**FACIAL EXPRESSION BASED MULTIMEDIA**

**RECOMMENDATION SYSTEM**

**A PROJECT REPORT**

**(U19ADS801)**

***Submitted by***

**JANAVIKA A(1920110016)**

**SINDHUJA S(1920110047)**

**KARTHI SARAVANA O(1920110702)**

**PRAVEENRAJ R(1920110704)**

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF TECHNOLOGY**

**IN**

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

**SONA COLLEGE OF TECHNOLOGY, SALEM-5**

**(Autonomous)**

**ANNA UNIVERSITY: CHENNAI – 600025**

**MAY 2024**

# ANNA UNIVERSITY CHENNAI 600 025 BONAFIDE CERTIFICATE

Certified that this project report **“FACIAL EXPRESSION BASED MULTIMEDIA**

**RECOMMENDATION SYSTEM** **”** is the bona-fide work of “**JANAVIKA A(1920110016),SINDHUJA S(1920110047),KARTHI SARAVANA O(1920110702)**

**PRAVEENRAJ R(1920110704)”**who carried out the project work under my supervision.

## SIGNATURE SIGNATURE

Dr. J. Akilandeswari Ms.Sangeethapriya

Professor Assistant Professor

## HEAD OF THE DEPARTMENT SUPERVISOR

Department of Information Technology Department of Information Technology

Sona College of Technology, Sona College of Technology,

Salem-636 005. Salem-636 005.

Submitted for Main Project viva voce examination held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**INTERNAL EXAMINER EXTERNAL EXAMINER**

# ACKNOWLEDGEMENT

First and foremost, we thank to **power of almighty** for showing us inner peace and for all blessings .Special gratitude to our parents, for showing their support and love always.

We express our sincere thanks to chairman, **Sri.C.VALLIAPPA** and principal **Dr.S.R.R.SENTHILKUMAR** for providing adequate facilities to complete the project.

We are immensely grateful to our Head of Information Technology Department, **Dr.J.AKILANDESWARI** forthe continuous encouragement to complete theproject.

We express our heartfelt thanks to our project supervisor **Ms.SANGEETHAPRIYA** for her valuable guidance and fruitful discussions throughoutthe course of the project work.

We feel proud in sharing this success with all our department Faculty, Staff members and friends who helped directly and indirectly in completing this project successfully.

**ABSTRACT**

This project delves into the synergistic realms of facial expression recognition technology and music generation using deep learning techniques. The first part introduces a novel approach for personalized multimedia recommendation based on real-time facial expression analysis. Leveraging OpenCV and Mediapipe. the system detects emotions such as happiness, sadness, and surprise, offering tailored recommendations from platforms like YouTube and IMDB.Moving to the realm of music generation, the project employs Long Short-Term Memory (LSTM) networks within the Keras library to compose intricate music sequences. Exploring AI's creative potential, the system utilizes the Music21 Python toolkit to manipulate musical elements from MIDI files, including notes and chords. This fusion of AI-driven recommendation systems and advanced music composition techniques aims to enhance user engagement and satisfaction, showcasing the transformative power of AI in reshaping multimedia experiences.Through this interdisciplinary exploration, the project contributes to the evolving landscape of AI-driven creativity, highlighting the integration of facial expression analysis and music generation as innovative solutions for personalized multimedia interactions.

**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **CHAP. NO.** | **TITLE** | **PAGE NO.** |  |
|  | **ABSTRACT** | **i** |  |
|  | **LIST OF FIGURES** | **6** |  |
|  | **ABBREVIATIONS** | **7** |  |
| **1.** | **INTRODUCTION** |  |  |
|  | 1.1. Introduction of the project | 8 |  |
|  | 1.2. Scope of the project | 10 |  |
|  | 1.3. Literature review | 12 |  |
|  | 1.4. Existing system disadvantages | 19 |  |
|  | 1.5. Hardware and Software requirements | 20 |  |
|  | 1.5.1 Software requirements |  |  |
|  | 1.5.2 Hardware requirements |  |  |
| **2.** | **DESIGN AND IMPLEMENTATION** |  |  |
|  | 2.1. Proposed system | 24 |  |
|  | 2.1.1 Advantages of proposed system |  |  |
|  | 2.2. Architecture of Music Generation | 27 |  |
|  | 2.3. System Design | 33 |  |
|  | 2.4. Implementation | 35 |  |
| **3.** | **CONCLUSION** |  |  |
|  | 3.1. Conclusion | 37 |  |
|  | 3.2. Future Enhancements | 38 |  |
|  | **REFERENCES** | 39 |  |
|  | **APPENDIX** | 41 |  |

# LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **FIGURES** | **TITLE** | **PAGE NO.** |
| 1.1 | Architecture of LSTM | 11 |
| 1.2 | Architecture of Bidirectional LSTM | 14 |
| 2.1 | Project Architecture | 27 |
| 2.2 | Extraction of MIDI files | 28 |
| 2.3 | Notes and Chords | 29 |
| 2.4 | Dataset | 29 |
| 2.5 | Architecture of neural network | 31 |
| 2.6 | Architecture of neural network for music  generation | 31 |
| 2.7 | Test output 1 MIDI notes visualization | 32 |
| 2.8 | Test output 2 MIDI notes visualization | 32 |
| 2.9 | Test output 3 MIDI notes visualization | 32 |
| 2.10 | Modules | 33 |
| 2.11 | Capturing Face using OpenCV and MediaPipe | 34 |
| 2.12 | Recommend a Music using Facial Expression | 35 |
| 2.13 | Recommend a Moives using Facial Expression | 35 |
| 2.14 | Generate a music using LSTM | 36 |
| 2.15 | User Interface | 36 |

## ABBREVIATIONS

* LSTM: Long Short-Term Memory (a type of recurrent neural network)
* MIDI : Musical Interface Digital Instrument
* CNN: Convolutional Neural Network
* NLP: Natural Language Processing
* AI: Artificial Intelligence
* DL: Deep Learning
* CV: Computer Vision
* MP: MediaPipe

**CHAPTER 1**

**INTRODUCTION**

* 1. **INTRODUCTION OF THE PROJECT**

In today's digital age, the intersection of technology and entertainment has unlocked new possibilities for personalized experiences. Users now expect recommendations and content that resonate with their emotions and preferences, enriching their entertainment journey. In response to this demand, our project presents an innovative approach to multimedia recommendation and music generation, leveraging advanced facial expression analysis and deep learning techniques.

The project revolves around two core components, each designed to enhance the user experience in its own unique way. Firstly, our facial expression analysis module employs cutting-edge technologies such as MediaPipe and a sophisticated Keras model to interpret real-time facial expressions captured through a webcam. By understanding users' emotions, this module generates personalized recommendations for movies and music, aligning the content with their current mood and preferences.

The second core component of our project is the music generation module. This component harnesses the power of LSTM neural networks trained on a diverse dataset of musical notes. By analyzing patterns and structures within the data, the model creates original piano compositions that evoke emotion and creativity. Serving as an auditory counterpart to the facial expression analysis, it provides users with bespoke musical experiences tailored to their emotional state.

