



Mag-sign in sa Google



Gamitin ang iyong Google Account para mag-sign in sa Medium



Wala nang tatandaang password. Mabilis, simple, at secure ang pag-sign in.

Magpatuloy



Rabeya Tus Sadia · [Follow](#)

4 min read · Mar 14, 2023

Listen

Share



Pomeranian working on an iPad from [Unsplash](#)

Crying is the primary means of communication between the baby and the outside world. When a baby is crying, it is difficult for a novice parent to understand the

baby's needs immediately. If parents can accurately determine the cause of the baby's cry, they can understand the baby's emotional and physiological changes and needs.

In this article, I will discuss how deep learning can be utilized to categorize better and label audio. I'll use my attempt at labeling a baby's cries as an illustration of this problem. In addition to being an entertaining experiment in CNN-based audio classification, this could be useful in the development of a monitor to alert parents that their baby is crying. Here is the [notebook](#) for this coding implementation of the baby cry classification.

## **Building dataset**

Obtaining a useful dataset was the first step in the classifier construction process. I was unable to locate a readily accessible dataset to create a model. So I decided to build one from data I collected from the internet. I used the [Donate-A-Cry corpus](#) for samples of sounds of babies crying. The dataset has about ~1000 sound clips of 7 secs in length. The dataset has five classes. They are 'hungry,' 'burping,' 'discomfort,' 'belly\_pain,' and 'tired.'

## **Building a model**

The audio clips have a sample rate of 16000 Hz and a duration of about ~7 secs. This means there are about  $16000 \times 7$  numbers per second representing the audio data. We take a fast fourier transform (FFT) of a 2048 sample window, slide it by 512 samples and repeat the process of the 7-sec clip. The resulting representation can be shown as a 2D image and is called a Short-Time Fourier Transform (STFT). Since humans perceive sound on a logarithmic scale, we'll convert the STFT to the [mel scale](#). The [librosa](#) library lets us load an audio file and convert it to a melspectrogram.

## **Coding Implementation**

Import necessary libraries for the implementation.

```
import matplotlib.pyplot as plt
from scipy import signal
from scipy.io import wavfile as wav
import numpy as np
from numpy.lib import stride_tricks
import torch
import torchvision
import cv2
import librosa
import librosa.display
import numpy as np
from sklearn.model_selection import train_test_split
import pandas as pd
import os
import pandas as pd
import matplotlib.pyplot as plt
from torch.utils.data import Dataset, DataLoader
from sklearn.model_selection import train_test_split
import cv2
from torchvision import transforms
import numpy as np
```

Import libraries

Utility functions to convert from audio to image.

## Utility functions for Audio-to-image

```
# Reading the audio file and applying some transformations (trimming, padding...)
```

```
def read_audio(conf, pathname, trim_long_data):
    y, sr = librosa.load(pathname, sr=conf.sampling_rate)
    # trim silence
    if 0 < len(y): # workaround: 0 length causes error
        y, _ = librosa.effects.trim(y) # trim, top_db=default(60)
    # make it unified length to conf.samples
    if len(y) > conf.samples: # long enough
        if trim_long_data:
            y = y[0:0+conf.samples]
    else: # pad blank
        padding = conf.samples - len(y)      # add padding at both ends
        offset = padding // 2
        y = np.pad(y, (offset, conf.samples - len(y) - offset), 'constant')
```

Convert audio to image

Utilizing librosa library, generating the mel-spectrogram from the audio file.

```
def audio_to_melspectrogram(conf, audio):
    spectrogram = librosa.feature.melspectrogram(audio,
                                                sr=conf.sampling_rate,
                                                n_mels=conf.n_mels,
                                                hop_length=conf.hop_length,
                                                n_fft=conf.n_fft,
                                                fmin=conf.fmin,
                                                fmax=conf.fmax)
    spectrogram = librosa.power_to_db(spectrogram)
    spectrogram = spectrogram.astype(np.float32)
    return spectrogram
```

A set of settings that you can adapt to fit your audio files (frequency, average duration, number of Fourier transforms)

```

class conf:
    # Preprocessing settings
    sampling_rate = 16000
    duration = 7
    hop_length = 100*duration |
    fmin = 20
    fmax = sampling_rate // 2
    n_mels = 128
    n_fft = n_mels * 20
    samples = sampling_rate * duration

