Real-Time Emotion-Driven Music Recommendation Using Facial Expression Analysis and Spotify

A Project Report

Submitted in partial fulfillment of the requirements

Project Team:

Siddharth Patil
Vansh Jalora
Abhay Varnekar
Vedant Anasune

Department of Computer Engineering
Veermata Jijabai Technological Institute (VJTI)
Mumbai, India

November 11, 2025

Abstract

Music playlists are typically static and fail to adapt to a user's changing emotional state. This project presents an innovative solution: a live emotion-responsive music player that curates Spotify playlists based on real-time facial expression analysis. The system employs OpenCV Haar Cascade for facial detection and extraction from webcam input, followed by preprocessing operations including grayscaling, resizing, and normalization. A custom Convolutional Neural Network (CNN) performs discrete emotion classification across seven categories: happy, sad, angry, fear, surprise, disgust, and neutral. To ensure stability, emotion updates are processed at five-second intervals. An asynchronous Spotify controller monitors playback continuously and triggers predefined playlist URIs mapped to detected emotions when songs finish or emotional states change. This report discusses the system architecture, implementation methodology, experimental results, and considerations for responsible AI deployment, including limitations related to variable lighting conditions and user privacy.

Keywords: Emotion-aware systems, Expression recognition, Music recommendation, Streaming integration, Facial analysis, Computational efficiency, User privacy

Contents

1 Introduction

1.1 Motivation and Background

The intrinsic connection between auditory experiences and human emotional states has been extensively validated through interdisciplinary research spanning psychology, neuroscience, and computational modeling. Contemporary streaming platforms such as Spotify and Apple Music offer sophisticated Application Programming Interfaces (APIs) that provide access to vast music catalogs and powerful playback controls. While their recommendation engines effectively leverage explicit behavioral signals—including user history, curated collections, and manual searches—they largely ignore implicit emotional cues that could significantly enhance user experience.

This project explores the opportunity to complement existing recommendation systems by introducing an implicit input modality: the user's real-time emotional state, detected through automated facial expression analysis. The discipline of affective computing establishes frameworks for narrowing the behavioral-emotional divide, facilitating systems that perceive, decode, and adapt to human affect in real-time.

1.2 Project Objectives

The primary objectives of this project are:

1. To develop a high-throughput vision processing pipeline capable of real-time facial detection, alignment, and preprocessing for discrete seven-class emotion prediction
2. To implement a direct mapping mechanism that triggers specific Spotify playlist URIs based on detected emotion classes
3. To design an asynchronous architecture that decouples live video processing from emotion analysis and Spotify monitoring for optimal user experience
4. To validate the system's practical viability and computational feasibility through controlled testing
5. To address considerations regarding sensing reliability, individual privacy rights, and algorithmic equity

1.3 Scope and Innovations

This work presents an end-to-end platform for emotion-responsive music selection, employing continuous webcam monitoring for expression analysis and dynamically adjusting Spotify suggestions based on detected emotional patterns. The system operates entirely on local hardware, preventing raw video transmission and prioritizing user privacy while maintaining interactive responsiveness.

The approach illustrates pathways toward context-responsive listening environments while raising important considerations regarding the responsible deployment of affective computing technologies in consumer applications.

2 Literature Review and Theoretical Background

2.1 Emotion Modeling Frameworks

The study of computational affect emerged from converging advances in computer science, psychological theory, and interaction design. Two fundamental approaches characterize emotion representation in computational systems:

2.1.1 Discrete Emotion Models

The discrete model, exemplified by Ekman's foundational six-emotion taxonomy (joy, sorrow, anger, fear, surprise, disgust) augmented with a neutral state, facilitates straightforward classification objectives. This categorical approach aligns directly with our implementation, which maps seven distinct emotion labels to specific user actions and playlist selections. The discrete model offers several advantages for practical applications:

- Clear, interpretable classification targets
- Direct mapping to user-facing actions
- Alignment with human intuition about emotional categories
- Simplified system architecture and debugging

2.1.2 Continuous Emotion Models

Alternatively, continuous frameworks, particularly Russell's circumplex structure, position affect along orthogonal valence (negative-to-positive) and arousal (calm-to-excited) dimensions. While enabling nuanced affective characterization and capturing gradients of emotional experience, this approach introduces complexity in mapping continuous values to discrete playlists. For this project, the discrete model was selected to enable more direct emotion-to-playlist correspondence.