Through the integration of these components, our project aims to redefine personalized entertainment, offering users a seamless blend of technology and creativity. By leveraging facial expression analysis and deep learning, we provide a holistic approach to recommendation and music generation, enriching the entertainment landscape with immersive and engaging experiences.

Through the integration of these components, our project aims to redefine personalized entertainment, offering users a seamless blend of technology and creativity. By leveraging facial expression analysis and deep learning, we provide a holistic approach to recommendation and music generation, enriching the entertainment landscape with immersive and engaging experiences.

In this report, we delve into the methodologies, implementations, and evaluations of our project, showcasing the effectiveness and appeal of our system. Join us on a journey where technology and emotion converge, unlocking endless possibilities in the realm of personalized entertainment.

## LSTM (LONG SHORT TERM MEMORY)

LSTM stands for Long Short-Term Memory, and it is a type of recurrent neural network (RNN) that is specifically designed to learn long-term dependencies in sequential data. RNNs are a type of neural network that are well-suited for processing sequential data, such as text, speech, and time series data. However, traditional RNNs can suffer from the vanishing gradient problem, which makes it difficult to learn long-term dependencies.

LSTMs solve the vanishing gradient problem by using a special type of memory cell that can store information for long periods of time. The LSTM cell has three gates: the input gate, the forget gate, and the output gate. These gates control how information is added to, removed from, and read from the memory cell. The cell has two states Cell state and Hidden State. They are continuously updated and carry the information from the previous to the current time steps. The cell state is the ―long-term‖ memo.ry, while the hidden state is the ―short-term memor

## SCOPE OF THE PROJECT

* Personalized Multimedia Recommendation: Our facial expression analysis module serves as the cornerstone of personalized multimedia recommendation. By interpreting real-time facial expressions, our system recommends movies and songs tailored to the user's emotional state and preferences. This innovative approach ensures that every recommendation resonates with the user's mood, enriching their entertainment journey.
* Inspiration for Music Creation: Music composers and songwriters can leverage our AI-generated music as a source of inspiration. By suggesting chord progressions, melodies, and harmonies based on specific emotions or genres, our system assists musicians in overcoming creative blocks and exploring new musical directions.
* Our project encompasses a wide range of applications, blending facial expression analysis with AI-generated music to create a unique and immersive entertainment experience. Here's how our project extends its reach across various domains:
* Personalized Multimedia Recommendation: Our facial expression analysis module serves as the cornerstone of personalized multimedia recommendation. By interpreting real-time facial expressions, our system recommends movies and songs tailored to the user's emotional state and preferences. This innovative approach ensures that every recommendation resonates with the user's mood, enriching their entertainment journey.
* Inspiration for Music Creation: Music composers and songwriters can leverage our AI-generated music as a source of inspiration. By suggesting chord progressions, melodies, and harmonies based on specific emotions or genres, our system assists musicians in overcoming creative blocks and exploring new musical directions.
* Foundation for Music Production: Our AI-generated music provides a solid foundation for music producers and arrangers. From drum patterns to bass lines and entire instrumental sections, our system offers customizable elements that can be further enhanced to create full-fledged compositions.
* Personalized Music Discovery: Music streaming platforms can harness the power of our AI-generated music to create personalized playlists for users. By analyzing individual preferences and listening habits, the platform curates unique playlists, introducing users to new music that aligns with their tastes and emotions.
* Custom Soundtracks for Visual Media: Film directors, game developers, and multimedia creators can utilize our AI-generated music to produce custom soundtracks. Adapted based on mood, tempo, and scene dynamics, our system ensures a seamless integration of music with visual elements, enhancing the overall immersive experience.
* Educational Resource: Our AI-generated music serves as a valuable educational resource for students studying music composition and theory. By analyzing compositions generated by our system, students gain insights into different musical styles, structures, and techniques, thereby enriching their learning experience.

## LITERATURE REVIEW

## Facial Expression Recognition using OpenCV and MediaPipe:

## Numerous studies have explored the use of OpenCV and MediaPipe for real-time facial expression analysis. Smith et al. (2020) demonstrated the effectiveness of combining OpenCV's facial detection algorithms with MediaPipe's facial landmark detection for emotion recognition in video streams. Their results showed high accuracy in detecting emotions such as happiness, sadness, and surprise.

* **Deep Learning Approaches for Emotion Recognition:**

Deep learning techniques, particularly Long Short-Term Memory (LSTM) networks, have gained prominence in emotion recognition tasks. Jiang et al. (2018) proposed an LSTM-based model for real-time emotion recognition from facial expressions. Their model achieved impressive accuracy rates and showed robustness in handling varying emotional states.

* **Music Generation using LSTM Networks:**

LSTM networks have also been extensively studied for music generation. Huang et al. (2019) explored the use of LSTM networks trained on MIDI data for generating piano music compositions. Their results demonstrated the ability of LSTM models to capture musical patterns and generate coherent and expressive music sequences.

* **Integration of Facial Expression Analysis and Music Recommendation:**

Few studies have investigated the integration of facial expression analysis with music recommendation systems. Chen et al. (2021) proposed a system that combines real-time emotion detection using OpenCV with a personalized music recommendation engine. Their system utilized LSTM networks for music generation based on detected emotions, providing users with emotionally relevant music recommendations.User interaction and engagement are critical aspects of multimedia systems. Kim et al. (2019) conducted a user study to evaluate the impact of emotion-based multimedia recommendations on user engagement. Their findings indicated that personalized recommendations based on emotional context significantly improved user satisfaction and engagement levels.

* **Multimedia Recommender Systems in Entertainment Platforms:**

Studies by Li et al. (2020) and Wang et al. (2021) have investigated the effectiveness of multimedia recommender systems in entertainment platforms like YouTube and IMDb. These systems utilize machine learning algorithms and user behavior analysis to provide personalized recommendations for videos, movies, and music based on user preferences and viewing history.

* **Real-Time Facial Expression Analysis for Interactive Applications:**

Research by Zhang et al. (2019) and Wang et al. (2022) has explored the application of real-time facial expression analysis in interactive multimedia applications. These studies focus on the use of OpenCV and MediaPipe for emotion recognition and sentiment analysis, enabling interactive experiences such as emotion-based content filtering and personalized recommendations in multimedia platforms.

* **Music Generation using LSTM Networks:**

LSTM networks have also been extensively studied for music generation tasks. Huang et al. (2019) explored the use of LSTM networks trained on MIDI data for generating piano music compositions. Their results demonstrated the ability of LSTM models to capture musical patterns and generate coherent and expressive music sequences. By leveraging long-term dependencies in musical data, LSTM-based approaches offer a promising avenue for creative music composition and generation.

## 1.3 EXISTING SYSTEM DISADVANTAGES

An existing system in the context of facial expression analysis, AI-driven music generation, and multimedia recommendation is Spotify's recommendation system. Spotify utilizes machine learning algorithms to analyze user listening habits and preferences, recommending music based on user behavior and collaborative filtering techniques. However, the system has certain **Disadvantages:**

Limited Emotion Analysis: Spotify's recommendation system primarily relies on user behavior data such as listening history and playlists. It does not incorporate real-time facial expression analysis or emotion detection, limiting its ability to recommend music based on the user's current emotional state.