```

To generate the image dataset, we first need to remove the .wav extension and add the .jpg.

```

def rename_file(img_name):
    img_name = img_name.split("/")[-1]
    img_name = img_name[:-4]
    img_name += ".jpg"
    return img_name

```

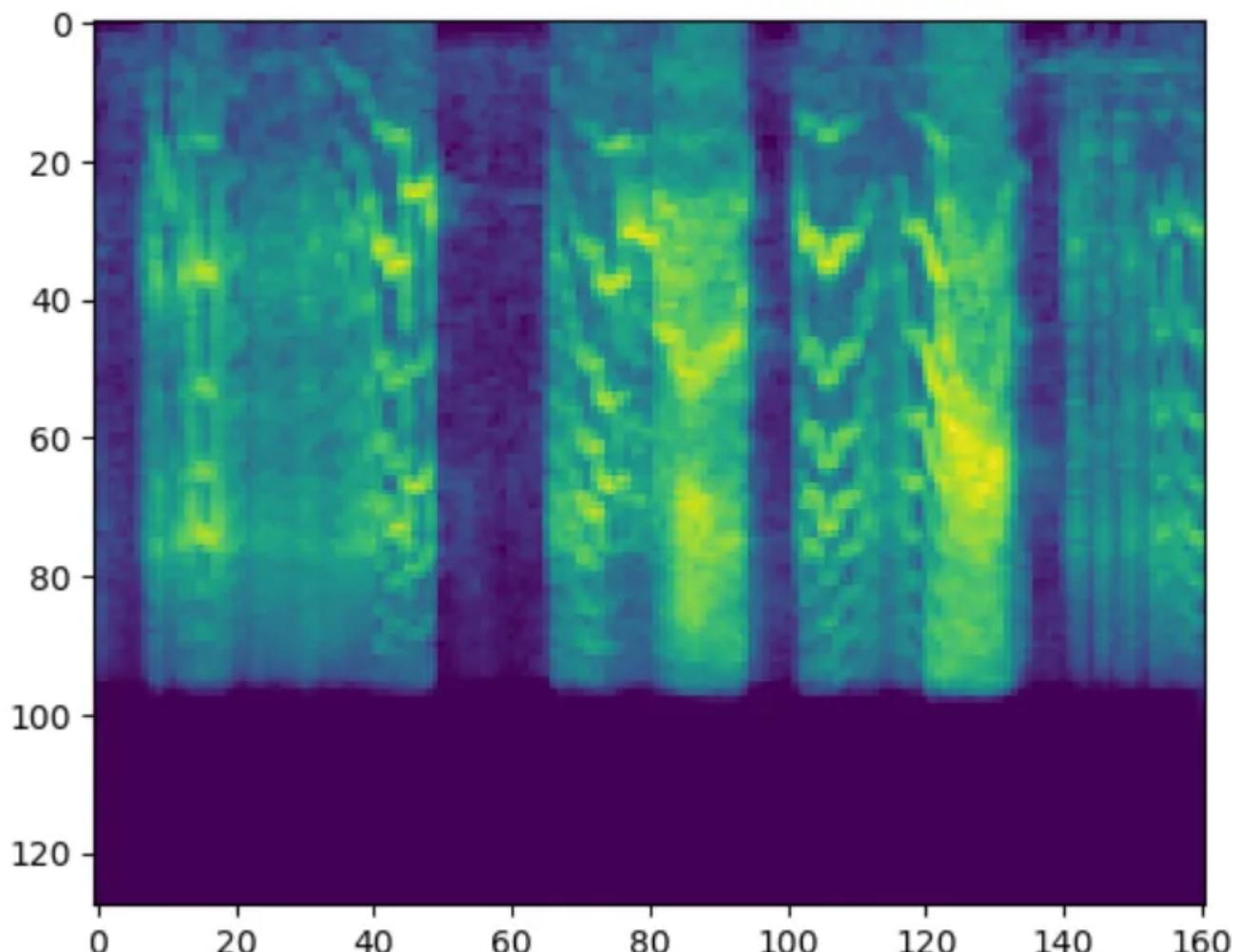
Set the converted image to categorized folder.

```

to_folders=['hungry', 'burping', 'discomfort', 'belly_pain', 'tired']
for to_folder in to_folders:
    try:
        os.makedirs('/kaggle/working/'+to_folder)
    except:
        continue
    for i, fn in enumerate(os.listdir('/kaggle/input/donate-cry-updated-dataset/c'))
        path = '/kaggle/input/donate-cry-updated-dataset/donateacry-corpus-master'
        save_image_from_sound(path,to_folder)

```

The melspectrogram of a baby crying looks like the image below:



---

The dataset is divided into train and test(75:25) into their respective folders.