2.2 Facial Expression Recognition Technology

2.2.1 Historical Development

Early expression recognition systems depended on manually-engineered visual descriptors such as Histogram of Oriented Gradients (HOG) and Local Binary Patterns (LBP). These hand-crafted features required domain expertise and often struggled with variations in lighting, pose, and individual facial characteristics.

2.2.2 Modern Deep Learning Approaches

Contemporary approaches utilize convolutional neural network architectures, achieving superior accuracy under uncontrolled environmental conditions. The shift to learned representations allows models to automatically discover salient features from data, improving generalization across diverse populations and imaging conditions.

2.2.3 Benchmark Datasets

Large-scale benchmark corpora such as AffectNet provide categorical annotations across diverse populations, enabling robust model training. These datasets capture real-world variability in expression, demographics, and imaging conditions, essential for developing systems that generalize beyond laboratory settings.

2.2.4 Temporal Processing

Prediction stability benefits from temporal aggregation methods, such as fixed-interval updates, that suppress frame-level variations and noise. Detection frameworks like OpenCV's Haar Cascades ensure consistent facial region tracking across video sequences, while local processing minimizes transmission delays and privacy exposure risks.

2.3 Musical Emotion and Recommendation

Musical emotion analysis seeks computational models of affect conveyed or elicited by audio content. Prior investigations have explored affect-to-playlist mappings with validation through subjective user assessments. Research indicates that musical features such as tempo, mode, timbre, and harmonic structure correlate with perceived emotional content.

This project contributes to the field by implementing a direct mapping from discrete facial emotions to pre-curated Spotify playlist URIs, acting as a practical bridge between observed facial affect and musical experience. The approach prioritizes real-time responsiveness over complex content-based music analysis.

2.4 Responsible AI Development

Expression analysis technologies risk perpetuating training corpus biases, particularly affecting demographic groups defined by skin tone, age, and gender. Historical research has documented significant accuracy disparities across demographic categories in commercial facial analysis systems.

Ethical deployment mandates several protective measures:

- Informed consent protocols with clear explanation of system capabilities
- Real-time activity visualization to maintain user awareness
- Device-local computation to minimize data exposure
- Transparent data governance with no persistent storage of video
- Regular bias auditing across demographic groups
- User control mechanisms including system disable options

3 System Architecture and Design

3.1 Architectural Overview

The platform architecture employs an asynchronous design pattern, separating computational tasks to ensure responsive user interface performance. The system is organized into three logically independent components that operate with different update intervals and priorities:

3.1.1 Display Component

This component maintains sole responsibility for capturing the webcam feed and rendering it to the user in real-time. It visualizes bounding boxes provided by the face detection subsystem, ensuring the video stream remains smooth and responsive regardless of backend processing loads. The display operates at native webcam frame rates (typically fifteen to thirty frames per second).

3.1.2 Emotion Component

Operating on a fixed five-second interval, this component retrieves the latest frame from the video buffer and executes the complete emotion recognition pipeline: facial detection, image preprocessing, and CNN inference. The resulting emotion classification updates the user's current affective state. This interval prevents classification jitter and provides more stable emotion readings by filtering out momentary expressions or artifacts.

3.1.3 Spotify Component

This component monitors the user's playback status every two seconds through the Spotify API. Its primary logic detects when a song has concluded or when the detected emotion has changed significantly. Upon such events, it triggers playback of a new playlist matching the user's current mood classification.

3.2 Design Rationale

This modular architecture decouples the user interface from computational processing, ensuring that network latency from Spotify API calls or computational spikes in CNN inference do not cause video freezing or interface lag. Each component can be optimized, tested, and scaled independently, facilitating maintenance and future enhancements.

3.3 Data Flow and Processing Pipeline

The complete data flow progresses through the following stages:

1. **Video Acquisition:** Continuous webcam capture provides raw RGB frames
2. **Facial Localization:** Haar Cascade detector identifies facial regions of interest
3. **Image Preprocessing:** Grayscale conversion, resolution standardization, and normalization prepare the facial region for neural network input
4. **Emotion Classification:** CNN inference produces probability distribution over seven emotion classes
5. **Temporal Stabilization:** Five-second sampling interval filters transient expressions
6. **Playlist Mapping:** Discrete emotion maps to specific Spotify playlist URI

7. **Playback Control:** Spotify controller initiates playlist playback when appropriate

4 Implementation Methodology

4.1 Facial Detection and Localization

The initial stage of the vision pipeline involves localizing the user's face within the webcam frame. The system employs OpenCV's high-performance Haar Cascade classifier, specifically utilizing the pre-trained frontal face detection model. This classical computer vision approach offers several advantages: computational efficiency on CPU hardware, deterministic behavior, and no additional training requirements.