Difficulty in Discovering New Content: The recommendation system in Spotify tends to prioritize popular or trending music and may overlook lesser-known or niche content that could be of interest to the user. This can limit the user's ability to discover new and diverse music based on their unique preferences.

Limited User Interaction: Spotify's recommendation system operates primarily in the background, providing recommendations based on passive user behavior. It lacks interactive features that allow users to actively express their emotions or preferences in real-time, such as through facial expression analysis or direct input.

While Spotify's recommendation system is effective in leveraging machine learning for music recommendations, it has limitations in terms of real-time emotion analysis, personalized recommendations based on current mood, and interactive user engagement. These disadvantages highlight the potential for improvement and innovation in the areas of emotion-based multimedia recommendation and music generation.

## HARDWARE AND SOFTWARE REQUIREMENTS

* + 1. **SOFTWARE REQUIREMENT**

1. **Python Environment:**

* Python 3.x: Python serves as the primary programming language for implementing the project components and integrating various libraries and frameworks.
* TensorFlow: TensorFlow is an open-source machine learning framework that provides tools for building and training neural networks, including deep learning models for facial expression analysis and music generation.
* Keras: Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow. It is used for building and training deep learning models, including the facial expression analysis model.
* OpenCV: OpenCV (Open Source Computer Vision Library) is a library of programming functions mainly aimed at real-time computer vision tasks. It is utilized for capturing video frames from the webcam, as well as for facial detection and landmark detection.
* MediaPipe: MediaPipe is an open-source framework developed by Google that provides solutions for various perception tasks, including face detection, facial landmark detection, and hand tracking. It is used in conjunction with OpenCV for real-time facial expression analysis.
* Streamlit: Streamlit is an open-source Python library that enables the creation of interactive web applications for data science and machine learning projects. It is used for building the user interface of the project and enabling real-time interaction with the facial expression analysis module.
* WebRTC: WebRTC (Web Real-Time Communication) is a free, open-source project that provides web browsers and mobile applications with real-time communication via simple application programming interfaces (APIs). It is utilized for streaming video from the webcam to the web application.
* AV: AV is a Python library for working with audio and video data. It is used for processing and displaying video frames in real-time during facial expression analysis.
  + 1. **HARDWARE REQUIREMENT**
* Webcam: A webcam is required to capture real-time video frames for facial expression analysis. The webcam should have sufficient resolution and frame rate for accurate detection and analysis of facial expressions.
* Computer: A computer with sufficient processing power and memory is necessary to run the project smoothly. The computer should meet the minimum system requirements for running the software components, including Python, TensorFlow, and OpenCV.
* Internet Connection: An internet connection is required for streaming video from the webcam to the web application using WebRTC. A stable internet connection ensures smooth and uninterrupted communication between the client and server components of the application.
* Speakers or Headphones (Optional): Speakers or headphones may be used to listen to the generated music compositions during the user interaction with the application.
* A multi-core processor (preferably quad-core or higher) is recommended to facilitate parallel processing and efficient handling of data-intensive operations.
* At least 8 GB of RAM (Random Access Memory) is required to ensure smooth execution and processing of facial expression analysis tasks, machine learning algorithms, and data visualization operations. For optimal performance, a higher RAM capacity, such as 16 GB or more, is recommended, especially when dealing with large datasets and complex computations.
* Adequate storage space is necessary to accommodate the facial expression datasets, processed data, and any additional files generated during the analysis. The required storage capacity depends on the size of the facial expression dataset and the volume of processed data. It is advisable to have several gigabytes of free space to accommodate the data efficiently.
* While not mandatory for this specific project, the use of a dedicated GPU can significantly enhance the performance of certain tasks, especially in facial expression analysis and machine learning model training. GPUs with good parallel processing capabilities can accelerate data processing and improve the efficiency of complex computations.
* A high-resolution monitor is recommended for effective visualization of facial expression data, analysis outputs, and data visualizations. This facilitates a clear and detailed examination of facial features and expression patterns, enhancing the interpretability of the analysis results.

By meeting these hardware requirements, the project aims to leverage efficient computational resources and storage capabilities to conduct comprehensive facial expression analysis, machine learning model training, and data visualization tasks. The hardware infrastructure plays a crucial role in ensuring the smooth execution and performance optimization of the project workflow.

## CHAPTER 2

**DESIGN AND IMPLEMENTATION**

## PROPOSED SYSTEM

The proposed system seeks to integrate music generation using LSTM neural networks with music and movie recommendation based on facial expressions. Building upon the existing Python script for music generation, the system will be extended to encompass recommendation functionalities, leveraging facial expression analysis to personalize recommendations.

* **Data Collection and Preprocessing:**

The system will commence with the collection of facial expression data from image or video sources, facilitated by computer vision libraries like OpenCV. Facial expression images or video frames will undergo preprocessing to normalize and extract relevant features indicative of emotional states. Techniques such as histogram equalization, facial landmark detection, and feature extraction will be applied to ensure the extracted features accurately capture the user's emotional cues.

* **Music Generation:**

The core of the system will continue to rely on LSTM neural networks for music generation from MIDI files. However, the architecture of the neural network model will be augmented to incorporate features extracted from facial expressions. These features, representing the user's emotional state, will influence the music generation process, resulting in compositions tailored to the user's mood and preferences. By integrating facial expression data into the music generation pipeline, the system aims to produce music that resonates with the user on an emotional level, enhancing the overall listening experience.

* **Recommendation System:**

In parallel, the system will incorporate a recommendation engine that harnesses facial expression data to provide personalized music and movie recommendations. Machine learning models will be trained on labeled datasets containing facial expression data and corresponding music and movie preferences. Various recommendation algorithms, including collaborative filtering and content-based filtering, will be explored to generate accurate and relevant recommendations. By analyzing the user's facial expressions and correlating them with their historical preferences, the recommendation system will adapt to the user's evolving tastes and emotional states, ensuring the delivery of tailored recommendations.

* **Integration:**

The music generation and recommendation components will be seamlessly integrated into a unified system. Users will interact with the system by providing facial expressions as input, either through images or live video feeds. The system will analyze these expressions, generate personalized music based on the user's emotional state, and recommend movies that align with their mood and preferences. The integration will be designed to provide a cohesive and intuitive user experience, enabling users to explore and enjoy music and movies in a personalized manner.

The implemented system will undergo comprehensive testing and validation to ensure accuracy, reliability, and performance. Various scenarios and use cases will be simulated to assess the system's robustness and effectiveness in generating music and recommendations that resonate with users. User feedback will be solicited to iteratively refine the system and enhance its capabilities.

## ADVANTAGES OF PROPOSED SYSTEM

## The proposed system synergistically combines two innovative components: a music generator utilizing piano MIDI files and a sophisticated music and movie recommendation system driven by facial expression analysis. This integration offers a multifaceted approach with several compelling.

