

GENERATING A MUSIC PLAYLIST USING FACIAL EXPRESSIONS

Kalle Sravani¹, Maddu Haindavi², Thurerao Divya³, Sumera⁴

^{1,2,3,4}Stanley College of Engineering and Technology For Women, Hyderabad, Telangana, India.

DOI: <https://www.doi.org/10.58257/IJPREMS39534>

ABSTRACT

To listen to music is the favorite pastime of many people, each of who have their own preferred type of music. Users always face the problem of searching through their folders to find the right music to fit their mood, and to make a list of their favorite songs. The facial expressions project attempts to help form a mechanized and automated system for personalizing music to a user's emotional state. To help aid this goal, this project provides the framework for gathering songs and the user's emotional state at the same time so the songs for each orchestrated emotion can be precisely generated later. A webcam captures facial expressions which are processed by a learning algorithm capable of inferring the most likely emotion corresponding to the detected facial expression. The already known expression is then identified and a corresponding playlist is created in advance style. The created playlist gets read by Spotify API and thus saves user time to search for songs that are most often selected for a particular mood. When selecting the required songs, the API sorts them according to the mood-denoting emotion of joy, sadness, anger or neutral which is detected Spark API.

Keywords: Face Emotion Detection, Emotion Identification, Music Playlist, Convolutional Neural Networks, Spotify Application List Interface,

1. INTRODUCTION

While music is crucial for day-to-day activities, looking for new songs often proves to be a daunting task. This project uses computer vision to automatically select music based on the user's facial expression. Computer vision allows for the interpretation of digital image data by a computer such that meaningful information can be extracted for use in machine learning. A webcam captures facial expressions that are processed to establish what emotional state a user is in. In our system, we use the facial expression recognition approach based on the Haar Cascade classifier, which is an object detection framework introduced by Viola and Jones. Emotion recognition is accomplished with Convolutional Neural Networks (CNNs), which have been shown to classify images with high accuracy. CNN is a multi-layer, feedforward neural network that relies on the complex mechanisms of the brain's nervous system, and mimics the human visual system, recognizing hierarchical spatial features through the following processes: convolution and pooling, and finally fully connected layers. The identified emotion is associated with a designated playlist which is automatically played. The use of Haar Cascade for face detection and CNN for feature extraction guarantees performance in practical situations. This allows for an improvement in user experience and makes it easier to browse for music. Because of the neural network's ability to discern patterns within an image's data, the task becomes more efficiently executed and the processing speed of the system increases.

2. LITERATURE SURVEY

The literature review gives an ideological ground to this project by considering different methods of generating music playlists from facial expressions.

Classic Music Recommendation Systems:

Classic music recommendation systems are based on user input in the forms of manual playlist generation or genre- or popularity-based song selection. Although efficient, these systems involve active user participation and fail to respond to real-time emotional fluctuations, which restrict their capacity to deliver a custom experience.

Facial Expression-Based Emotion Recognition:

Computer vision and deep learning advancements have made emotion detection possible via facial expressions. Researchers have utilized datasets such as FER-2013, AffectNet, and EmotiW to train Convolutional Neural Networks (CNNs) for emotion recognition like happiness, sadness, anger, and surprise. Although CNN models perform with high accuracy, their success relies on proper training, extensive labeled datasets, and strong environmental adaptability.

Machine Learning in Music Recommendation:

Machine learning algorithms, such as deep neural networks and hybrid emotion models, have been used to make music recommendations personalized. These systems examine facial expressions and translate them into mood-based playlists. Challenges such as real-time processing, generalization across different facial structures, and changing lighting conditions impact performance.

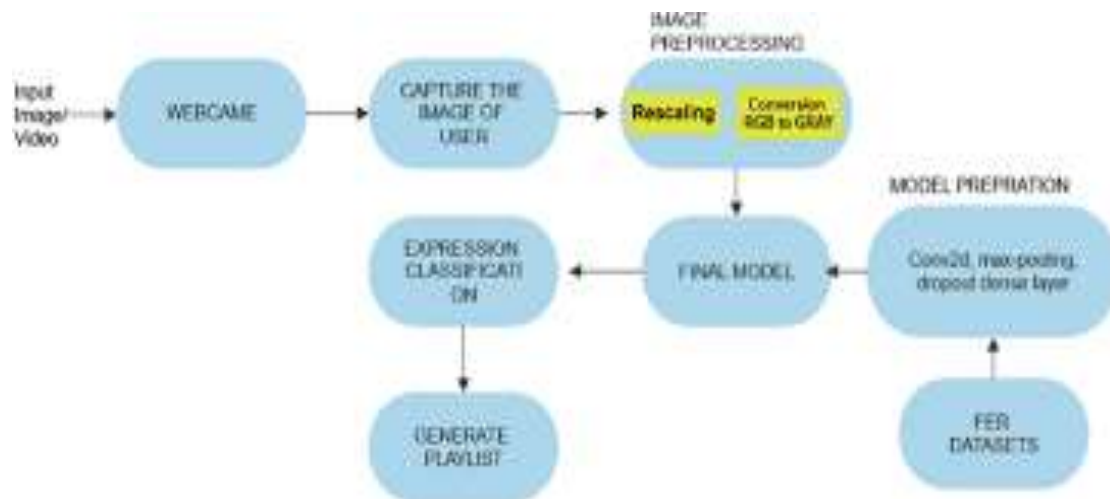
API Integration for Automated Playlist Generation:

Integration with music streaming services such as Spotify API has enabled real-time playlist generation according to the emotions detected. Such APIs utilize pre-defined song categories that correlate with emotions to enhance user experience. Even though they are efficient, API-based systems demand reliable internet connections and proper mapping of emotional states and music genres.

3. AIMS AND OBJECTIVES

This project aims to create an emotion-aware music recommendation system that autonomously constructs playlists with multi-faceted user experience through emotive facial expression tracking. This system automatically detects and recognizes emotions such as happiness, sadness, anger, surprise, and neutral using CNNs and Haar cascade self-classification methods. The emotion is then recalled with a set playlist and automatically played without requiring the user to select songs. In addition, a music streaming API like the Spotify API is incorporated into the project to enable effortless streaming during the optimization of deep learning models for speed and accuracy. System performance is analyzed using precision, time efficiency, and dependability, focusing on the ultimate goal of simplifying the music recommendation process.