4.1.1 Detection Algorithm

The Haar Cascade operates through a multi-scale sliding window approach. The detector scans the image at multiple scales to identify faces at varying distances from the camera. Three critical parameters govern detection behavior:

Scale Factor Set to 1.1, this parameter creates a scale pyramid where each successive layer is ten percent smaller than the previous. This multi-scale approach allows the fixed-size detector window to identify faces of different sizes corresponding to varying distances from the camera.

Minimum Neighbors Set to 5, this parameter implements a consensus-voting mechanism for non-maximum suppression. A detection is validated only when at least five overlapping detection windows confirm the presence of a face, dramatically reducing false positive detections.

Minimum Size Set to 30×30 pixels, this constraint filters out detections smaller than the specified dimensions, ignoring noise and irrelevant small-scale features that cannot represent actual faces at the system's resolution.

For each validated detection, the algorithm returns a bounding box defined by coordinates and dimensions. This Region of Interest (ROI) is extracted from the original frame for subsequent processing stages.

4.2 Image Preprocessing Pipeline

Once the facial ROI is isolated, it undergoes a standardized sequence of transformations to condition it for neural network input. This preprocessing pipeline executes sequentially on every detected face:

4.2.1 Grayscale Conversion

The three-channel RGB image is converted to single-channel grayscale representation using a perceptually-weighted luminance formula:

$$\text{Gray} = 0.299 \times R + 0.587 \times G + 0.114 \times B \tag{1}$$

This transformation reduces computational complexity by two-thirds while focusing the model on morphological features rather than color information, which provides minimal discriminative value for expression recognition.

4.2.2 Resolution Standardization

The grayscale image is resized to a fixed resolution of 48×48 pixels using bilinear interpolation. This standardization ensures consistent input tensor dimensions for the neural network, regardless of the original detection scale. The 48×48 resolution represents a balance between computational efficiency and retention of discriminative facial features.

4.2.3 Tensor Conversion and Scaling

The processed image is converted into a floating-point tensor, scaling pixel values from the integer range [0, 255] to floating-point range [0.0, 1.0]. This normalization facilitates stable gradient computation during neural network operations.

4.2.4 Statistical Normalization

Finally, the tensor undergoes normalization with mean $= 0.5$ and standard deviation $= 0.5$, centering the data distribution:

$$I_{\text{norm}} = \frac{I_{\text{tensor}} - 0.5}{0.5} = 2 \times I_{\text{tensor}} - 1 \tag{2}$$

This transformation maps the pixel range from [0.0, 1.0] to [1.0, 1.0], improving neural network convergence by centering activations around zero and facilitating balanced gradient flow during backpropagation.

4.3 Training Data and Augmentation

4.3.1 Dataset Selection

Model development incorporates the AffectNet dataset, a large-scale corpus providing over seventeen thousand facial images with categorical emotion annotations. The dataset includes images collected from web sources, capturing natural variability in expression, demographics, pose, and imaging conditions.

4.3.2 Emotion Categories

The system utilizes seven discrete emotion classes from the dataset: Angry, Disgust, Fear, Happy, Sad, Surprise, and Neutral. This classification scheme extends Ekman's original six basic emotions with a neutral category to handle non-expressive or ambiguous faces.

4.3.3 Data Format

The YOLO-format version of AffectNet provides pre-computed bounding boxes for facial regions, which are used to extract the facial ROI before applying the preprocessing pipeline. This format facilitates efficient data loading and eliminates the need for additional face detection during training.

4.4 Neural Network Architecture

The expression recognition model employs a custom Convolutional Neural Network architecture designed to process 48×48 grayscale images. The network consists of a feature extraction backbone and a classification head.

4.4.1 Convolutional Layers

The backbone comprises multiple convolutional blocks, each containing a convolution operation, max pooling, and dropout regularization. The convolution operation applies learned 3×3 filters to input feature maps:

$$Y[i,j] = \text{ReLU}\left(\sum_{m=0}^{2}\sum_{n=0}^{2} X[i+m-1, j+n-1] \times W[m,n] + b\right) \tag{3}$$

With padding of one pixel, spatial dimensions are preserved within each convolutional layer. The Rectified Linear Unit (ReLU) activation introduces non-linearity, enabling the network to learn complex decision boundaries.

4.4.2 Pooling Operations

Max pooling layers downsample feature maps by extracting maximum values over 2×2 windows:

$$P[i,j] = \max(X[2i,2j], X[2i+1,2j], X[2i,2j+1], X[2i+1,2j+1]) \tag{4}$$

This operation reduces spatial dimensions by half while providing translational invariance and reducing computational requirements in subsequent layers.