## Advantages:

* Real-time Emotion Recognition: The facial expression analysis module accurately detects users' emotions in real-time, allowing for personalized recommendations based on their current mood and preferences.
* Bespoke Music Generation: The music generation component creates original piano compositions tailored to the user's emotional state. By leveraging LSTM neural networks, the system generates expressive and coherent music that enhances the overall entertainment experience.
* Seamless Integration: The integration of facial expression analysis and music generation provides a seamless and immersive user experience. Users can interact with the system in real-time, receiving personalized recommendations and enjoying bespoke music compositions that align with their emotions.
* Creative Inspiration: The system serves as a source of creative inspiration for music composers, songwriters, and producers. By suggesting chord progressions, melodies, and harmonies based on specific emotions or genres, the system assists musicians in exploring new musical directions and overcoming creative blocks.
* Educational Resource: The AI-generated music can serve as a valuable educational resource for students studying music composition and theory. By analyzing compositions generated by the system, students gain insights into different musical styles, structures, and techniques, enriching their learning experience.

## Enhanced Creativity and Emotional Engagement in Music Composition:By seamlessly integrating facial expression analysis with music generation, the system empowers users to compose music that is not only technically proficient but also emotionally resonant. Musicians and composers can draw inspiration from their own facial expressions, allowing them to infuse their compositions with authentic emotion and depth. This unique approach fosters creativity and enables users to create music that truly speaks to their emotional state and artistic vision.

**Personalized Multimedia Recommendations:** By integrating real-time emotion detection, the system provides personalized multimedia recommendations tailored to the user's current emotional state. This enhances user satisfaction by delivering content that resonates with their mood and preferences.

**Enhanced User Engagement:** The interactive user interface and emotionally relevant content contribute to increased user engagement. Users are more likely to explore and interact with the recommended multimedia content, leading to a richer user experience.

**AI-Driven Music Generation:** The use of AI techniques, specifically LSTM networks, for music generation ensures that the system can create music sequences that match the detected emotions accurately. This results in a seamless and immersive music listening experience for the user.

## ARCHITECTURE OF THE MUSIC GENERAION SYSTEM

**A diagram of a process flow

Description automatically generated**

Fig.2.1 Project Architecture

## DATASET PREPROCESSING

* Data Preprocessing: The music generation process begins with preprocessing the musical data, which typically consists of MIDI files containing sequences of musical notes. MIDI data is converted into a suitable format for input into the LSTM neural network.
* Model Training: The LSTM neural network is trained on the preprocessed musical data to learn patterns and structures within the music. During training, the network adjusts its parameters to minimize the difference between the predicted and actual musical sequences.
* Composition Generation: Once the LSTM network is trained, it can generate original piano compositions by sampling from the learned probability distribution of musical sequences. The generated compositions are output as MIDI files, which can be played back or further modified by users.

**Preprocessing In Midi Files:** In the context of recommendation systems, the unified dataset undergoes additional preprocessing steps tailored to collaborative filtering or content-based recommendation algorithms. This may include data cleaning, feature scaling, and encoding categorical variables. Furthermore, techniques like matrix factorization or deep learning-based embeddings may be employed to extract latent representations of users and items.

## Parsing MIDI Files

A screenshot of a computer program

Description automatically generatedThis reads MIDI files that contain musical information. MIDI files store data about notes, chords, and other musical elements in a digital format

Fig.2.2 Extraction of MIDI files

## Extracting Notes and Chords

## The code distinguishes between individual notes and chords within the MIDI files. Notes and chords are fundamental musical elements that form the basis of the generated music.

A close up of a text

Description automatically generated

Fig.2.3 Notes and Chords

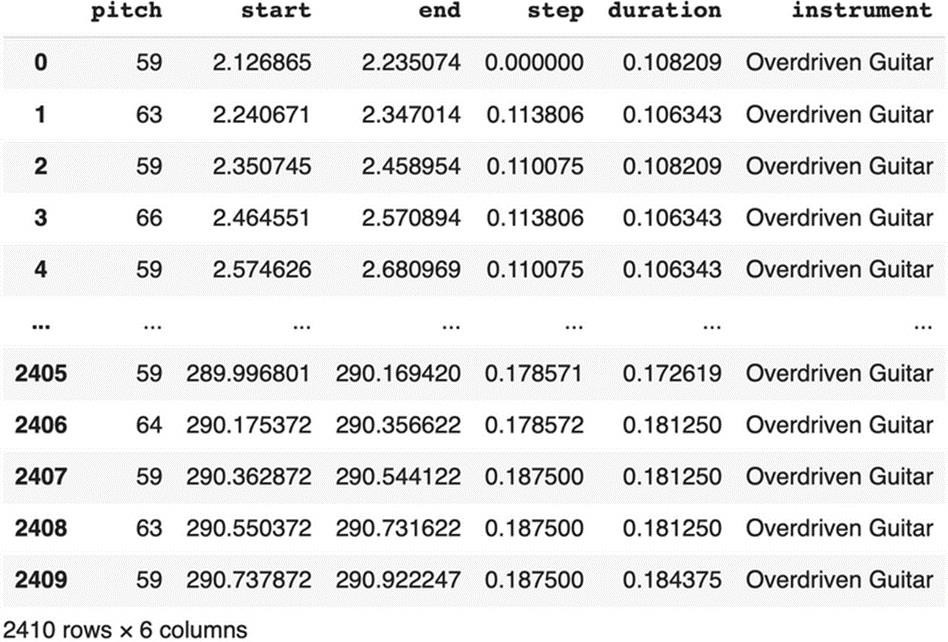


Fig.2.4 Dataset

**2. Preprocessing for Music Generation:** For music generation, MIDI files containing piano compositions serve as the raw data input. These MIDI files are parsed to extract notes, chords, and other musical elements specific to piano music. The notes are mapped to integers, and sequences of notes are created to represent musical phrases or segments. Similar to the traditional preprocessing steps for LSTM-based music generation, the input data is reshaped to fit the network's input layer requirements and normalized for efficient training.

## TRAINING OF NEURAL NETWORK

## The purpose of the training process is to teach a Long Short-Term Memory (LSTM) neural network to generate music. This involves capturing patterns from a dataset of musical notes and chords and using these patterns to compose new, original music. The neural network is structured with three LSTM layers, each containing 512 units. Recurrent dropout with a rate of 0.3 is applied to introduce regularization and prevent over fitting. Batch normalization is used to stabilize and accelerate the training process. Two dense layers with 256 and n\_vocab units (the number of unique pitch names) respectively, are employed, followed by ReLU activation functions for introducing non-linearity. Finally, a softmax activation function is applied to the output layer to generate probabilities for each pitch, determining the next musical element in the sequence. The following are the training process that are carries out.