4. SYSTEM ARCHITECTURE



The illustration shows a music recommendation system which works with facial recognition and deep learning technologies, operating based on emotions. The procedure starts with taking a picture as an input. The picture is then sent to a Convolutional Neural Network where the image is screened for a face. When a face is found in the image, the appropriate features of the face are captured to assess the user's emotions. These captured features are then applied to emotion recognition, the process in which a person's feeling such as happiness, sadness or anger is determined. After determining the user's emotions, a set of songs that match the user's mood is categorically selected from a database of songs using a music classifier. The resulting playlist is recommended as an output of the system, along with the categorized song which is played to the user. This system automatically chooses the songs and increases users satisfaction allowing them enjoy songs corresponding to their mood. And, with the use of deep learning, computer vision, and music classification, this framework gives the user a personalized music experience.

5. ALGORITHM

1) Capture Facial Expression

- Take a photo or capture a real-time video of the user's face.

2) Preprocess Image

- Convert the image to black and white.
- Adjust the image for optimal model precision.

3) Facial Feature Detection

- Analyze facial landmarks using a pre trained deep learning model (OpenCV, DeepFace, TensorFlow, etc).
Find critical areas such as eyes, eyebrows, and mouth.

4) Expression Classification

- Use an emotion classification model (CNN based) to recognize faces towards happy, sad, angry, surprised, calm, neutral, or any other defined emotions.

5) Map Emotion to Music Genre

- Emotion Happy is equal to Pop, Dance, Upbeat Songs. Sad is equals to Slow, Acoustic, Blues. Anger equals to Rock, Metal, Rap. Surprise equals to Electronic, Experimental. Calm equals to Lo-Fi, Classical, Jazz.

6) Generate Playlist

- Search the music database and select songs that represent the category of emotion detected.
- Make a playlist with a set number of songs (10) and return it.

7) Display Playlist & Play Music

- Show the users generated playlist. Give users the option to skip or change songs.

6. USED TECHNOLOGY

Frontend Development:

Hyper Text Markup Language (HTML): Presents the web pages structure and its essentials.

Cascading Style Sheets (CSS): Manages graphical details and the designs of the interface.

JavaScript (JS): Provides the required actions and operates on the website.

Backend Development:

Python: The foremost programming language that has been utilized in backend management.

Django (Python Framework): A web framework of high level that permits quick construction and is scalable while being secured.

Development Environment Tools:

PyCharm IDE: An integrated development environment for Python that is strong and provides debugging, code summarizes, and other indispensable items.

Database Management System:

SQLite/PostgreSQL: A database that is employed with Django ORM for its simple querying and for data utilization and supervision.

Versioning Control System:

Git & GitHub: Utilized in monitoring the progress made, collaboration, and versioning of the code.

APIs and Additional Software:

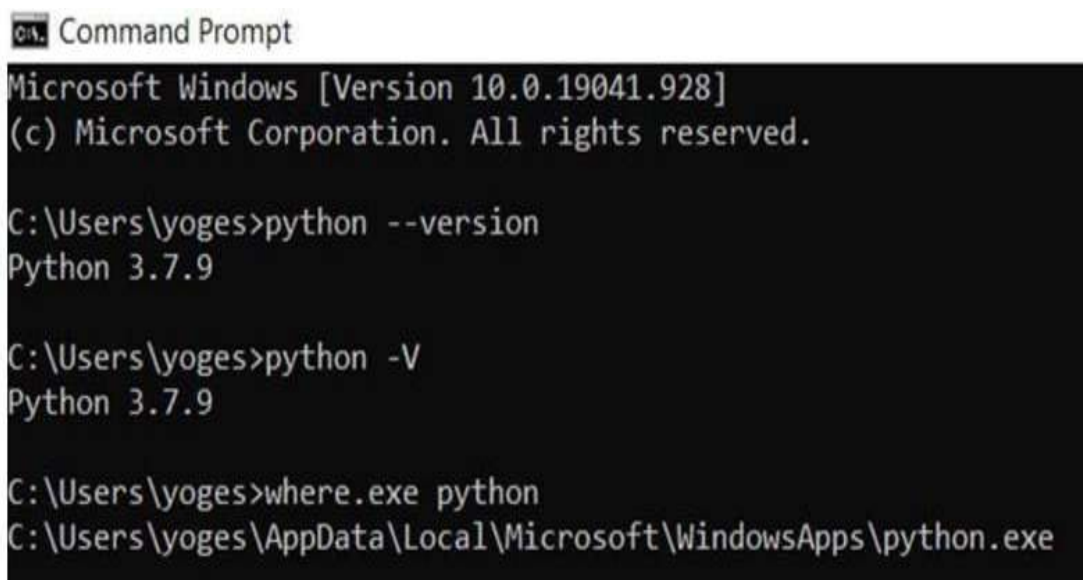
OpenCV: Employed for recognizing facial features and determining expression.\

TensorFlow/Keras: Deep learning construction used for the comprehension of facial expressions and their attribution to salient features.

Media APIs: Used to link the services of music streaming e.g. Spotify API.

7. MODULES AND RESULTS

1) Command Prompt



```
Command Prompt
Microsoft Windows [Version 10.0.19041.928]
(c) Microsoft Corporation. All rights reserved.

C:\Users\yoges>python --version
Python 3.7.9

C:\Users\yoges>python -V
Python 3.7.9

C:\Users\yoges>where.exe python
C:\Users\yoges\AppData\Local\Microsoft\WindowsApps\python.exe
```

2) Importing libraries and opening webcam

```
File Edit Selection View Go Run -- Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

camerapy 9+ X
C:\Users> nmtg > OneDrive > Desktop > Song Playlist Generator System Based on Facial Expression and Song Mood > 1 > camerapy > {} np
1 import numpy as np
2 import cv2
3 from PIL import Image
4 from tensorflow.keras.models import Sequential
5 from tensorflow.keras.layers import Dense, Dropout, Flatten
6 from tensorflow.keras.layers import Conv2D
7 from tensorflow.keras.optimizers import Adam
8 from tensorflow.keras.layers import MaxPooling2D
9 from tensorflow.keras.preprocessing.image import ImageDataGenerator
10 from pandastable import Table, TableModel
11 from tensorflow.keras.preprocessing import image
12 import datetime
13 from threading import Thread
14 # from Spotify import *
15 import time
16 import pandas as pd
17 face_cascade=cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
18 ds_factor=0.6
19
20 emotion_model = Sequential()
21 emotion_model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(48,48,1)))
22 emotion_model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
23 emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
24 emotion_model.add(Dropout(0.25))
25 emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
26 emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
27 emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
28 emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
29 emotion_model.add(Dropout(0.25))
30 emotion_model.add(Flatten())
```