4.4.3 Dimensionality Progression

The sequential application of convolutional and pooling layers progressively reduces spatial dimensions while increasing feature depth:

- Input: $48 \times 48 \times 1$ (grayscale image)
- After Conv1/2 and Pool1: $24 \times 24 \times 64$
- After Conv3 and Pool2: $12 \times 12 \times 128$
- After Conv4 and Pool3: $6 \times 6 \times 128$
- Flattened: 4608 features

4.4.4 Regularization

Dropout layers randomly zero a fraction of neurons during training (25% or 50% depending on layer depth), preventing overfitting by forcing the network to learn redundant representations. Remaining activations are scaled by $1/(1-p)$ to maintain expected output magnitude. During inference, dropout is disabled.

4.4.5 Fully Connected Layers

After flattening the spatial feature maps, fully connected (dense) layers perform matrix multiplication with learned weights:

$$Y = \mathrm{ReLU}(X \times W^T + b) \tag{5}$$

The first dense layer reduces the 4608 features to 1024 dimensions, while the final output layer produces seven raw scores (logits), one per emotion class.

4.4.6 Output and Classification

The output logits are passed through a softmax function to produce a probability distribution:

$$\mathrm{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{7} e^{z_j}} \tag{6}$$

The predicted emotion is determined by the class with maximum probability (argmax operation).

4.5 Training Methodology

4.5.1 Loss Function

The model is trained using categorical cross-entropy loss, which measures divergence between predicted probability distribution $\hat{y}$ and one-hot encoded true label y:

$$\mathcal{L}_{CE} = -\sum_{i=1}^{7} y_i \log(\hat{y}_i) \tag{7}$$

This loss function is particularly appropriate for multi-class classification, providing strong gradients for confident incorrect predictions while allowing confident correct predictions to contribute minimal loss.

4.5.2 Optimization

Standard stochastic gradient descent with momentum or adaptive learning rate methods (such as Adam) are employed to minimize the loss function. The optimizer adjusts network weights iteratively based on computed gradients, progressively improving classification accuracy on the training data.

4.6 Temporal Stabilization

To create a stable user experience and avoid playlist changes triggered by momentary expression fluctuations, the emotion component updates the user's affective state on a fixed five-second interval rather than processing every video frame. This temporal sampling acts as a strong filter against spurious expressions, ensuring that playlist transitions reflect sustained emotional states rather than transient facial movements.

4.7 Spotify Integration

4.7.1 Authentication Protocol

The system integrates with Spotify using the OAuth 2.0 Authorization Code flow, which provides secure access to user playback control while maintaining proper authorization boundaries. The authentication process proceeds as follows:

1. The application requests authorization with required scopes: user-modify-playback-state and user-read-playback-state
2. The user is directed to Spotify's authentication page in their web browser
3. Upon successful login, Spotify redirects to a local callback server running on localhost:8888
4. The callback server captures the authorization code from the redirect URL
5. The application exchanges the authorization code for access and refresh tokens
6. Tokens are stored locally for subsequent API requests

4.7.2 Playback Control

Once authenticated, the Spotify component identifies the user's active playback device and begins monitoring. The core functionality implements a direct mapping between the seven detected emotion classes and specific pre-curated Spotify playlist URIs. For example:

- Happy → "Happy Hits" playlist
- Sad → "Sad Lofi" playlist
- Angry → "Intense Rock" playlist
- Fear → "Calm Ambient" playlist
- Surprise → "Upbeat Pop" playlist
- Disgust → "Alternative Indie" playlist
- Neutral → "Chill Mix" playlist

When the detected emotion changes, the playback function initiates the corresponding playlist on the active device. A cooldown mechanism prevents rapid playlist switching within short time windows, ensuring musical continuity.

5 Experimental Evaluation

5.1 Implementation Environment

5.1.1 Hardware Specifications

All testing was conducted on standard consumer laptop hardware: Intel Core i7 14th generation processors operating at 4 GHz base frequency, with sixteen gigabytes of RAM.

Notably, discrete GPU acceleration was not employed, demonstrating the system's viability on widely available consumer hardware.

5.1.2 Software Stack

The implementation leverages several key software libraries: OpenCV for computer vision operations, PyTorch for neural network implementation and inference, Spotipy for Spotify API integration, and FastAPI for the OAuth callback server. All processing occurs locally on the client device.