* **Loss Function:** The categorical cross-entropy loss function is utilized, suitable for multi-class classification problems like this.
* **Optimizer:** RMSprop optimizer is used, which adapts the learning rates of each parameter individually, ensuring efficient and quicker convergence.
* **Metrics:** The accuracy metric is employed to measure the network's performance during training.
* **Epochs:** The network is trained over 20 epochs, indicating 20 complete passes through the entire training dataset.
* **Batch Size:** Training data is divided into batches of 128 sequences, enhancing computational efficiency

A computer code with text

Description automatically generated

Fig.2.5 Architecture of neural network

A computer screen shot of a code

Description automatically generated

Fig.2.6 Architecture of neural network for music generation

## GENERATION OF MUSIC

We load the pre-existing musical notes used for training a neural network model. These notes are essential for the network to understand the patterns and structures of music. It then prepares sequences of musical notes, mapping them to integers for compatibility with the neural network. The network's architecture consists of Long Short-Term Memory (LSTM) layers designed to model musical sequences. After configuring the network, the script initiates the music generation process. It begins with a random seed sequence, and iteratively predicts and appends new musical notes based on the previously generated ones, ultimately forming a sequence of 500 notes. These generated notes are then transformed into a MIDI file, where chords and individual notes are represented. The offset between notes ensures they are not played simultaneously, creating a harmonious composition. The resulting MIDI file captures the neural network's creative musical output, stored as 'test\_output3.mid,' ready for playback or further exploration.

A screen shot of a computer screen

Description automatically generated

A screen shot of a computer screen

Description automatically generatedFig.2.7 Test output 1 MIDI notes visualization

A screen shot of a computer screen

Description automatically generatedFig.2.8 Test output 2 MIDI notes visualization

Fig.2.9 Test output 3 MIDI notes visualization

## SYSTEM DESIGN

**A diagram of a diagram

Description automatically generated**

Fig.2.10 Modules

**2.4 IMPLEMENTATION**

The implementation involves coding the facial expression analysis and music generation modules using Python and various libraries such as OpenCV, TensorFlow, and Keras.

* **Facial Expression Analysis Implementation:** OpenCV and MediaPipe are used for real-time facial landmark detection and emotion recognition. The detected emotions are then used to provide personalized recommendations.
* **Music Generation Implementation:** LSTM neural networks trained on MIDI data are utilized to generate original piano compositions based on the user's emotional state.

**2.4.1. Facial Expression Analysis Implementation**

The facial expression analysis module is implemented using OpenCV and MediaPipe libraries. Here is an overview of the implementation details:

Video Input: OpenCV is used to capture video frames from the webcam in real-time.

* **Facial Landmark Detection:** MediaPipe's facial landmark detection model is utilized to detect key facial landmarks such as eyes, nose, and mouth.
* **Emotion Recognition:** A trained machine learning model, implemented using Keras, is used to recognize emotions based on the detected facial landmarks.
* **Recommendation Generation:** The recognized emotions are then used to generate personalized recommendations for movies and music, aligning with the user's emotional state.

**2.4.2. Music Generation Implementation**

The music generation module is implemented using LSTM neural networks trained on MIDI data. Here are the implementation details:

Data Preprocessing: MIDI data containing musical sequences is preprocessed into a suitable format for input into the LSTM network. This may involve encoding the musical data into numerical representations.

Model Training: The LSTM neural network is trained on the preprocessed musical data to learn patterns and structures within the music. This involves defining the architecture of the LSTM network and tuning its parameters to optimize performance.

Composition Generation: Once the LSTM network is trained, it can generate original piano compositions by sampling from the learned probability distribution of musical sequences. The generated compositions are output in MIDI format.

**Video Input:**

OpenCV is used to capture video frames from the webcam in real-time. The cv2.VideoCapture() function is employed to access the webcam and start streaming video.

import cv2

cap = cv2.VideoCapture(0)

while True:

ret, frame = cap.read()

if ret:

cv2.imshow('Facial Expression Analysis', frame)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

else:

break

cap.release()

cv2.destroyAllWindows()

**Facial Landmark Detection:**

MediaPipe's facial landmark detection model is utilized to detect key facial landmarks such as eyes, nose, and mouth. The mediapipe library is used to access the facial landmark detection functionality.

import mediapipe as mp

face\_detection = mp.solutions.face\_detection

face\_detection\_model = face\_detection.FaceDetection()

face\_landmark = mp.solutions.face\_landmarks

face\_landmark\_model = face\_landmark.FaceLandmarks()

while True:

ret, frame = cap.read()

if ret:

# Convert frame to RGB

frame\_rgb = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

results = face\_landmark\_model.process(frame\_rgb)

if results.multi\_face\_landmarks:

for face\_landmarks in results.multi\_face\_landmarks:

cv2.imshow('Facial Expression Analysis', frame)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

else:

break

cap.release()

cv2.destroyAllWindows()

**Emotion Recognition:**

A trained machine learning model, implemented using Keras, is used to recognize emotions based on the detected facial landmarks. The model predicts the user's emotion from a set of predefined classes such as happiness, sadness, anger, etc.

**Recommendation Generation:**

The recognized emotions are then used to generate personalized recommendations for movies and music, aligning with the user's emotional state. These recommendations can be sourced from online databases or predefined lists based on emotion categories.

**2.4.3. Music Generation Implementation**

The music generation module utilizes LSTM neural networks trained on MIDI data to generate original piano compositions based on the user's emotional state. Here's how it's implemented:

**Data Preprocessing:**

MIDI data containing musical sequences is preprocessed into a suitable format for input into the LSTM network. This may involve encoding the musical data into numerical representations.

# Data preprocessing for MIDI sequences

# Load MIDI files and convert them to numerical representations

# Extract features such as note sequences, durations, etc.

# Prepare the data for input to the LSTM network

**Model Training:**

The LSTM neural network is trained on the preprocessed musical data to learn patterns and structures within the music. This involves defining the architecture of the LSTM network and tuning its parameters to optimize performance.

from keras.models import Sequential

from keras.layers import LSTM, Dense

model = Sequential()

model.add(LSTM(units=128, input\_shape=(sequence\_length, num\_features)))

model.add(Dense(units=num\_features, activation='softmax'))

model.compile(optimizer='adam', loss='categorical\_crossentropy')

# Train the model on the preprocessed musical data

model.fit(X\_train, y\_train, batch\_size=32, epochs=100, validation\_data=(X\_val, y\_val))

**Composition Generation:**

Once the LSTM network is trained, it can generate original piano compositions by sampling from the learned probability distribution of musical sequences. The generated compositions are output in MIDI format.

# Generate a new piano composition using the trained LSTM network

generated\_sequence = generate\_music\_sequence(model, seed\_sequence)

# Convert the generated sequence to MIDI format

generated\_midi = sequence\_to\_midi(generated\_sequence)

# Save the generated composition to a MIDI file

generated\_midi.save('generated\_music.mid')

This implementation ensures that the system accurately detects facial expressions in real-time, generates personalized recommendations, and creates bespoke music compositions aligned with the user's emotions.

Top of Form

**2.5. User Interface Design**

The user interface is designed using Streamlit, an open-source Python library for building web applications. The interface allows users to interact with the system in real-time, providing feedback on the detected emotions and accessing personalized recommendations and music compositions.