## Train\_test\_split

```
# Take the source folder
source = r'/kaggle/input/donateacrycorpusfeaturesdataset/donateacrycorpus_features

X_train,X_test,Y_train,Y_test=train_test_split(all_img,all_img_label,train_size=0.7)
X_train = np.array(X_train)
X_test = np.array(X_test)
Y_train= np.array(Y_train)
Y_test= np.array(Y_test)
print(X_train.shape,Y_train.shape)
print(X_test.shape,Y_test.shape)
print(Y_train[0])
```

```
(342,) (342,)
(115,) (115,)
3
```

## Training Model

```
model = models.resnet50(pretrained=True)
num_ftrs = model.fc.in_features
model.fc = torch.nn.Linear(num_ftrs,5)
device=torch.device('cuda' if torch.cuda.is_available() else 'cpu')
model=model.to(device)
```

Downloading: "<https://download.pytorch.org/models/resnet50-0676ba61.pth>" to /root/.cache/checkpoints/resnet50-0676ba61.pth

100% [██████████] 97.8M/97.8M [00:00<00:00, 106MB/s]

Training accuracy:

---

```
epoch: 0 Accuracy = 81.28655242919922
epoch: 0 loss: 12.92688517398201
epoch: 1 Accuracy = 84.21052551269531
epoch: 1 loss: 8.40015251338482
epoch: 2 Accuracy = 84.50292205810547
epoch: 2 loss: 7.265490334481001
epoch: 3 Accuracy = 85.67251586914062
epoch: 3 loss: 6.4702746033668515
epoch: 4 Accuracy = 89.18128967285156
epoch: 4 loss: 5.418753217160702
epoch: 5 Accuracy = 91.52046966552734
epoch: 5 loss: 3.588867117650807
epoch: 6 Accuracy = 93.85964965820312
epoch: 6 loss: 3.1710626550018786
epoch: 7 Accuracy = 95.6140365600586
epoch: 7 loss: 2.2358758419752123
epoch: 8 Accuracy = 97.36842346191406
epoch: 8 loss: 1.6652651662006974
epoch: 9 Accuracy = 98.83040618896484
epoch: 9 loss: 0.9571665681432933
Finished Training
```

Testing accuracy

```
correct=0.0
for i,(inputs,labels) in enumerate(dataloader['test']):
    inputs=inputs.to(device)
    labels=labels.to(device)

    # zero the parameter gradients
    outputs = model(inputs)

    _, predicted = torch.max(outputs.data, 1)
    correct += (predicted == labels).float().sum()
#    print(correct)

accuracy = 100 * correct / len(test_set)

print("epoch:",epoch,"Accuracy = {}".format(accuracy))
# print statistics
## e.g:dataset=500, batch_size=10, #batch=50
#    print("Loss: ", loss.item())
print('Finished Test')
```

epoch: 0 Accuracy = 80.0

Finally, the accuracy of the baby crying classification is 80%. After a lot of tuning and modification, this accuracy could be increased.

Audio To Image Convert

Classification

Machine Learning

Baby Cry Classification



Follow



## Written by Rabeya Tus Sadia

66 Followers

Research Assistant, PhD student at University of Kentucky

---

More from Rabeya Tus Sadia



 Rabeya Tus Sadia in [Becoming Human: Artificial Intelligence Magazine](#)

## Understanding Graph Neural Network with hands-on example| Part-2

Welcome back to the next part of this Blog Series on Graph Neural Networks!