5.2 Performance Metrics

5.2.1 Classification Accuracy

To evaluate model performance, two primary metrics were tracked during training and validation:

Categorical Accuracy The percentage of predictions where the model's top-predicted class correctly matches the true label:

$$\text{Accuracy} = \frac{\text{Correct Predictions}}{\text{Total Predictions}} \tag{8}$$

Cross-Entropy Loss The optimization objective quantifying divergence between predicted and true distributions:

$$\mathcal{L}_{CE} = -\sum_{i=1}^{7} y_i \log(\hat{y}_i) \tag{9}$$

5.2.2 Results Analysis

The trained model achieves a final validation accuracy of 64% with validation loss of 1.43. The learning curves reveal significant divergence between training and validation metrics, indicating overfitting. While training accuracy approaches 100% with training loss minimizing to approximately 0.5, validation accuracy plateaus around 64% after four to five epochs.

This generalization gap suggests the model has memorized training data patterns but struggles with unseen examples. Despite this limitation, 64% accuracy on a challenging seven-class problem with inherently ambiguous categories represents acceptable performance for a proof-of-concept system. Common misclassifications occur between perceptually similar categories: neutral versus sad, fear versus surprise, and anger versus disgust—patterns that align with known challenges in expression recognition research.

The five-second emotion update interval mitigates the impact of individual misclassifications by aggregating temporal evidence, reducing erroneous playlist transitions that would degrade user experience.

5.3 Computational Performance

5.3.1 Algorithmic Complexity

The system's computational requirements arise from two primary operations:

Haar Cascade Detection Time complexity is $O(n \times m \times s \times f)$, where n and m represent image dimensions, s denotes the number of scale levels in the pyramid, and f represents the number of features in the cascade. Modern implementations employ early rejection and integral image optimizations to achieve real-time performance.

CNN Inference Complexity is dominated by convolutional operations, approximately $O(\sum k \cdot h \cdot w \cdot c)$, where k is kernel size, (h, w) are feature map dimensions, and c represents channel count. The relatively small 48×48 input resolution and modest network depth enable efficient CPU inference.

5.3.2 Measured Performance

In practice, the live video display operates between eighteen and twenty-five frames per second on CPU hardware. The emotion detection loop, operating asynchronously, consistently satisfies its five-second interval requirement with average processing time per frame under 500 milliseconds. This leaves substantial computational headroom for other system processes and maintains responsive user interaction.

5.4 User Experience Observations

Informal testing with multiple users revealed several qualitative findings:

- Users appreciated the system's responsiveness and found the five-second update interval appropriately balanced between reactivity and stability
- The visual feedback of facial detection bounding boxes increased user confidence in system operation
- Playlist transitions felt natural when occurring between songs rather than mid-track
- Lighting conditions significantly impacted detection reliability, with dim or harsh lighting degrading performance
- Users expressed privacy satisfaction with local processing and no data storage

6 Limitations and Challenges

6.1 Real-Time System Limitations

6.1.1 Performance Constraints

The continuous processing loop introduces several performance challenges. The combined latency of facial detection and CNN inference adds 200–500 milliseconds per frame. Continuous video capture and processing are CPU and RAM intensive, leading to significant battery drain on mobile hardware and thermal issues during extended operation.

6.1.2 Temporal Resolution

The five-second emotion check interval may miss rapid emotional transitions or fail to capture brief affective states. The system lacks contextual understanding and cannot distinguish genuine expressions from posed or exaggerated facial movements. Additionally, mid-song playlist transitions can create jarring user experiences, particularly during musically cohesive compositions.

6.2 Technical Architecture Limitations

6.2.1 Hardware Dependencies

The system fundamentally requires webcam access and stable internet connectivity for Spotify API communication. While GPU acceleration is optional, it significantly improves CNN inference performance. The architecture assumes desktop or laptop form factors and does not currently support mobile or embedded devices.

6.2.2 Software Dependencies

The implementation exhibits OS-specific dependencies and requires precise version matching between libraries (OpenCV, PyTorch, Spotipy). A pre-trained model file must be present in the expected location, and path specifications may require adjustment across different system configurations.

6.3 Scalability Limitations

The current architecture functions as a local-only proof-of-concept. It cannot handle concurrent users, lacks cloud deployment infrastructure, and does not integrate with persistent storage to learn user preferences or track historical patterns. The system cannot operate offline and provides no functionality when network connectivity is unavailable.

6.4 Integration Constraints

The implementation is tightly coupled to Spotify's platform with no support for alternative music services such as Apple Music or YouTube Music. Desktop-only operation prevents mobile deployment, and the fixed playlist mapping provides no personalization based on individual music taste beyond manual playlist curation.