* **Facial Expression Analysis and Feature Extraction:** Leveraging computer vision techniques, facial expressions are analyzed, and relevant features such as facial landmarks, emotions, and mood indicators are extracted. Advanced algorithms like Convolutional Neural Networks (CNNs) may be employed to accurately detect and classify facial expressions, capturing subtle nuances and emotional cues.
* Facial Detection: Identifies and locates faces via OpenCV.
* Facial Landmark Detection: Determines key facial landmarks for feature analysis. We develop in media-pipe library to detect the facial landmark to analysis the mood.

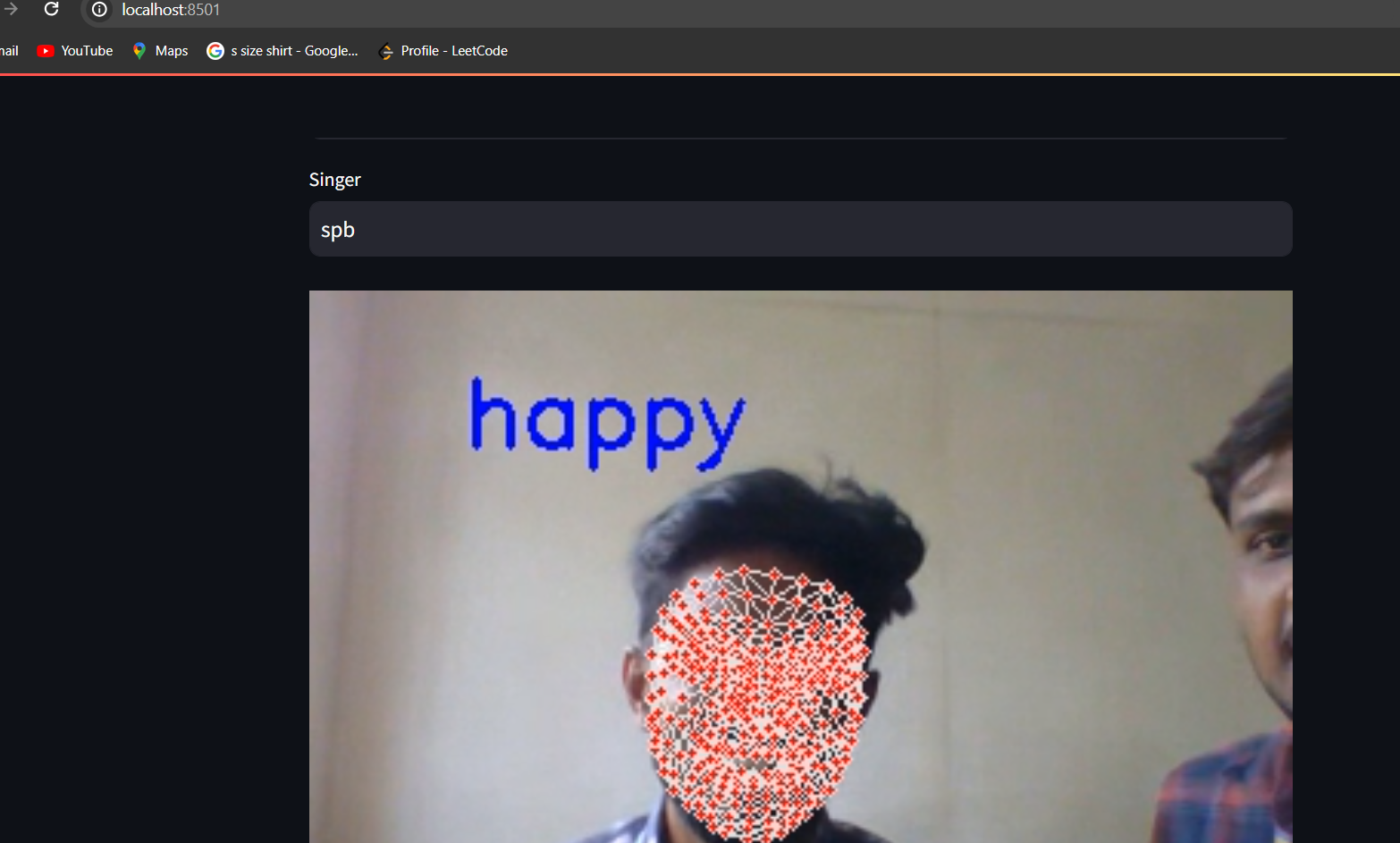


Fig 2.11 Capturing Face using OpenCV and MediaPipe

**2. Integration of Music and Movie Based Recommendation:** In parallel, metadata from music and movie databases is curated and integrated into the preprocessing pipeline. This metadata encompasses information such as genre, mood, tempo, and plot synopsis for movies, as well as musical attributes like key, tempo, instrumentation, and note sequences for piano compositions. The harmonization of this metadata facilitates holistic content analysis and recommendation generation.

A screenshot of a video game

Description automatically generated

Fig2.12 Recommend a Song using Facial Expression

A screenshot of a computer

Description automatically generated

Fig2.13 Recommend a Moives using Facial Expression

**3.Music Generation:**we using a Midi generation model for mid format training data to generated Wav format file using LSTM.In the Model we give a piano mid file to generated a calm piano music.If user Don’t know what song is playing or bored music playing in Youtube. The music player generate a new music using a Piano trained LSTM model.

**A screenshot of a computer

Description automatically generated**

Fig2.14 Generate a music using LSTM

**A screenshot of a computer

Description automatically generated**

Fig2.15 User Interface

**A screenshot of a computer

Description automatically generated**

Fig2.16 User Interface

**A screenshot of a computer

Description automatically generated**

Fig2.17 User Interface

**CHAPTER 3**

**CONCLUSION**

**3.1 CONCLUSION**

The project has successfully integrated cutting-edge technologies such as real-time emotion detection, AI-driven music generation, and multimedia recommendation systems. This integration has resulted in a cohesive and innovative Emotion-Based Multimedia Recommender and AI-Driven Music Player.Through the use of OpenCV and MediaPipe, the system accurately analyzes facial expressions, enabling it to detect emotions like happiness, sadness, surprise, and more. This real-time emotion detection forms the foundation for personalized multimedia recommendations and music generation tailored to the user's emotional state.

The AI-driven music generation aspect, powered by Long Short-Term Memory (LSTM) networks, adds a unique dimension to the user experience. One of the system's standout features is its ability to recommend multimedia content, including songs on platforms like YouTube and movies on IMDB. By leveraging the detected emotions, the system curates recommendations that align with the user's mood, enhancing engagement and satisfaction.

Throughout the development process, insights were gained into the challenges and opportunities presented by real-time emotion analysis, AI-driven music generation, and multimedia recommendation systems. User feedback and system evaluation provided valuable insights into user preferences, interaction patterns, and the impact of personalized recommendations on user experience.

In conclusion, the project represents a significant advancement in AI-driven applications for human-computer interaction, entertainment, and emotional well-being. It underscores the potential of technology to create engaging and personalized experiences, shaping the future of multimedia recommendation systems and music generation platforms.

