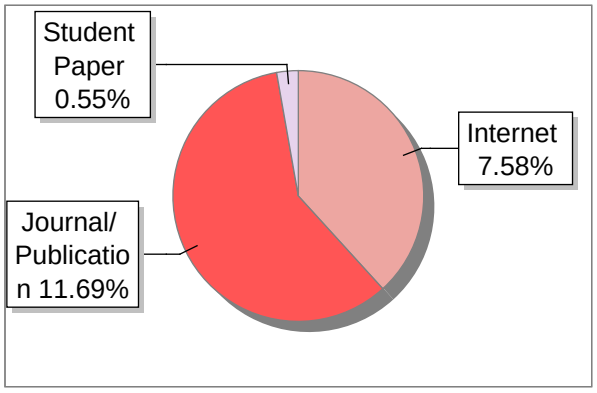
| | |
|---|---|
| Author Name | swarna |
| Title | cry |
| Paper/Submission ID | 413442 |
| Submission Date | 2021-11-22 12:17:42 |
| Total Pages | 48 |
| Document type | Article |

### *Result Information*

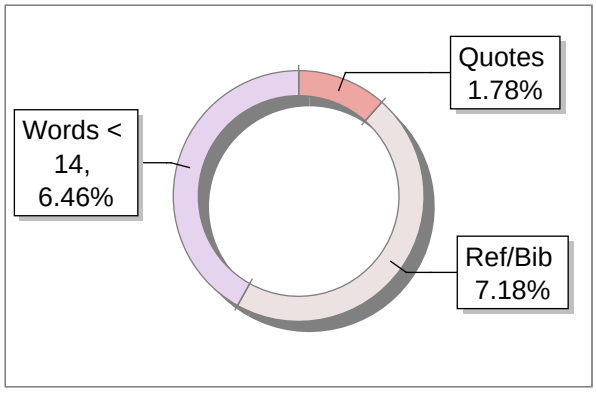
Similarity **14 %**

1 10 20 30 40 50 60 70 80 90

Sources Type




Report Content




***Alert..! Improper usage of 'Quotes' work in a report***

### *Exclude Information*

| | |
|---|---|
| References/Bibliography | Excluded |
| Quotes | Excluded |
| Sources: Less than 14 Words Similarity | Excluded |
| Excluded Source | **0 %** |

A Unique QR Code use to View/Download/Share Pdf File

DrillBit
Anti Plagiarism Software

**DrillBit Similarity Report**

| 14 | 10 | B |
|---|---|---|
| SIMILARITY % | MATCHED SOURCES | GRADE |

**A-Satisfactory (0-10%)**
**B-Upgrade (11-40%)**
**C-Poor (41-60%)**
**D-Unacceptable (61-100%)**

| LOCATION | MATCHED DOMAIN | % | SOURCE TYPE |
|---|---|---|---|
| 1 | Reading the signs Some basics of semiotics by DEELY-1993 | 3 | Publication |
| 2 | asmp-eurasipjournals.springeropen.com | 1 | Internet Data |
| 3 | fr.scribd.com | 2 | Internet Data |
| 4 | A fully automated approach for baby cry signal segmentation and boundary detecti by Abou-Abbas-2017 | 2 | Publication |
| 5 | data-flair.training | 2 | Internet Data |
| 6 | arxiv.org | 1 | Publication |
| 7 | Prevention and Fighting against Web Attacks through Anomaly Detection Technology by Sured-2020 | 1 | Publication |
| 8 | IEEE 2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure C, by Kotteti, Chandra Mo- 2018 | 1 | Publication |
| 9 | Corrigendum to Self-consistent modelling of Mercurys exosphere b by P-2010 | <1 | Publication |
| 10 | hal.archives-ouvertes.fr | <1 | Publication |