6.5 Bias and Fairness Concerns

Expression recognition models trained on public datasets may exhibit accuracy disparities across demographic groups defined by age, gender, and skin tone. The AffectNet dataset, while large and diverse, may not equally represent all populations. Deployment without bias auditing risks providing degraded service quality to underrepresented demographic groups.

7 Future Directions

7.1 Multimodal Emotion Detection

Integrating the current facial analysis CNN with additional modalities would provide more robust emotion detection. Voice analysis through recurrent neural networks could capture prosodic and acoustic affective cues, while text sentiment analysis via transformer models could incorporate chat or status messages. Sensor fusion combining these inputs would reduce single-modality failure points and provide more context-aware emotion signals.

7.2 Temporal Emotion Modeling

Replacing the fixed five-second interval with sequential models such as Long Short-Term Memory (LSTM) networks would enable analysis of emotion trends over time. This approach could anticipate mood shifts based on temporal patterns, providing smoother and more proactive playlist adjustments. Hidden Markov Models or other state-space models could explicitly represent emotion state transitions.

7.3 Platform Expansion

Extending the API controller to support Apple Music, YouTube Music, and other streaming services would increase accessibility and user choice. Developing mobile and wearable device implementations would enable emotion-responsive music in diverse contexts beyond desktop computing.

7.4 Personalization and Learning

Implementing user preference learning through reinforcement learning or collaborative filtering would personalize playlist selections beyond generic emotion mappings. Historical playback data could reveal individual music taste patterns, while explicit feedback mechanisms (skip, like, dislike) could fine-tune recommendations over time.

7.5 Clinical and Therapeutic Applications

The technology could be adapted as a non-intrusive mental health monitoring tool. Continuous affective state tracking might aid in early detection of mood disorders, provide quantitative data for therapy progress assessment, or facilitate personalized music therapy interventions. Such applications would require additional validation, clinical partnerships, and enhanced privacy protections.

8 Ethical Considerations and Responsible Deployment

8.1 Privacy Protection

The system architecture prioritizes privacy through several design choices: all video processing occurs locally on user devices, no raw video data is transmitted or stored, and user data never leaves the local machine. Authentication tokens are stored securely on the local filesystem with appropriate permissions.

8.2 Informed Consent

Deployment requires clear user consent with explanation of system capabilities and limitations. Users should understand what data is collected (facial images), how it is processed (local emotion classification), and what actions result (playlist changes). Visual indicators should communicate when the system is actively monitoring.

8.3 Bias Mitigation

Responsible deployment requires regular bias auditing across demographic groups to identify and address performance disparities. Model training should incorporate diverse, representative datasets, and fairness metrics should be monitored during development and deployment. Users should be informed if the system may perform suboptimally for their demographic group.

8.4 User Agency and Control

Users must maintain control over system operation with clear disable mechanisms, manual override options, and configuration controls. The system should respect user preferences regarding monitoring frequency, playlist selection criteria, and sensitivity thresholds. Transparency about system confidence levels helps users understand when classifications may be uncertain.

9 Conclusion

This project demonstrates the feasibility and potential value of discrete emotion-responsive music recommendation systems. By directly mapping facial expressions to curated Spotify playlists, the platform achieves acceptable classification accuracy, maintains low latency, and operates on consumer hardware while preserving user privacy through local processing.

The system successfully integrates computer vision, deep learning, and streaming API technologies into a cohesive user experience. The asynchronous architecture balances responsiveness with stability, while the five-second emotion update interval filters transient expressions. Authentication and playback control through Spotify's OAuth 2.0 framework provides secure, authorized access to user music libraries.

Experimental evaluation reveals both capabilities and limitations. The 64% validation accuracy on seven-class emotion recognition represents reasonable performance for a proof-of-concept, though significant overfitting indicates opportunities for improved generalization. Computational performance meets real-time requirements on standard CPU hardware, demonstrating practical viability for consumer deployment.

Future work should address identified limitations through multimodal sensing, temporal modeling, platform expansion, personalization, and bias mitigation. As emotion-responsive computing becomes increasingly prevalent in consumer applications, sustained attention to fairness, transparency, and user agency remains essential for responsible innovation in affective human-computer interaction.

The convergence of affective computing, ubiquitous sensing, and rich media APIs creates opportunities for context-aware applications that enhance human experience. This project contributes one example of how implicit emotional signals can complement explicit user input, suggesting pathways toward more adaptive and responsive interactive systems.

Acknowledgments

We express deep gratitude to Ms. Noshin Sabuwala for her thoughtful guidance and encouragement throughout this project. Her expertise in machine learning and human-computer interaction significantly influenced our approach and implementation choices.

