AI-Driven Mental Disorder Detection via Facial Expression Analysis

Srishti Dixit*, Devika Salimkumar*, Md Sibbir Hossain*

Department of Computer Science and Engineering, The City College Of New York

*sdixit000@citymail.cuny.edu, *dsalimk000@citymail.cuny.edu, *mhossai026@citymail.cuny.edu

***Abstract*—Mental health issues have become increasingly prevalent in modern society, yet early detection remains a significant challenge due to stigma, inaccessibility, and a shortage of mental health professionals. This paper proposes an AI-driven mental Disorder Detection system that utilizes facial expression analysis as a non-invasive and real-time indicator of potential emotional and psychological conditions. The system architecture is built on a React-based front end integrated with a Flask back end, enabling seamless webcam-based video capture and interaction with deep learning models hosted on Kaggle or Google Colab. The core of the detection system includes advanced deep learning models, EfficientNet-B4, Vision Transformer (ViT) and CNN-LSTM, trained on large-scale emotion recognition datasets such as AffectNet and RAF-DB. These models classify facial expressions into multiple emotional categories, mapping them to patterns often associated with mental health disorders like depression or anxiety. Custom training scripts were employed to address GPU limitations, incorporating periodic checkpointing and robust evaluation mechanisms. Experimental results demonstrate the feasibility of using emotion recognition as a preliminary tool for mental health screening. The proposed platform delivers real-time emotion classification with an interactive user interface and provides a scalable foundation for future clinical integration. While not a substitute for professional diagnosis, this AI-based system offers an innovative step toward accessible and preventive mental health care.**

***Index Terms*—Deep Learning, Emotion Detection, Facial Expression, Mental Health, ConvNeXt, AffectNet**

I. Introduction

Mental health disorders affect millions of individuals globally, impacting their emotional, psychological, and social well-being. Early detection and intervention can significantly improve outcomes, yet traditional diagnostic methods often rely on self-reporting, interviews, or expensive and time-consuming assessments conducted by trained professionals. In recent years, artificial intelligence (AI) has emerged as a transformative tool in the healthcare domain, offering scalable, non-invasive, and real-time solutions.

Facial expressions are a universal form of non-verbal communication and serve as critical indicators of emotional states. Studies suggest that changes in facial expressions can provide insight into underlying mental health conditions such as depression, anxiety, or bipolar disorder. This project leverages the capabilities of deep learning and computer vision to detect such expressions and assess potential signs of emotional disturbances.

Our system aims to develop a robust, accessible, and user-friendly web-based platform that captures real-time facial input and analyzes it using AI-driven models trained on emotion datasets. Integrating a React front end with a Flask back end and deploying state-of-the-art deep learning models provides an innovative tool for preliminary mental disorder detection.

A. Problem Statement

Mental health disorders often go undiagnosed due to social stigma, lack of resources, and limited access to mental health professionals. Traditional diagnosis techniques are time-intensive and not always feasible for early-stage detection. Furthermore, there is no widely adopted automated system that can evaluate emotional well-being through facial expression analysis in real time. The core problem this research addresses is:

How can deep learning techniques be utilized to develop a non-invasive, real-time system that detects signs of potential mental disorders by analyzing facial expressions?

B. Motivation

The motivation behind this project stems from the global mental health crisis, where millions suffer silently due to delayed diagnosis or inadequate access to care. The idea of utilizing facial expression recognition offers a promising avenue, as it requires only a camera and an internet connection—tools available to most individuals. Additionally, the rapid advancement of computer vision, transfer learning, and cloud-based computing platforms such as Kaggle/Google Colab allows for the development of intelligent, lightweight, and accessible AI solutions. This motivated us to design a system that not only detects emotion but also bridges the gap between mental health care and technological accessibility.

C. Contributions

The key contributions of this work are as follows:

- *Design and Implementation of a Real-Time Web-Based Detection System*: Developed using React and Flask to facilitate smooth video capture and server-side processing, enabling end-to-end interaction between the user and the AI backend.
- Secure User Authentication via Firebase: Integrated Firebase Authentication to handle user registration and login

functionality, ensuring a secure and scalable solution for managing access to the system.

- *Integration of Deep Learning Models*: Utilized EfficientNet-B4, Vision Transformer (ViT), and CNN-LSTM architectures for facial emotion classification. These models were trained and validated on multiple benchmark datasets, including AffectNet, RAF-DB, and FER2013, to ensure robustness and generalizability.
- *Custom Training and Evaluation Pipeline*: Designed and executed training workflows on Kaggle using available GPU resources. Implemented custom callbacks for saving model checkpoints every 10 epochs to enhance model reproducibility and version tracking.
- *Deployment-Ready Flask API*: Developed a modular and lightweight Flask API that receives input from the front end, performs real-time inference using trained models, and returns emotion predictions for display.
- *User-Friendly and Scalable Interface*: Delivered a visually appealing and responsive front-end interface that not only facilitates real-time emotion detection but is also adaptable for future enhancements or clinical applications.