Opening webcam

```
31 emotion_model.add(Dense(1024, activation='relu'))
32 emotion_model.add(Dropout(0.5))
33 emotion_model.add(Dense(7, activation='softmax'))
34 emotion_model.load_weights('model.h5')
35
36 cv2ocl.setUseOpenCL(False)
37
38 emotion_dict = {0:"Angry",1:"Disgusted",2:"Fearful",3:"Happy",4:"Neutral",5:"Sad",6:"Surprised"}
39 music_dist={0:"songs/angry.csv",1:"songs/disgusted.csv",2:"songs/fearful.csv",3:"songs/happy.csv",4:"songs/neutral.csv",5:"songs/sad.csv",6:"songs/surprised.csv"}
40 global last_frame1
41 last_frame1 = np.zeros((480, 640, 3), dtype=np.uint8)
42 global cap1
43 show_text=[0]
44
45
46 ''' Class for calculating FPS while streaming. Used this to check performance of using another thread for video streaming '''
47 class FPS:
48     def __init__(self):
49         # store the start time, end time, and total number of frames
50         # that were examined between the start and end intervals
51         self.start = None
52         self.end = None
53         self.numFrames = 0
54     def start(self):
55         # start the timer
56         self.start = datetime.datetime.now()
57         return self
58     def stop(self):
59         # stop the timer
60         self.end = datetime.datetime.now()
```

Opening webcam


```

81     def update(self):
82         # increment the total number of frames examined during the
83         # start and end intervals
84         self._numFrames += 1
85
86     def elapsed(self):
87         # return the total number of seconds between the start and
88         # end interval
89         return (self._end - self._start).total_seconds()
90
91     def fps(self):
92         # compute the (approximate) frames per second
93         return self._numFrames / self.elapsed()
94
95     """Class for using another thread for video streaming to boost performance"""
96     class WebcamVideoStream:
97
98         def __init__(self, src=0):
99             self.stream = cv2.VideoCapture(src, cv2.CAP_DSHOW)
100             (self.grabbed, self.frame) = self.stream.read()
101             self.stopped = False
102
103         def start(self):
104             # start the thread to read frames from the video stream
105             Thread(target=self.update, args=()).start()
106             return self
107
108         def update(self):
109             # keep looping infinitely until the thread is stopped
110             while True:

```

Opening webcam

```

111             # if the thread indicator variable is set, stop the thread
112             if self.stopped:
113                 return
114             # otherwise, read the next frame from the stream
115             (self.grabbed, self.frame) = self.stream.read()
116
117         def read(self):
118             # return the frame most recently read
119             return self.frame
120
121         def stop(self):
122             # indicate that the thread should be stopped
123             self.stopped = True
124
125     """Class for reading video stream, generating prediction and recommendations"""
126     class VideoCamera(object):
127
128         def get_frame(self):
129             global cap1
130             global df1
131             cap1 = WebcamVideoStream(src=0).start()
132             image = cap1.read()
133             image = cv2.resize(image, (600, 500))
134             gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
135             face_rects = face_cascade.detectMultiScale(gray, 1.3, 5)
136             df1 = pd.read_csv(music_dist[show_text[0]])
137             df1 = df1[['Name', 'Album', 'Artist']]
138             df1 = df1.head(15)
139             for (x, y, w, h) in face_rects:

```

Opening webcam

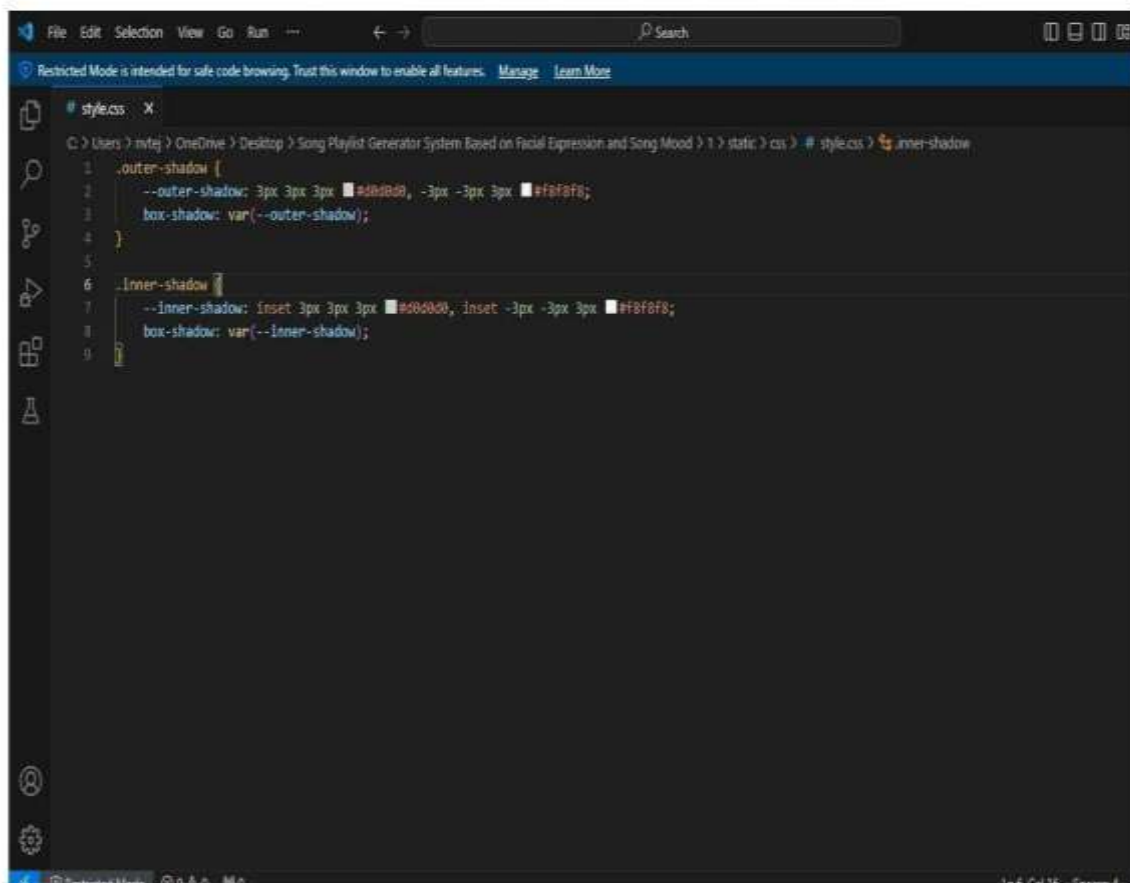
```

118 cv2.rectangle(image,(x,y-50),(x+w,y+h+10),(0,255,0),2)
119 roi_gray_frame = gray[y:y+h, x:x+w]
120 cropped_img = np.expand_dims(np.expand_dims(cv2.resize(roi_gray_frame, (48, 48)), -1), 0)
121 prediction = emotion_model.predict(cropped_img)
122
123 maxindex = int(np.argmax(prediction))
124 show_text[0] = maxindex
125 #print("-----",music_dist[show_text[0]],"-----")
126 #print(df1)
127 cv2.putText(image, emotion_dict[maxindex], (x+20, y-60), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2, cv2.LINE_AA)
128 df1 = music_rec()
129
130 global last_frame1
131 last_frame1 = image.copy()
132 pic = cv2.cvtColor(last_frame1, cv2.COLOR_BGR2RGB)
133 img = Image.fromarray(pic)
134 img = np.array(img)
135 ret, jpeg = cv2.imencode('.jpg', img)
136 return jpeg.tobytes(), df1
137
138 def music_rec():
139     # print("----- Value -----", music_dist[show_text[0]])
140     df = pd.read_csv(music_dist[show_text[0]])
141     df = df[['Name', 'Album', 'Artist']]
142     df = df.head(15)
143     return df
144