We also extend heartfelt thanks to our team members for their collaboration, commitment, and collective drive. This work represents true team effort across computer vision, machine learning, software architecture, and API integration.

Finally, we acknowledge the developers of the open-source tools and datasets that made this project possible: the OpenCV community, PyTorch team, Spotify API developers, and creators of the AffectNet dataset.

References

[1] R. W. Picard, *Affective Computing.* Cambridge, MA, USA: MIT Press, 1997.

[2] J. A. Russell, "A circumplex model of affect," *Journal of Personality and Social Psychology*, vol. 39, no. 6, pp. 1161–1178, 1980.

[3] A. Mollahosseini, B. Hasani, and M. H. Mahoor, "AffectNet: A database for facial expression, valence, and arousal computing in the wild," *IEEE Transactions on Affective Computing*, vol. 10, no. 1, pp. 18–31, Jan.–Mar. 2019.

[4] F. Kök, "AffectNet-YOLO format," Kaggle, 2023. [Online]. Available: https://www.kaggle.com/datasets/fatihkgg/affectnet-yolo-format

[5] A. Aljanaki, Y.-H. Yang, and M. Soleymani, "Developing a benchmark for emotional analysis of music," *PLoS ONE*, vol. 12, no. 3, p. e0173392, 2017.

[6] K. Howell and M. E. P. Davies, "Deep MIR: A survey of deep learning for music information retrieval," *ACM Computing Surveys*, vol. 55, no. 2, pp. 1–46, Feb. 2023.

[7] T. Eerola and J. K. Vuoskoski, "Emotional responses to music: The need to consider underlying mechanisms," *Behavioral and Brain Sciences*, vol. 31, no. 5, pp. 559–575, 2008.

[8] Y.-H. Yang and H. H. Chen, *Machine Recognition of Music Emotion: A Review.* New York, NY, USA: ACM Transactions on Intelligent Systems and Technology, 2012.

[9] J. Buolamwini and T. Gebru, "Gender shades: Intersectional accuracy disparities in commercial gender classification," in *Proc. Conference on Fairness, Accountability and Transparency (FAT*)*, New York, NY, USA, 2018, pp. 77–91.

[10] D. E. King, "Dlib-ml: A machine learning toolkit," *Journal of Machine Learning Research*, vol. 10, pp. 1755–1758, Jul. 2009.

[11] A. Howard et al., “Searching for MobileNetV3,” in *Proc. IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea (South), Oct. 2019, pp. 1314–1324.

[12] M. Tan and Q. V. Le, “EfficientNet: Rethinking model scaling for convolutional neural networks,” in *Proc. 36th International Conference on Machine Learning (ICML)*, Long Beach, CA, USA, 2019, pp. 6105–6114.

A System Workflow Diagrams

This appendix provides detailed descriptions of the system workflow components illustrated in the original research paper.

A.1 Background Spotify Monitor Flow

The Spotify monitoring component operates on a two-second polling interval, continuously checking playback status and responding to state changes. The workflow follows these steps:

1. **Monitor Loop (2-second interval):** The component polls the Spotify API to retrieve current playback information including track progress, playlist status, and active device.
2. **Song Change Detection:** The monitor compares the current track identifier with the previously stored track to detect song transitions.
3. **Cooldown Check:** Before initiating any playlist change, the system verifies that at least ten seconds have elapsed since the last transition, preventing rapid switching that degrades user experience.
4. **Emotion Comparison:** The monitor retrieves the most recent emotion classification from the emotion component and compares it with the emotion corresponding to the current playlist.
5. **Playlist Switching:** If the detected emotion differs from the current playlist mapping and the cooldown period has expired, the system initiates playback of the playlist corresponding to the new emotion.
6. **Cooldown Activation:** Upon successful playlist transition, a ten-second cooldown timer begins, during which no additional transitions will occur regardless of detected emotion changes.

This design ensures musical continuity while remaining responsive to sustained emotional state changes, balancing reactivity with stability.

A.2 User and Backend Journey

The complete user interaction flow integrates frontend display, backend processing, and service integration:

1. **Video Display:** The user sees continuous webcam feed with overlaid facial detection bounding boxes rendered in real-time, providing immediate visual feedback about system operation.

2. **Five-Second Processing Cycle:** Every five seconds, the emotion component captures the current frame and processes it through the complete pipeline: facial detection, preprocessing, and CNN classification.

3. **Face Detection Visualization:** When a face is successfully detected, a green bounding box appears around the facial region, confirming that the system is actively monitoring and processing expressions.