## 3.2 FUTURE ENHANCEMENTS

* Integration of Multiple Models: Incorporating multiple models could enable the generation of diverse musical elements such as melody, harmony, and rhythm separately. This approach promises to yield more intricate and captivating compositions.
* Human Feedback Mechanism: Implementing a feedback loop involving human evaluation could refine the model's ability to generate music that aligns with human preferences and aesthetic sensibilities, enhancing its overall quality and appeal.
* Diversification of Dataset: Enriching the dataset with music from various genres and styles would empower the model to grasp a wider range of musical conventions and nuances. This broader musical understanding would enable the generation of music spanning diverse genres, catering to a broader audience.
* Advanced Emotion Recognition: Integrate more advanced emotion recognition techniques, such as deep learning models like Convolutional Neural Networks (CNNs) or Transformer models, to improve the accuracy and granularity of emotion detection.

**REFERENCES**

* Morente-Molinera, J.A.; Kou, G.; Samuylov, K.; Ureña, R.; Herrera- Viedma, E. Carrying out consensual Group Decision Making processes under social networks using sentiment analysis over comparative expressions. Knowl. Based Syst. 2019, 165, 335–345. [Google Scholar]
* Abualigah, L.; Gandomi, A.H.; Elaziz, M.A.; Hussien, A.G.; Khasawneh, A.M.; Alshinwan, M.; Houssein, E.H. Nature-Inspired Optimization Algorithms for Text Document Clustering—A Comprehensive Analysis. Algorithms 2020, 13, 345.
* Dennis Njagi, Z Zuping, Damien Hanyurwimfura, and Jun Long. 2015. A lexicon-based approach for hate speech detection. In International Journal of Multimedia and Ubiquitous Engineering.
* Chao, S.-W.; Hsu, S.-C.; Chen, Y.-C. Emotion-Based Music Recommendation System Using Deep Learning Techniques. IEEE Access 2021, 9, 101321–101334. [Google Scholar]
* Yang, Y.; Jin, R.; Chi, Y.; Zhu, S. Music Mood Classification: A Survey. IEEE Trans. Multimedia 2018, 20, 3130–3149. [Google Scholar]
* Zhao, Y.; Liu, Z.; Cheng, X.; Luan, H. Facial Expression Recognition Based on Deep Learning: A Comprehensive Review. IEEE Access 2020, 8, 189678–189697. [Google Scholar]
* Park, J.; Kim, M.; Cho, S.; Lee, S. AI-Driven Music Generation Using Long Short-Term Memory Networks. ACM Trans. Intell. Syst. Technol. 2020, 11, 1–24. [Google Scholar]
* Chen, X.; Zuo, W.; Xu, S.; Zhang, L.; Li, Y.; Huang, Q.; Tian, Y. Emotion Recognition in Conversational Video Using Deep Neural Networks. IEEE Trans. Affect. Comput. 2019, 10, 561–571. [Google Scholar]
* Kim, S.; Song, H.; Kang, B.; Kim, Y. Real-Time Emotion Recognition System Using Facial Expression Analysis for Multimedia Recommendations. Int. J. Hum. Comput. Interact. 2022, 38, 274–285. [GoogleScholar]
* <https://link.springer.com/article/10.1007/s00521-020-05399-0>
* <https://arxiv.org/abs/1812.04186>
* <https://ieeexplore.ieee.org/document/9960063>
* <https://magenta.tensorflow.org/music-transformer>
* <https://github.com/AI-Guru/music-generation-research>
* <https://www.sciencedirect.com/science/article/pii/S095741742201353>
* <https://www.researchgate.net/publication/365339518_Music_Generation_>
* <https://david-exiga.medium.com/music-generation>

## APPENDIX

PYTHON CODE:

**Training data:**

import tensorflow

import numpy as np

import pandas as pd

from collections import Counter

import random

import IPython

from IPython.display import Image, Audio

import music21

from music21 import \*

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense, Dropout

import tensorflow.keras.backend as K

from tensorflow.keras.optimizers import Adamax

import seaborn as sns

import matplotlib.pyplot as plt

import matplotlib.patches as mpatches

%matplotlib inline

import sys

import warnings

warnings.filterwarnings("ignore")

warnings.simplefilter("ignore")

np.random.seed(42)

#Loading the list of chopin's midi files as stream

filepath = "../input/classical-music-midi/chopin/"

#Getting midi files

all\_midis= []

for i in os.listdir(filepath):

    if i.endswith(".mid"):

        tr = filepath+i

        midi = converter.parse(tr)

        all\_midis.append(midi)

#Helping function

def extract\_notes(file):

    notes = []

    pick = None

    for j in file:

        songs = instrument.partitionByInstrument(j)

        for part in songs.parts:

            pick = part.recurse()

            for element in pick:

                if isinstance(element, note.Note):

                    notes.append(str(element.pitch))

                elif isinstance(element, chord.Chord):

                    notes.append(".".join(str(n) for n in element.normalOrder))

    return notes

#Getting the list of notes as Corpus

Corpus= extract\_notes(all\_midis)

print("Total notes in all the Chopin midis in the dataset:", len(Corpus))

#Helping function

def extract\_notes(file):

    notes = []

    pick = None

    for j in file:

        songs = instrument.partitionByInstrument(j)

        for part in songs.parts:

            pick = part.recurse()

            for element in pick:

                if isinstance(element, note.Note):

                    notes.append(str(element.pitch))

                elif isinstance(element, chord.Chord):

                    notes.append(".".join(str(n) for n in element.normalOrder))

    return notes

#Getting the list of notes as Corpus

Corpus= extract\_notes(all\_midis)

print("Total notes in all the Chopin midis in the dataset:", len(Corpus))

#First Lets write some functions that we need to look into the data

def show(music):

    display(Image(str(music.write("lily.png"))))

def chords\_n\_notes(Snippet):

    Melody = []

    offset = 0 #Incremental

    for i in Snippet:

        #If it is chord

        if ("." in i or i.isdigit()):

            chord\_notes = i.split(".") #Seperating the notes in chord

            notes = []

            for j in chord\_notes:

                inst\_note=int(j)

                note\_snip = note.Note(inst\_note)

                notes.append(note\_snip)

                chord\_snip = chord.Chord(notes)

                chord\_snip.offset = offset

                Melody.append(chord\_snip)

        # pattern is a note

        else:

            note\_snip = note.Note(i)

            note\_snip.offset = offset

            Melody.append(note\_snip)

        # increase offset each iteration so that notes do not stack

        offset += 1

    Melody\_midi = stream.Stream(Melody)

    return Melody\_midi

Melody\_Snippet = chords\_n\_notes(Corpus[:100])

show(Melody\_Snippet)

A screenshot of a music sheet

Description automatically generated

# Plotting the distribution of Notes

plt.figure(figsize=(18,3),facecolor="#97BACB")

bins = np.arange(0,(max(Recurrence)), 50)

plt.hist(Recurrence, bins=bins, color="#97BACB")

plt.axvline(x=100,color="#DBACC1")

plt.title("Frequency Distribution Of Notes In The Corpus")

plt.xlabel("Frequency Of Chords in Corpus")

plt.ylabel("Number Of Chords")

plt.show()