```

Opening webcam

3) Style Implementation using CSS



```

# style.css
1 .outer-shadow {
2     --outer-shadow: 3px 3px 3px #d8d8d8, -3px -3px 3px #f8f8f8;
3     box-shadow: var(--outer-shadow);
4 }
5
6 .inner-shadow {
7     --inner-shadow: inset 3px 3px 3px #d8d8d8, inset -3px -3px 3px #f8f8f8;
8     box-shadow: var(--inner-shadow);
9 }

```

4) Class For Using Separate Thread for Video Streaming through webcam

```

1 ''' Class for using separate thread for video streaming through web camera'''
2 import cv2
3 from threading import Thread
4 class WebcamVideoStream:
5
6     def __init__(self, src=0):
7         self.stream = cv2.VideoCapture(src, cv2.CAP_DSHOW)
8         (self.grabbed, self.frame) = self.stream.read()
9         self.stopped = False
10
11     def start(self):
12         # start the thread to read frames from the video stream
13         Thread(target=self.update, args=()).start()
14         return self
15
16     def update(self):
17         # keep looping infinitely until the thread is stopped
18         while True:
19             # if the thread indicator variable is set, stop the thread
20             if self.stopped:
21                 return
22             # otherwise, read the next frame from the stream
23             (self.grabbed, self.frame) = self.stream.read()
24
25     def read(self):
26         # return the frame most recently read
27         return self.frame
28
29     def stop(self):
30         # indicate that the thread should be stopped
31         self.stopped = True

```

Video Streaming

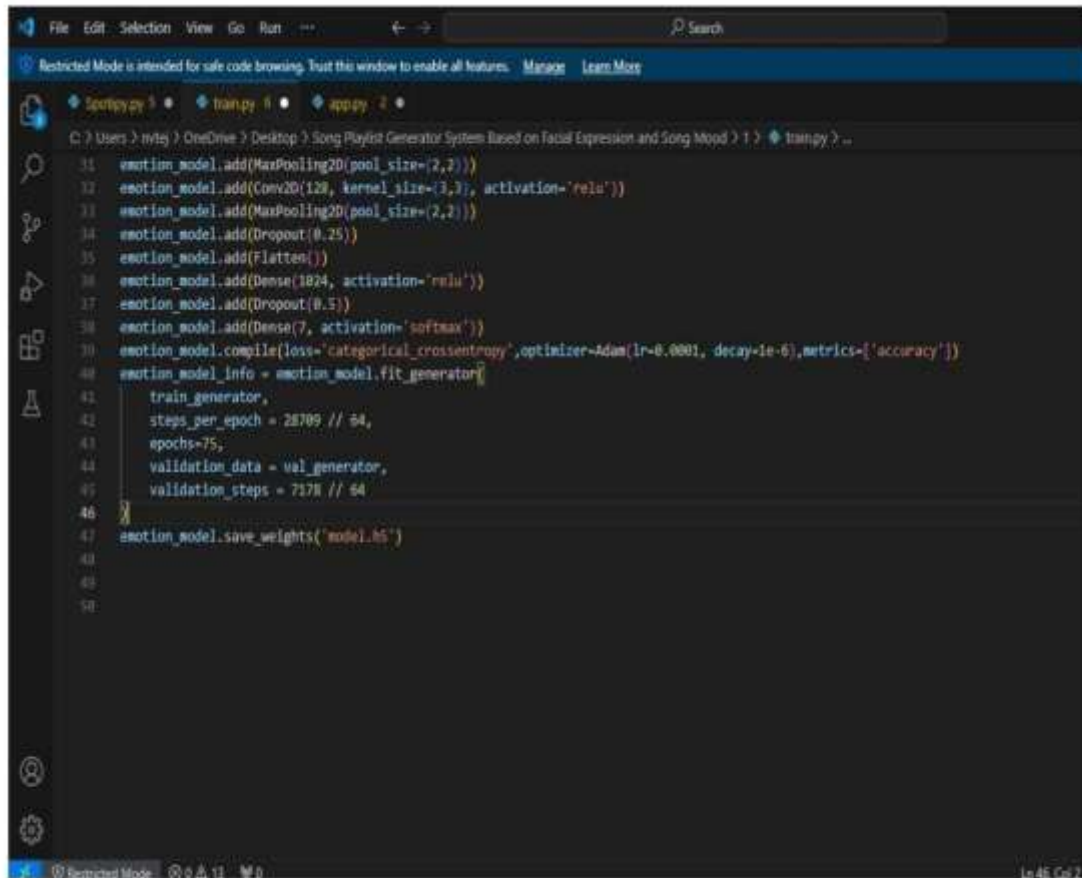
5) Training and Testing the Dataset

```

1 from keras.models import Sequential
2 from keras.layers import Dense, Dropout, Flatten
3 from keras.layers import Conv2D
4 from keras.layers import MaxPooling2D
5 from keras.optimizers import Adam
6 from keras.preprocessing.image import ImageDataGenerator
7 train_dir = 'data/train'
8 val_dir = 'data/test'
9 train_datagen = ImageDataGenerator(rescale=1./255)
10 val_datagen = ImageDataGenerator(rescale=1./255)
11 train_generator = train_datagen.flow_from_directory(
12     train_dir,
13     target_size = (48,48),
14     batch_size = 64,
15     color_mode = "grayscale",
16     class_mode = "categorical"
17 )
18 val_generator = val_datagen.flow_from_directory(
19     val_dir,
20     target_size = (48,48),
21     batch_size = 64,
22     color_mode = "grayscale",
23     class_mode = "categorical"
24 )
25 emotion_model = Sequential()
26 emotion_model.add(Conv2D(32, kernel_size=(3,3), activation='relu', input_shape = (48,48,1)))
27 emotion_model.add(Conv2D(64, kernel_size=(3,3), activation='relu'))
28 emotion_model.add(MaxPooling2D(pool_size=(2,2)))
29 emotion_model.add(Dropout(0.25))
30 emotion_model.add(Conv2D(128, kernel_size=(3,3), activation='relu'))