7 min read · Jul 20, 2021

 779





# Cheat Sheets for AI

## Neural Networks, Machine Learning, DeepLearning & Big Data

The Most Complete List  
of Best AI Cheat Sheets



Stefan Kojouharov in [Becoming Human: Artificial Intelligence Magazine](#)

## Downloadable: Cheat Sheets for AI, Neural Networks, Machine Learning, Deep Learning & Data Science...

Downloadable PDF of Best AI Cheat Sheets in Super High Definition

9 min read · Mar 22, 2019



4K



23

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec a purus odio.  
maximus ante non consecetur ultricies. Nunc imperdiet laetus. Integer facilisis nunc in mauris suscipit commodo quis ac nisi malesuada, suscipit porta purus vestibulum. Vivamus rutrum ante dignissim, eleifend ante nec, tempus velit. Nam blandit ultrices nisl. Vestibulum vel facilisis ligula. Etiam cursus, laetus at ullamcorper finibus, lectus augue porta felis, scelerisque vehicula turpis enim ut nibh. Donec tincidunt leo arcu, quis feugiat augue semper ut. Etiam imperdiet enim non ornare ultrices. Donec sollicitudin elit nulla, sit amet suscipit lorem gravida id. Proin dictum urna eget nisi semper congue. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis a purus odio.

Mauris ultrices, nibh sit amet volutpat viverra, turpis turpis tristique nibh, quis dapibus metus ante id purus. Suspendisse feugiat commodo nisl, vel blandit risus placerat eu. Phasellus laoreet ac ipsum sed maximus. Fusce risus massa, faucibus at felis quis, posuere ornare arcu. Phasellus convallis lobortis elit a luctus. Donec et eros ut enim varius efficitur. Phasellus sit amet nisi facilisis, efficitur dolor sed, effictur est. Phasellus nec lectus vitae arcu commodo cursus quis sit amet orci. Sed nec egestas ante, id hendrerit lectus. Nulla in dapibus tellus. Maecenas non odio suscipit, commodo ipsum eu, facilisis orci. Donec et finibus felis. Orci varius natoque penatibus et magnis dis partant montes, nascetur ridiculus mus.

Mauris ac nulla commodo, maximus risus at, ornare ante. Sed eleifend odio a magna euismod ultrices. Ut semper dui justo, id aliquam erat convallis sit amet. Etiam blandit non est id cursus. Maecenas semper eleifend libero et molestie. Nulla mattis nulla eu enim dignissim, ac consequat nulla ultrices. Nulla nec est tincidunt, faucibus enim vitae, vehicula neque. Nulla tincidunt ornare ex eget porta.

Sed sed sagittis erat. Aenean eget velit quis tellus pulvinar aliquam. Etiam pharetra molestie mi, pulvinar semper nisl porta quis. Suspendisse vulputate, ipsum vitae posuere sollicitudin, lectus sapien luctus neque, blandit finibus lorem magna nec turpis. Vestibulum ac pellentesque eros, in laoreet ante. Praesent vitae dictum odio, ut venenatis mi. Fusce sit amet accumsan augue. Integer nec varius sapien, nec scelerisque erat. Ut feugiat dui ac mauris placerat efficitur. Nunc vel ullamcorper augue.

Cras faucibus, toror ut tincidunt venenatis, nisl elit pretium dolor, eget lacinia ex lectus sed urna. Vestibulum nec iaculis urna. Donec eget arcu quis massa porttitor pellentesque. Vestibulum porta turpis ac augue gravida posuere. Suspendisse dictum, sapien eget vulputate auctor, lectus libero tempus odio, egestas ultricies ligula orci at arcu. Vestibulum eget ultricies dui. Nunc rutrum neque non erat mollis ullamcorper. Suspendisse viverra, mauris vel molestie suscipit, ex tellus vehicula sem, id efficitur nunc ipsum vel elit. Curabitur tincidunt pretium mauris at elementum. Cras non mollis lectus. Aliquam malesuada varius placerat. Interdum et malesuada fames ac ante ipsum primis in faucibus. Nunc et laoreet orci. Ut non malesuada justo, non bilendum urna.

Sed mollis nulla urna, at suscipit libero vehicula vitae. Suspendisse in condimentum nisl. Mauris volutpat, augue ut ullamcorper sagittis, toror dolor aliquet dolor, eget ultricies metus dui ac eros.

taciturnus eget id dolor. Quisque vitae quam suscipit, suscipit curabitur eu ornare mauris. Nam

imperfecti laici. Integer facilisis nunc in mauris suscipit commodo quis ac elit. Cras tempus odio at nisl malesuada, suscipit porta purus vestibulum. Vivamus rutrum ante dignissim, eleifend ante nec, tempus velit. Nam blandit ultrices nisl. Vestibulum vel facilisis ligula. Etiam cursus, laetus at ullamcorper finibus, lectus augue porta felis, scelerisque vehicula turpis enim ut nibh. Donec tincidunt leo arcu, quis feugiat augue semper ut. Etiam imperdiet enim non ornare ultrices. Donec sollicitudin elit nulla, sit amet suscipit lorem gravida id. Proin dictum urna eget nisi semper congue. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis a purus odio.