D. Paper Organization

The rest of the paper is structured as follows: Section II discusses related work, Section III outlines our methodology, Section IV describes system architecture, Section V presents results and evaluations, and Section VI concludes with future work.

II. Related Work

The intersection of artificial intelligence and mental health diagnosis has garnered significant attention in recent years. Several studies have proposed innovative methods to detect mental disorders using diverse data sources, including social media content, facial expressions, and structured clinical datasets. This section reviews recent advancements relevant to our study, highlighting methodological innovations, datasets, challenges addressed, and how they align or differ from our proposed approach.

AI-Driven Mental Disorders Categorization from Social Media: Published in the 2024 International Conference on Machine Intelligence and Smart Innovation (ICMISI)[1], this study presents a deep learning-based pre-screening framework for detecting mental disorders using Reddit posts. Traditional models such as logistic regression and feedforward neural networks underperformed due to imbalanced data and limited contextual understanding. The authors leveraged transformer-based models like BERT and RoBERTa, achieving up to 91 percent accuracy on benchmark datasets (Kim and Low). Despite strong performance, the study is limited by text-only analysis and subreddit-specific language biases. While our work focuses on facial expression analysis rather than text, this study validates the effectiveness of transformer architectures and inspires future multimodal extensions of our system.

Hybrid Learning Architecture for Mental Disorder Detection: An article published in IEEE Access (Vol. 12, 2024)[2]proposed a hybrid ensemble model combining CNNs, Vision Transformers (ViT), and YOLOv8 for facial emotion recognition. Utilizing AffectNet and FER2013 datasets, the study detected disorders such as depression and anxiety based on emotion recognition. Saliency maps and Grad-CAM were used to enhance explainability. Our work aligns closely with this approach in dataset usage and model types but introduces temporal analysis through a CNN-LSTM hybrid, which provides a deeper understanding of sequential emotional patterns—an area unexplored in this study.

Machine Learning-Based Detection of PTSD and Related Disorders: The 2023 ICESC conference paper [3] offers a systematic review of machine learning models for PTSD detection. It emphasizes the importance of explainability using tools like SHAP, LIME, and heatmaps. This study underscores the need for transparent and trustworthy AI in mental health, which is directly applicable to our model. Although their focus is on text and multimodal data, their review supports our decision to integrate explainability techniques such as Grad-CAM in our facial expression-based system.

Multi-Task Learning for Mental Disorder Detection and Emotion Analysis: Presented at the 2024 ISAS symposium [4], this work introduces a multi-task learning framework that jointly addresses mental disorder detection, sentiment analysis, and emotion recognition using Reddit data. While it improves classification performance through shared feature learning, it is limited to text-based input. Our project currently follows a single-task learning paradigm focused on video-based facial emotion recognition but plans to explore multimodal and multi-task frameworks in future work to boost accuracy and robustness.

CNN-SVM Hybrid for Depression Detection: In the 2024 ICONAT conference [5], a CNN-SVM hybrid model was proposed for diagnosing five types of depression-related disorders. Achieving high accuracy (F1-score ¿ 92 percent), the model shows strong classification performance, especially on clinical datasets. While this approach combines deep learning with classical ML classifiers, our project extends this by employing more advanced architectures such as EfficientNet-B4, ViT, and LSTM for dynamic video input processing. Additionally, our focus includes real-time system deployment and Firebase-based secure authentication—an aspect not covered in this study.

Summary and Positioning: While the reviewed works contribute significantly to the field of AI-assisted mental health diagnostics, most focus either on text analysis or static facial images. Our proposed system bridges this gap by using video-based facial expression analysis and combining it with deep learning models and real-time web architecture. Additionally, Firebase authentication ensures secure and scalable access control, and the usage of datasets like FER2013, AffectNet, and RAF-DB provides a strong foundation for robust emotion recognition

III. Methodology

A. Dataset Descriptions

Our model was trained on a combined dataset constructed from the following sources:

1) FER2013: The FER2013 dataset consists of 35,887 grayscale images sized 48x48 pixels, categorized into 7 emotions: anger, disgust, fear, happiness, sadness, surprise, and neutral. Due to its simplicity and availability, it served as a useful benchmark for initial model validation.

2) RAF-DB: The Real-world Affective Faces Database (RAF-DB) includes approximately 30,000 highly diverse facial images annotated with basic and compound emotion labels. We used the RAF-DB's single-label variant with 7 emotion classes, enabling better generalization to real-world expressions.

3) AffectNet: AffectNet is one of the largest datasets in facial emotion recognition, with more than 1 million facial images manually annotated into 8 emotion categories and valence-arousal dimensions. We used a cleaned, balanced subset due to computational limits but retained class diversity.

B. Data Preprocessing

To ensure consistency across datasets and to optimize model performance, the following preprocessing pipeline was applied:

- **Face Detection:** Using MediaPipe's FaceMesh to crop and align faces.
- **Resizing:** All images were resized to 224x224 pixels to match ConvNeXt input requirements.
- **Color Conversion:** FER2013 grayscale images were converted to 3-channel RGB format.
- **Normalization:** Pixel values were normalized using ImageNet statistics.
- **Augmentation:** Applied random horizontal flips, color jitter, slight rotation, and Mixup augmentation during training to mitigate overfitting.