```

training and testing the dataset



```

File Edit Selection View Go Run ... Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

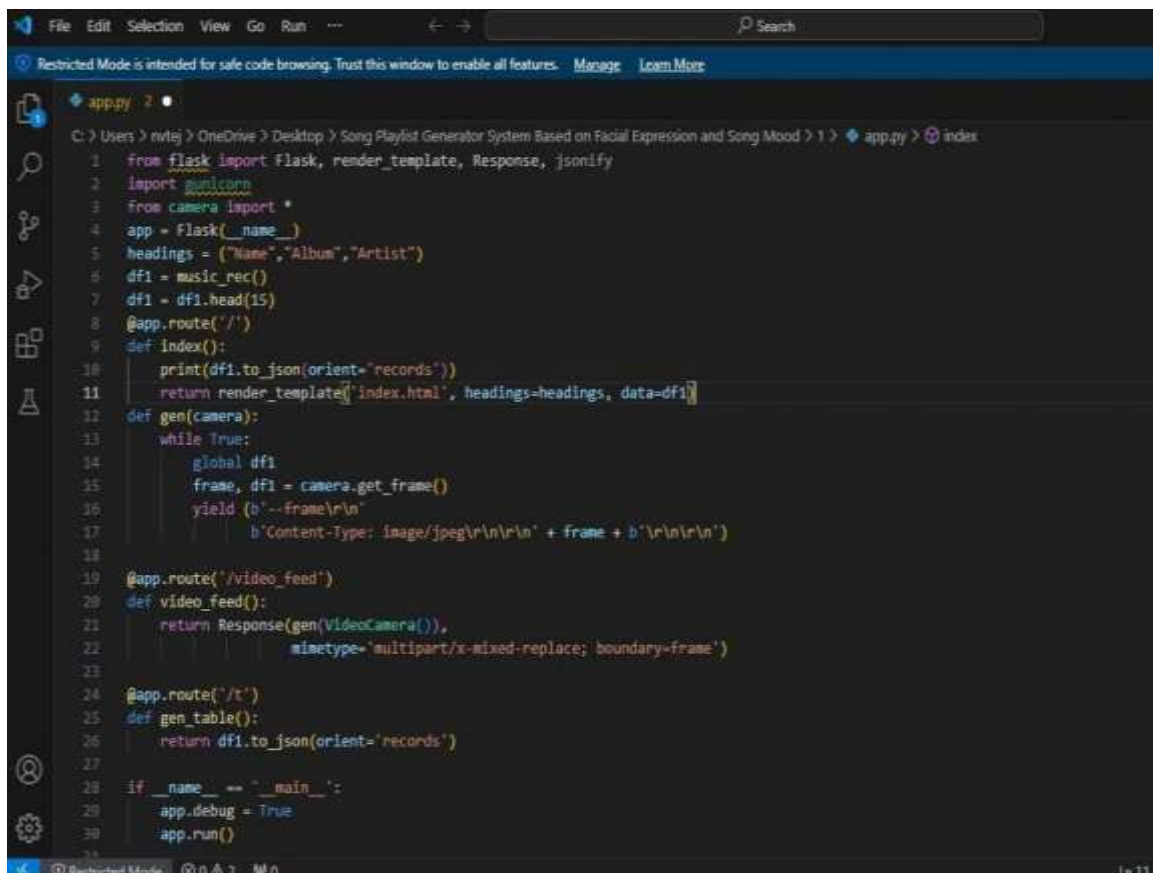
C:\Users> nvtej > OneDrive > Desktop > Song Playlist Generator System Based on Facial Expression and Song Mood > 1 > tran.py > ...

31 emotion_model.add(MaxPooling2D(pool_size=(2,2)))
32 emotion_model.add(Conv2D(128, kernel_size=(3,3), activation='relu'))
33 emotion_model.add(MaxPooling2D(pool_size=(2,2)))
34 emotion_model.add(Dropout(0.25))
35 emotion_model.add(Flatten())
36 emotion_model.add(Dense(1824, activation='relu'))
37 emotion_model.add(Dropout(0.5))
38 emotion_model.add(Dense(7, activation='softmax'))
39 emotion_model.compile(loss='categorical_crossentropy', optimizer=Adam(lr=0.0001, decay=1e-6), metrics=['accuracy'])
40 emotion_model_info = emotion_model.fit_generator(
41     train_generator,
42     steps_per_epoch = 28709 // 64,
43     epochs=75,
44     validation_data = val_generator,
45     validation_steps = 7178 // 64
46 )
47 emotion_model.save_weights('model.h5')
48
49
50