Mauris ultrices, nibh sit amet volutpat viverra, turpis turpis tristique nibh, quis dapibus metus ante id purus. Suspendisse feugiat commodo nisl, vel blandit risus placerat eu. Phasellus laoreet ac ipsum sed maximus. Fusce risus massa, faucibus at felis quis, posuere ornare arcu. Phasellus convallis lobortis elit a luctus. Donec et eros ut enim varius efficitur. Phasellus sit amet nisi facilisis, efficitur dolor sed, effictur est. Phasellus nec lectus vitae arcu commodo cursus quis sit amet orci. Sed nec egestas ante, id hendrerit lectus. Nulla in dapibus tellus. Maecenas non odio suscipit, commodo ipsum eu, facilisis orci. Donec et finibus felis. Orci varius natoque penatibus et magnis dis partant montes, nascetur ridiculus mus.

Mauris ac nulla commodo, maximus risus at, ornare ante. Sed eleifend odio a magna euismod ultrices. Ut semper dui justo, id aliquam erat convallis sit amet. Etiam blandit non est id cursus. Maecenas semper eleifend libero et molestie. Nulla mattis nulla eu enim dignissim, ac consequat nulla ultrices. Nulla nec est tincidunt, faucibus enim vitae, vehicula neque. Nulla tincidunt ornare ex eget porta.

Sed sed sagittis erat. Aenean eget velit quis tellus pulvinar aliquam. Etiam pharetra molestie mi, pulvinar semper nisl porta quis. Suspendisse vulputate, ipsum vitae posuere sollicitudin, lectus sapien luctus neque, blandit finibus lorem magna nec turpis. Vestibulum ac pellentesque eros, in laoreet ante. Praesent vitae dictum odio, ut venenatis mi. Fusce sit amet accumsan augue. Integer nec varius sapien, nec scelerisque erat. Ut feugiat dui ac mauris placerat efficitur. Nunc vel ullamcorper augue.

Cras faucibus, toror ut tincidunt venenatis, nisl elit pretium dolor, eget lacinia ex lectus sed urna. Vestibulum nec iaculis urna. Donec eget arcu quis massa porttitor pellentesque. Vestibulum porta turpis ac augue gravida posuere. Suspendisse dictum, sapien eget vulputate auctor, lectus libero tempus odio, egestas ultricies ligula orci at arcu. Vestibulum eget ultricies dui. Nunc rutrum neque non erat mollis ullamcorper. Suspendisse viverra, mauris vel molestie suscipit, ex tellus vehicula sem, id efficitur nunc ipsum vel elit. Curabitur tincidunt pretium mauris at elementum. Cras non mollis lectus. Aliquam malesuada varius placerat. Interdum et malesuada fames ac ante ipsum primis in faucibus. Nunc et laoreet orci. Ut non malesuada justo, non bilendum urna.

Sed mollis nulla urna, at suscipit libero vehicula vitae. Suspendisse in condimentum nisl. Mauris volutpat, augue ut ullamcorper sagittis, toror dolor aliquet dolor, eget ultricies metus dui ac eros. Phasellus id lorem nec mauris faucibus luctus eget id dolor. Quisque vitae quam suscipit, suscipit

 Leo Ertuna in [Becoming Human: Artificial Intelligence Magazine](#)

## How to automatically deskew (straighten) a text image using OpenCV

Today I would like to share with you a simple solution to image deskewing problem (straightening a rotated image) with Python and OpenCV

6 min read · Sep 6, 2020

 216  3



 Rabeya Tus Sadia in [Becoming Human: Artificial Intelligence Magazine](#)

## Understanding Graph Neural Network with hands-on example| Part-1

Hello and welcome to this post, in which I will study a relatively new field in deep learning involving graphs—a very important and...