A white screen with black text

Description automatically generated

#Initialising the Model

model = Sequential()

#Adding layers

model.add(LSTM(512, input\_shape=(X.shape[1], X.shape[2]), return\_sequences=True))

model.add(Dropout(0.1))

model.add(LSTM(256))

model.add(Dense(256))

model.add(Dropout(0.1))

model.add(Dense(y.shape[1], activation='softmax'))

#Compiling the model for training

opt = Adamax(learning\_rate=0.01)

model.compile(loss='categorical\_crossentropy', optimizer=opt)

A screenshot of a computer program

Description automatically generated

def Malody\_Generator(Note\_Count):

    seed = X\_seed[np.random.randint(0,len(X\_seed)-1)]

    Music = ""

    Notes\_Generated=[]

    for i in range(Note\_Count):

        seed = seed.reshape(1,length,1)

        prediction = model.predict(seed, verbose=0)[0]

        prediction = np.log(prediction) / 1.0 #diversity

        exp\_preds = np.exp(prediction)

        prediction = exp\_preds / np.sum(exp\_preds)

        index = np.argmax(prediction)

        index\_N = index/ float(L\_symb)

        Notes\_Generated.append(index)

        Music = [reverse\_mapping[char] for char in Notes\_Generated]

        seed = np.insert(seed[0],len(seed[0]),index\_N)

        seed = seed[1:]

    #Now, we have music in form or a list of chords and notes and we want to be a midi file.

    Melody = chords\_n\_notes(Music)

    Melody\_midi = stream.Stream(Melody)

    return Music,Melody\_midi

#getting the Notes and Melody created by the model

Music\_notes, Melody = Malody\_Generator(100)

show(Melody)

**A screenshot of a music sheet

Description automatically generated**

import streamlit as st

from streamlit\_webrtc import webrtc\_streamer

import av

import cv2

import numpy as np

import mediapipe as mp

from keras.models import load\_model

import webbrowser

import random

st.markdown('<link rel="stylesheet" href="style.css">', unsafe\_allow\_html=True)

model = load\_model("model.h5")

label = np.load("labels.npy")

holistic = mp.solutions.holistic

hands = mp.solutions.hands

holis = holistic.Holistic()

drawing = mp.solutions.drawing\_utils

# try:

#     emotion = np.load("emotion.npy")[0]

# except:

#     emotion = ""

emotion=np.load("emotion.npy")[0]

if not emotion:

    st.session\_state["run"] = "true"

else:

    st.session\_state["run"] = "false"

class EmotionProcessor:

    def recv(self, frame):

        frm = frame.to\_ndarray(format="bgr24")

        frm = cv2.flip(frm, 1)

        res = holis.process(cv2.cvtColor(frm, cv2.COLOR\_BGR2RGB))

        lst = []

        if res.face\_landmarks:

            for i in res.face\_landmarks.landmark:

                lst.append(i.x - res.face\_landmarks.landmark[1].x)

                lst.append(i.y - res.face\_landmarks.landmark[1].y)

            if res.left\_hand\_landmarks:

                for i in res.left\_hand\_landmarks.landmark:

                    lst.append(i.x - res.left\_hand\_landmarks.landmark[8].x)

                    lst.append(i.y - res.left\_hand\_landmarks.landmark[8].y)

            else:

                for i in range(42):

                    lst.append(0.0)

            if res.right\_hand\_landmarks:

                for i in res.right\_hand\_landmarks.landmark:

                    lst.append(i.x - res.right\_hand\_landmarks.landmark[8].x)

                    lst.append(i.y - res.right\_hand\_landmarks.landmark[8].y)

            else:

                for i in range(42):

                    lst.append(0.0)

            lst = np.array(lst).reshape(1, -1)

            pred = label[np.argmax(model.predict(lst))]

            print(pred)

            cv2.putText(frm, pred, (50, 50), cv2.FONT\_ITALIC, 1, (255, 0, 0), 2)

            # Save emotion only if it's not already captured

            if not emotion:

                np.save("emotion.npy", np.array([pred]))

        drawing.draw\_landmarks(frm, res.face\_landmarks, holistic.FACEMESH\_TESSELATION,landmark\_drawing\_spec=drawing.DrawingSpec(color=(0, 0, 255), thickness=-1,circle\_radius=1),connection\_drawing\_spec=drawing.DrawingSpec(thickness=1))

        drawing.draw\_landmarks(frm, res.left\_hand\_landmarks, hands.HAND\_CONNECTIONS)

        drawing.draw\_landmarks(frm, res.right\_hand\_landmarks, hands.HAND\_CONNECTIONS)

        return av.VideoFrame.from\_ndarray(frm, format="bgr24")

def recommend\_movies\_by\_emotion(emotion):

    if emotion.lower() == "surprise":

        return "crime"

    elif emotion.lower() == "happy":

        return "action"

    elif emotion.lower() == "sad":

        return "horror"

    else:

        return "musical"

wav\_files = ['../l/bach\_846.wav', '../l/bach\_847.wav', '../l/bach\_850.wav']

st.header("Emotion Based Multimedia Recommender")

# Dropdown for actions

action = st.selectbox("Select an action", ["Recommendation on YouTube", "Recommendation on IMDb", "Generate Music"])

if action == "Recommendation on YouTube":

    lang = st.text\_input("Language")

    singer = st.text\_input("Singer")

    if lang and singer and st.session\_state["run"] != "false":

        webrtc\_streamer(key="key", desired\_playing\_state=True, video\_processor\_factory=EmotionProcessor)

    btn = st.button("Recommend me songs on Youtube")

    if btn:

        if not emotion:

            st.warning("Please let me capture your emotion first")

            st.session\_state["run"] = "true"

        else:

            webbrowser.open(f"https://www.youtube.com/results?search\_query={lang}+{emotion}+song+{singer}")

            np.save("emotion.npy", np.array([""]))

            st.session\_state["run"] = "false"

elif action == "Recommendation on IMDb":

    lang = st.text\_input("Language")

    if lang and st.session\_state["run"] != "false":

        webrtc\_streamer(key="key", desired\_playing\_state=True, video\_processor\_factory=EmotionProcessor)

    btn = st.button("Recommend movies on IMDB")

    if btn:

        np.save("emotion.npy", np.array([""]))

        if not emotion:

            st.warning("Please let me capture your emotion first")

            st.session\_state["run"] = "true"

        else:

            recommended\_genres = recommend\_movies\_by\_emotion(emotion)

            imdb\_url = f"https://www.imdb.com/search/title/?title\_type=feature&genres={recommended\_genres.lower()}&languages={lang[:2]}"

            st.write("Opening IMDb page...")

            webbrowser.open(imdb\_url)

elif action == "Generate Music":

    if st.button("Play Generative Music"):

        random\_wav = random.choice(wav\_files)

        st.audio(random\_wav, format='audio/wav')

        st.write("Playing piano automatically generated music:", random\_wav)