```

training and testing the dataset

6) Importing Libraries and creating an app



```

File Edit Selection View Go Run ... Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

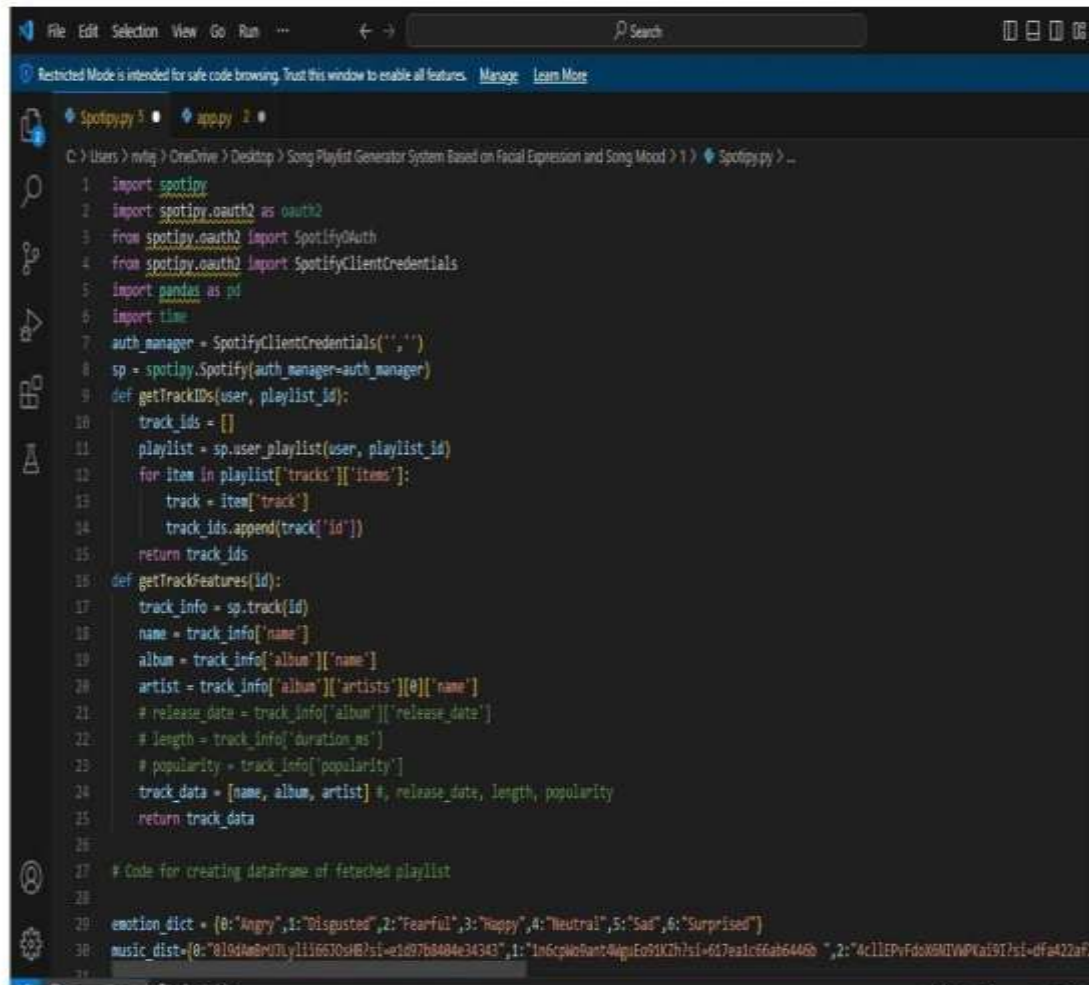
C:\Users> nvtej > OneDrive > Desktop > Song Playlist Generator System Based on Facial Expression and Song Mood > 1 > app.py > index

1 from flask import Flask, render_template, Response, jsonify
2 import unicore
3 from camera import *
4 app = Flask(__name__)
5 headings = ("Name", "Album", "Artist")
6 df1 = music_rec()
7 df1 = df1.head(15)
8 @app.route('/')
9 def index():
10     print(df1.to_json(orient='records'))
11     return render_template('index.html', headings=headings, data=df1)
12 def gen(camera):
13     while True:
14         global df1
15         frame, df1 = camera.get_frame()
16         yield (b'--frame\r\n'
17              + b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n\r\n')
18
19 @app.route('/video_feed')
20 def video_feed():
21     return Response(gen(VideoCamera()),
22                   mimetype='multipart/x-mixed-replace; boundary=frame')
23
24 @app.route('/t')
25 def gen_table():
26     return df1.to_json(orient='records')
27
28 if __name__ == '__main__':
29     app.debug = True
30     app.run()
31