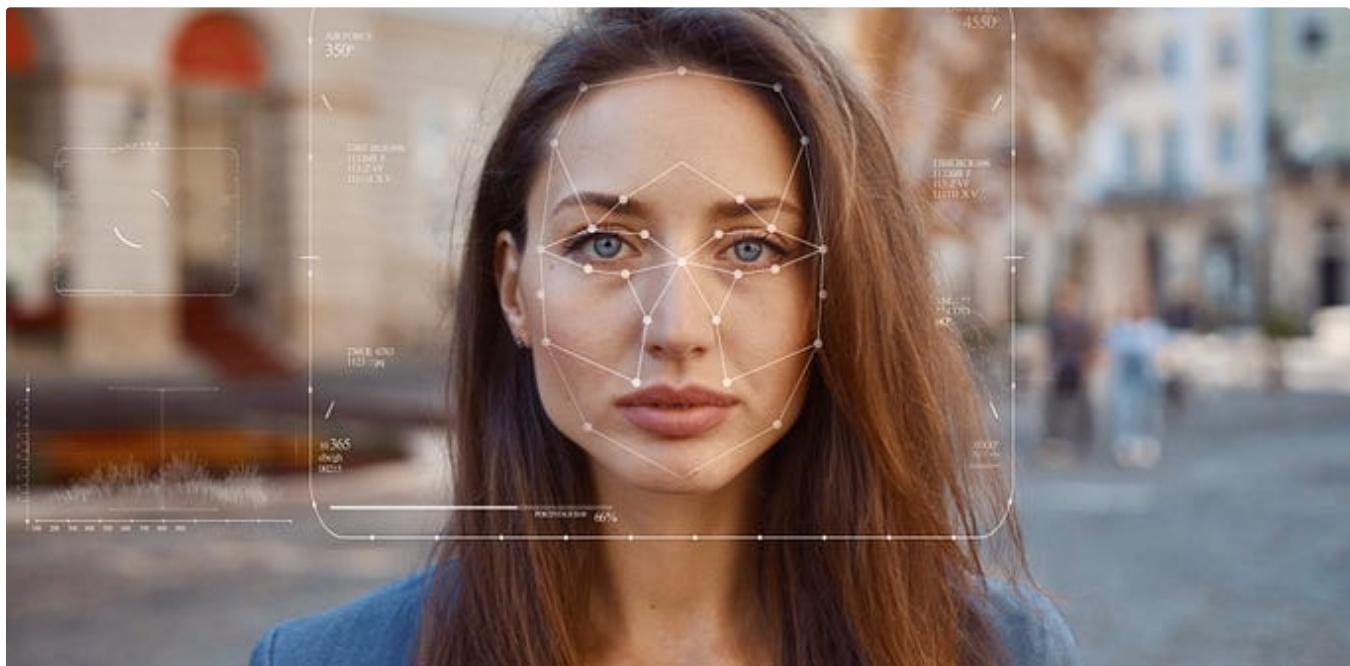
7 min read · Jul 20, 2021

 700 



See all from Rabeya Tus Sadia

## Recommended from Medium



Zaid Khan

Real-time Facial Emotion Recognition using Deep Learning and OpenCV

Learning how to build a convolutional neural network to detect real-time facial emotions.

11 min read · Nov 16, 2023



Q 2



 Mujtaba Raza

## Voice Classification Using MFCC Features and Deep Neural Networks: A Step-by-Step Guide

Introduction:

3 min read · Dec 3, 2023



4



---

### Lists



#### Predictive Modeling w/ Python

20 stories · 1101 saves



#### Practical Guides to Machine Learning

10 stories · 1315 saves



#### Natural Language Processing

1377 stories · 870 saves



#### The New Chatbots: ChatGPT, Bard, and Beyond

12 stories · 358 saves

05	06	07	08	09	10	11	...	tonnetz
...	...	...	...	...	...	...	...	std
1.482478	0.531371	1.481593	2.691455	0.866868	1.341231	1.347792	...	0.054125
1.654031	0.067592	1.366848	1.054094	0.108103	0.619185	1.038253	...	0.063831
1.937570	0.880839	-0.923192	-0.927232	0.666617	1.038546	0.268932	...	0.040730
1.367364	0.998411	1.770694	1.604566	0.521217	1.982386	4.326824	...	0.074358
0.775204	0.084794	-0.289294	-0.818410	0.043851	-0.804761	-0.990958	...	0.095003
...	...	...	...	...	...	...	...	...
1.662914	2.115189	-0.237794	5.695442	0.830353	1.951819	-0.190785	...	0.128410
0.130612	-0.263825	-0.628103	-0.082687	-0.229483	-0.492753	-0.746905	...	0.132964
-0.224011	-0.530972	1.713526	1.418444	1.325197	0.120333	1.307971	...	0.108324
0.066005	-0.857354	-0.780860	0.626281	-0.630938	-0.787229	1.402555	...	0.088311
-0.249835	0.360283	-0.366701	0.033578	-0.834606	-1.154845	-0.834298	...	0.091421