4. **Emotion Classification:** The CNN produces probability distributions over seven emotion categories, selecting the class with maximum probability as the current emotional state.

5. **Emotion Comparison:** The newly detected emotion is compared with the previously stored emotion state to determine if a change has occurred.

6. **Conditional Playlist Switch:** If the emotion has changed and the cooldown period permits, the Spotify controller initiates playback of the corresponding playlist, seamlessly transitioning the user's musical experience.

7. **Continuous Monitoring:** The system returns to continuous video display, awaiting the next five-second processing cycle while the Spotify monitor continues its independent two-second polling.

This architecture ensures that the user interface remains responsive while computationally intensive processing occurs asynchronously in the background, maintaining smooth video playback regardless of backend load.

B Technical Specifications

B.1 Input Requirements

- **Webcam:** Minimum resolution 640×480 pixels at 15 fps
- **Lighting:** Adequate frontal illumination for facial feature visibility
- **Internet:** Stable broadband connection for Spotify API communication
- **Spotify Account:** Premium account for playback control capabilities

B.2 Software Dependencies

- **Python:** Version 3.8 or higher
- **OpenCV:** Version 4.5 or higher for computer vision operations
- **PyTorch:** Version 1.9 or higher for neural network inference
- **Spotipy:** Version 2.19 or higher for Spotify API integration

- **FastAPI:** Version 0.68 or higher for OAuth callback server
- **Torchvision:** Version 0.10 or higher for image transformations

B.3 Model Specifications

- **Input Dimensions:** $48 \times 48 \times 1$ (grayscale)
- **Architecture:** Custom CNN with four convolutional blocks
- **Parameters:** Approximately 2.5 million trainable weights
- **Output:** Seven-class probability distribution
- **Inference Time:** 50–150 ms per frame on CPU

B.4 Emotion-Playlist Mapping

The default emotion-to-playlist mapping can be customized by users but follows this general structure:

Table 1: Default Emotion to Playlist Mapping

Detected Emotion	**Playlist Category**
Happy	Upbeat, cheerful, energetic music
Sad	Melancholic, slow tempo, reflective
Angry	Intense, aggressive, high-energy
Fear	Calming, ambient, soothing
Surprise	Dynamic, varied, unpredictable
Disgust	Alternative, unconventional
Neutral	Balanced, moderate tempo

C Deployment Considerations

C.1 Installation Process

Deployment requires several sequential steps:

1. Install Python and required libraries through package manager
2. Configure Spotify Developer account and create application
3. Obtain Client ID and Client Secret from Spotify Dashboard
4. Configure redirect URI to localhost:8888 in Spotify settings
5. Download pre-trained model weights file
6. Configure environment variables for authentication credentials

7. Initialize OAuth authentication flow on first run
8. Grant necessary permissions through Spotify authorization page

C.2 Privacy and Security

The system implements several privacy-preserving measures:

- All video processing occurs on local hardware
- No video frames are transmitted over network
- No persistent storage of facial images
- OAuth tokens stored with file system permissions
- User retains full control over system activation
- Visual indicators show when monitoring is active
- Easy disable mechanism through application interface

C.3 Performance Optimization

Several optimization strategies improve system performance:

- Asynchronous architecture prevents blocking operations
- Frame skipping during high CPU load maintains responsiveness
- Cached playlist URIs reduce API call frequency
- Efficient tensor operations minimize memory allocation
- Haar Cascade integral image optimization accelerates detection
- Batch processing disabled to minimize latency

D Conclusion Remarks

This comprehensive project report documents the development, implementation, and evaluation of a real-time emotion-driven music recommendation system. The integration of facial expression analysis with streaming music services demonstrates practical applications of affective computing in consumer entertainment contexts.

The system successfully balances multiple competing requirements: real-time responsiveness, classification accuracy, computational efficiency, user privacy, and musical continuity. Through careful architectural design and appropriate algorithmic choices, the platform operates effectively on consumer hardware while maintaining user trust through transparent, local processing.

While current limitations exist in classification accuracy, environmental robustness, and platform integration, the proof-of-concept validates the core approach and identifies clear pathways for future enhancement. The findings contribute to broader understanding of how implicit affective signals can complement explicit user input in interactive systems.

As affective computing technologies mature and become more ubiquitous, projects like this highlight both the opportunities and responsibilities inherent in systems that sense and respond to human emotion. Continued attention to technical performance, user experience, ethical deployment, and inclusive design will be essential as emotion-aware applications transition from research prototypes to production services.