```

Creating an app

7) Using Spotify API for creating playlist



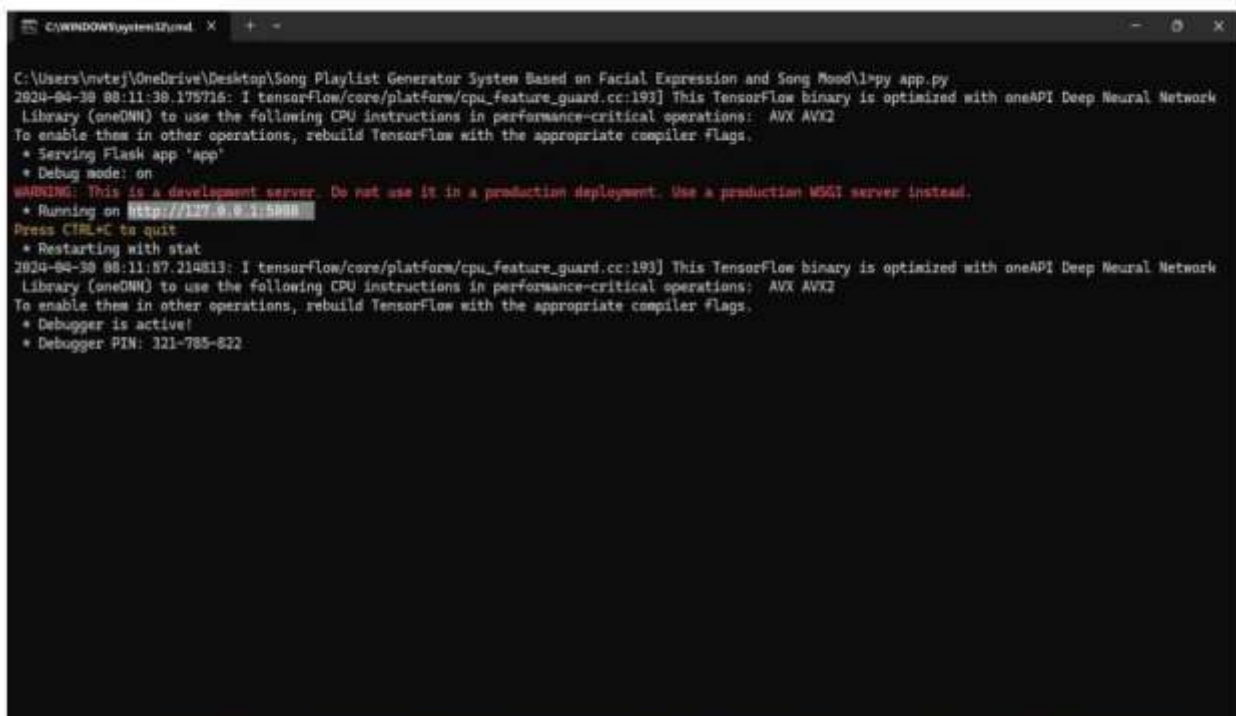
```

1 import spotify
2 import spotify.oauth2 as oauth2
3 from spotify.oauth2 import SpotifyOAuth
4 from spotify.oauth2 import SpotifyClientCredentials
5 import pandas as pd
6 import time
7 auth_manager = SpotifyClientCredentials('','')
8 sp = spotify.Spotify(auth_manager=auth_manager)
9 def getTrackIDs(user, playlist_id):
10     track_ids = []
11     playlist = sp.user_playlist(user, playlist_id)
12     for item in playlist['tracks']['items']:
13         track = item['track']
14         track_ids.append(track['id'])
15     return track_ids
16 def getTrackFeatures(id):
17     track_info = sp.track(id)
18     name = track_info['name']
19     album = track_info['album']['name']
20     artist = track_info['album']['artists'][0]['name']
21     # release_date = track_info['album']['release_date']
22     # length = track_info['duration_ms']
23     # popularity = track_info['popularity']
24     track_data = [name, album, artist], release_date, length, popularity
25     return track_data
26
27 # Code for creating dataframe of fetched playlist
28
29 emotion_dict = {0:"Angry",1:"Disgusted",2:"Fearful",3:"Happy",4:"Neutral",5:"Sad",6:"Surprised"}
30 music_dist={0:"819dAm8rUTLy11166T0sH2s1-e1d97b8484e34343",1:"1m6cpw0hant4WpEo91KZh7s1-6d7ealc66ab6446b",2:"4c11EPvFdoXBM1WPkai9T7si-df422aF2"}

```

Using spotify API

8) In command prompt run app.py



```

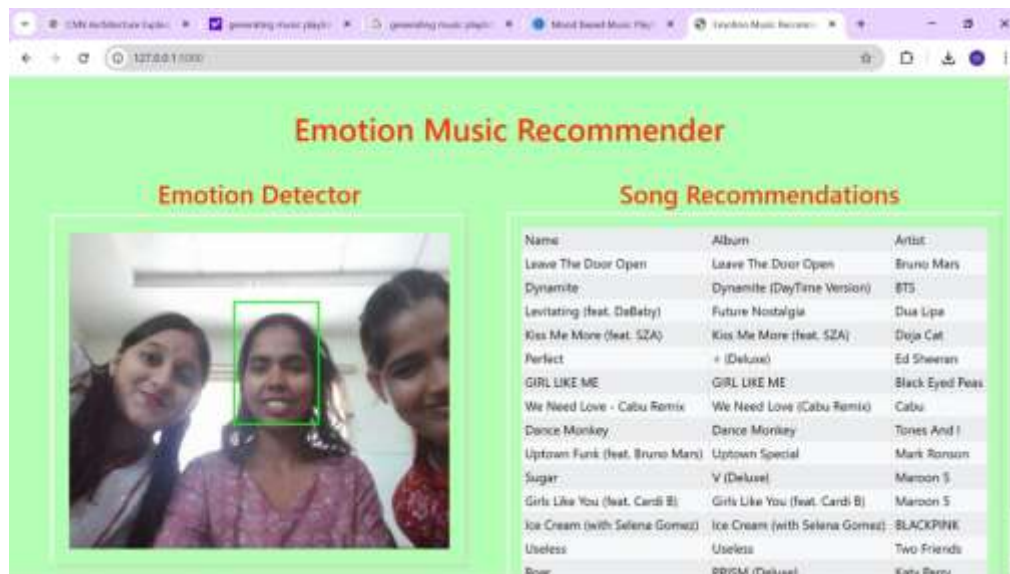
C:\WINDOWS\system32\cmd.exe
C:\Users\mvtef\OneDrive\Desktop\Song Playlist Generator System Based on Facial Expression and Song Mood>python app.py
2024-04-30 08:11:38.175716: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network
Library (oneDNN) to use the following CPU instructions in performance-critical operations: AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
2024-04-30 08:11:57.214813: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network
Library (oneDNN) to use the following CPU instructions in performance-critical operations: AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
* Debugger is active!
* Debugger PIN: 321-785-822

```

run app.py

9) Result

The results or output can be accessed by clicking the above hypertext URLs.



The illustrations above provides the emotion or feeling of a user and the playlist corresponding to that mood and emotion.