 Kaavya Mahajan

## Extracting audio features using Librosa

From the top 2500 audios of the last 50 years, I extract a few relevant features using Librosa, for the model I have trained.

7 min read · Oct 20, 2023

 100

 1




 Serkan Celik in Huawei Developers

## Basics of Audio Processing

Hi! In this article, we will talk about the basic components of MindAudio and the audio pre-processing stages.

6 min read · Dec 8, 2023



65

 Apparao Mulpuri

## Speaker Diarization in Python: A Step-by-Step Guide

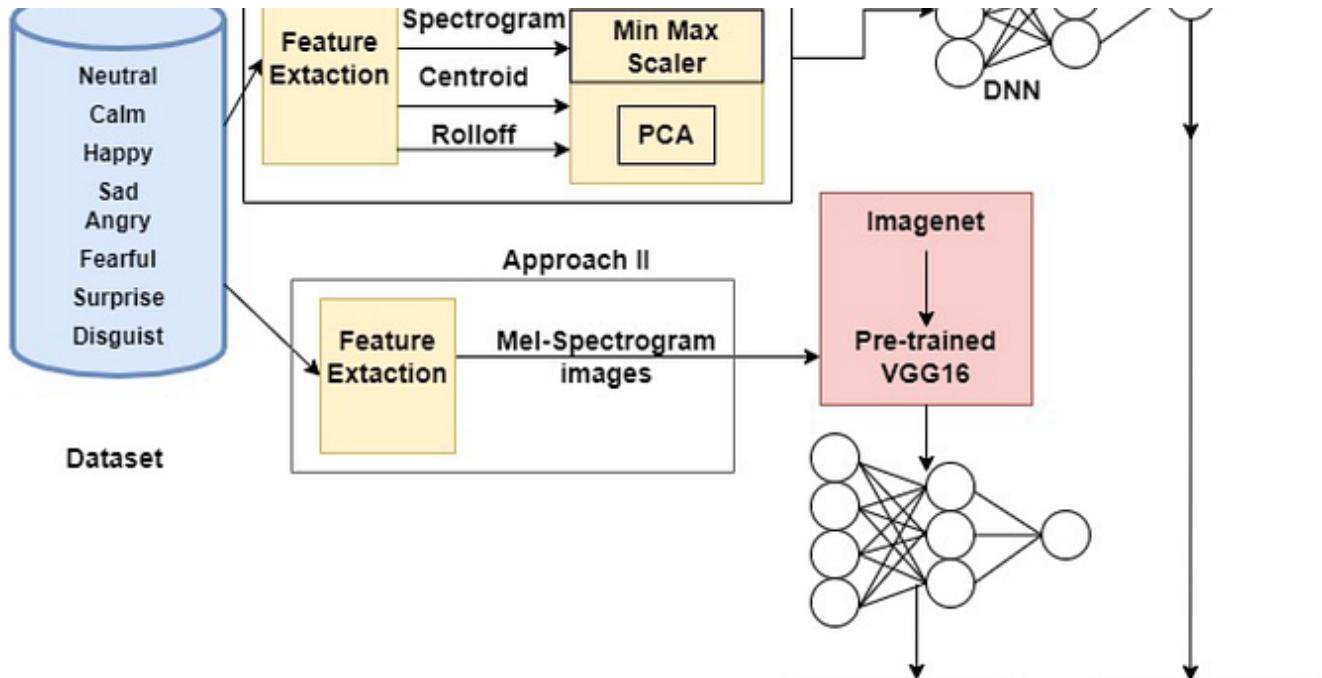
Introduction

2 min read · Nov 28, 2023



80





Muhammad Abdullah

## Voice Classification Using MFCC Features and DNN

Voice classification is the task of identifying the speaker of a voice recording. This has a wide range of applications, including speaker...

4 min read · Dec 3, 2023



53



See more recommendations