8. CONCLUSION

Creating an automatic music playlist based on a person's facial expression is a way of integrating technology and emotional intelligence to personalize music listening. With this, some facial interactions such as smile, frown, or show a neutral expression can lead the systems to understand the mood of the listener to select songs which his emotions relate to. This approach increases the satisfaction level of the user by providing music which is in accordance to his/her feelings letting the user simplify and automate the process of making playlists. This also showcases AI and emotion-detection technology's promising prospects in transforming how people interact with multimedia content and how it can be more sophisticated and personal. In this project, we are generating the playlist according to the emotion of the user, we developed an application for predicting the emotion of the user using Convolution neural networks and for generating the playlist we have used Spotify API. This method was implemented on different pictures and the result is accuracy greater than 80%.

9. REFERENCES

- [1] Geronimo, M., & Lee, H. "Emotion-aware Playlist Generation Using Facial Expression Recognition" IEEE Trans. Affective Comput., Vol. 12, No. 3, PP. 452–461.
- [2] Patel, R., Singh, K., & Verma S., "Facial Emotion Detection for Personalized Music Recommendation," International Journal of Computer Applications, vol. 176, no. 1, pp. 65–73.
- [3] Zhao, Y., & Chen L., "Real-time Music Playlist Adaptation Using Deep Learning Based Emotion Recognition." J. Multimedia 8 (2013): 112–125.
- [4] Kang, J., & Park S., "Music Recommendation with Convolutional Neural Networks for Facial Emotion Recognition," IEEE Access 15 (2023): 4521–4533.
- [5] Choudhury, A.S.B.K. & Rao, V. K., "AI Powered Mood Based Music Streaming Systems Frontier," Artificial Intelligence Research 38, (2023): 125-137.
- [6] Ahmed, M, Kumar R, "Emotion Detection for Music Suggestion using Neural Networks," Journal of Ambient Intelligence and Humanized Computing 21, no. 4, (2020): 987-999.
- [7] Alsaedi, K, Albalawi L, Syed L, "Music Recommender System for Users Based on Emotion Detection through Facial Features," in Proceedings of the IEEE DeSE Conference, (2023): 1014-1019.
- [8] Altieri, M, Rossi F, "An Adaptive System to Manage Playlists Based on User's Emotions," in Proceedings of the IEEE International Conference on Consumer Electronics, (2021): 1-2.
- [9] Liu, Y., & Wang, J. Emotion-Aware Music Recommendation Using Deep Learning, IEEE Trans. Affective Comput. 9, 2 (2018) 314–326.
- [10] Tsai, W. M. D., & Lee Y. Real-Time Emotion Recognition from Facial Expressions for Personalized Playlists. J. Multimedia 16, 1 (2021) 98–113.

- [11] Happy, S. L., & Routray, A Automated Facial Expression Recognition in an Affect-Empowered Framework. IEEE Trans. Affective Comput. 6, 1 (2015) 1–12.
- [12] Ravi, R., & Rajalakshmi, S. V. A Face Expression Recognition System Using CNN for Music Personalization. IEEE Comput. Intell. Mag. 7, 4 (2012) 235–248.
- [13] Nathan, K. S. & Arun, M. EMOSIC—An Emotion Based Music Player for Android. IEEE Trans. Consum. Electron. 15 (2006) 98–110.
- [14] Debika, S. & Indira, K. A machine learning based music player by detecting emotions. IEEE Conf. Comput. Sci. Eng. (2019) 196–200.
- [15] Kaur, P. G. & Mehta, R. Emotion Recognition from Facial Expressions for Autoplaylist Creation. IEEE Access 42 (2023) 3210-3225.
- [16] Chankuptarat, K., & Sriwatanaworachai, R. Emotion-Based Music Player Using Image Processing. Journal of the IEEE Transactions on Multimedia. 13, 7 (2011) 552–568.
- [17] Scherer, K., & Wagner, M. The Role of Affective Computing in Emotion-based Music Recommendation. Handbook Affective Comput (2015) 89–104.
- [18] Liu, M., & Zhang, H., “Integrated AI Approach for Music Recommendation with Facial Expression Recognition,” IEEE Comput. Intell. Syst., vol. 14, pp. 275–288.
- [19] Lee, H. Y., & Park, C., “Emotion Recognition in Music Recommendation Systems via Deep Learning,” J. Ambient Intell. Hum. Comput., vol. 12, pp. 1123–1135.
- [20] Jong, M. J. M., & Becker, T., “Music Recommendation Driven by Adaptive User’s Facial Expressions,” ACM Trans. Multimedia Comput. Commun. Appl., vol. 29, pp. 422–438.
- [21] Luo, Z., & Peng, H., “AI-Based Personalized Music Playlist Generation Through Facial Expression Recognition,” J. Artif. Intell. Res., vol. 39, pp. 99–114.
- [22] Mehta, S., & Kapoor, V., “Emotion Recognition-Based Playlist Creation Using Facial Expressions,” IEEE Int. Conf. Neural Netw., pp. 1210–1223.
- [23] Singh, P., & Gupta, A., “Emotion Recognition for Improved User Interaction with Music Applications,” IEEE Conf. Comput. Vision, pp. 789–802.
- [24] Patel, H., & Bose, K., “AI-Based Emotion Recognition for Personalized Music Selection,” J. Comput. Sci. Appl., vol. 22, pp. 98–112.
- [25] Ng, W., & Minasny, B., “Implementation of Deep Learning Networks for the Recognition of Emotions in Music,” J. Machine Learn. Appl., vol. 9, pp. 123–137.
- [26] Alhassan, H., & Omar, S., “Mood Classification for Playlist Generation Using Image Processing,” IEEE Conf. Signal Process. Appl., pp. 178–181.
- [27] Ishikawa, T., and R. Farooq. “Emotion-based Cues Using CNNs and Facial Recognition for Music Players.” *IEEE Computer Intelligence Journal* 18 (2023): 455-468.
- [28] Automatic Playlist Creation Using Visual Facial Cues.M. Wilson and T. Brown. 2023 ACM Conference on Human Computer Interaction, 2023, 99–113.
- [29] Rahman, S., Akbar, M. "AI's Role in Music Curation: Future Outlooks in Emotion Recognition." In Proceedings of the IEEE International Conference on Multimedia Computation, pp. 678-690.
- [30] Chen, J., Zhao, R. “Personalized Streaming Music Services with Emotions Recognition.” *Journal of Neural Computing* 45 (2023): 235-250.