C. Model Architectures

We experimented with several deep learning architectures before finalizing ConvNeXt-Large.

1) EfficientNet-B4: Used as an early baseline. While EfficientNet-B4 offered parameter efficiency, it failed to exceed 60% validation accuracy on combined datasets.

2) Vision Transformer (ViT): ViT was tested due to its strong global attention capabilities. However, it required longer convergence and high batch size to outperform CNNs, making it less feasible under GPU constraints.

3) CNN-LSTM: Explored for future temporal emotion modeling from video frames. While not used in final deployment, its addition marks the direction of future work in dynamic emotion analysis.

4) ConvNeXt-Large: ConvNeXt-Large was selected as the final architecture due to its powerful convolutional backbone, enhanced feature extraction, and state-of-the-art performance in image classification. Pretrained weights from ImageNet-22k were fine-tuned on our datasets.

D. Training Configuration

The model was trained using PyTorch on a Kaggle Pro GPU (NVIDIA T4), and later on Google Colab Pro+ for extended epochs. Below is the configuration:

- **Epochs:** 70
- **Batch Size:** 64
- **Optimizer:** AdamW with weight decay
- **Learning Rate Scheduler:** Cosine Annealing with warm restarts
- **Loss Function:** Cross-Entropy with weighted loss to counter class imbalance
- **Regularization:** Mixup (alpha=0.2), Label smoothing
- **Stabilization:** Stochastic Weight Averaging (SWA)
- **Evaluation:** Test Time Augmentation (TTA) with 5 inference paths per image

Model checkpoints were saved every 10 epochs using custom callback functions to allow recovery and version comparison.

IV. System Architecture

The architecture of the proposed AI-Driven Mental Disorder Detection system is designed to facilitate real-time facial emotion recognition using deep learning models trained on large datasets. It addresses the limitations of local hardware by integrating cloud-based model inference, while also ensuring usability, security, and scalability. The architecture comprises three primary components: (i) a React-based frontend for user interaction, (ii) a Flask-based backend for API and file handling, and (iii) a cloud-based inference engine hosted on Google Colab (or Kaggle) to run GPU-dependent models.

A. Overview of Workflow

The complete system workflow is as follows:

- The user accesses the React frontend and records a short video using their webcam.
- The recorded video is sent to the Flask backend as a binary blob.
- The backend stores the video temporarily and uploads it to a pre-authorized Google Drive folder via the Google Drive API.
- A cloud-based inference script (hosted in Google Colab) accesses the video from Drive, runs the deep learning model on the video frames, and writes the results back to Google Drive as result.json.
- The backend periodically checks or retrieves this result and returns it to the frontend.
- The frontend displays the predicted emotional state to the user in a clean UI.

This architecture optimizes GPU usage, simplifies the frontend-backend communication, and provides a practical approach to remote model inference in resource-constrained environments.

B. Frontend Layer: React Application

The frontend is developed using React.js, offering an interactive interface for users to record a video, trigger prediction, and view results.

- Webcam Capture: Uses navigator.mediaDevices.getUserMedia() to access the user's webcam and capture a short video (typically 5–10 seconds).
- Video Encoding: The video is encoded as .webm, converted to a Blob object, and sent as a POST request to the backend.
- Authentication (Firebase): Users must sign in using Firebase Authentication, which handles secure registration and login using email/password or OAuth providers.
- Result Display: The UI updates with the detected emotion and a brief message once predictions are fetched from the backend.

This layer ensures accessibility across devices, encourages user engagement, and provides authentication using Firebase, enabling secure access control for future healthcare-grade applications.

C. Backend Layer: Flask + Google Drive API

The backend serves as the communication bridge between the frontend and the cloud-based model. Built with Flask, it is lightweight, extensible, and integrates multiple services.
Key Modules:

- app.py: The main entry point of the Flask server. Handles HTTP requests for uploading videos, triggering model processing, and fetching results.
- send_to_drive.py: Uses the Google Drive API and a service_account.json key to upload the video to a specific Drive folder.
- fetch_result.py: Periodically checks the same Drive folder for a result.json file and parses the prediction output.

Authentication:

- Service Account: A Google Cloud service account provides programmatic access to Google Drive. This JSON key contains credentials and permissions for secure API interaction.
- Temporary Storage: Uploaded files are stored in the uploads/ directory before being moved to the cloud.
- Input video: input_video.webm
- Output: result.json (contains predicted class, probability scores, timestamps)

The backend also includes error handling, timestamping, and file cleanup mechanisms to manage multiple requests and maintain performance.

D. Cloud-Based Inference Layer: Google Colab

Due to GPU limitations on local machines, model inference is performed on Google Colab, which supports free GPU access and seamless Drive integration.
Process:

- Drive Mounting: Colab notebook mounts Google Drive using from google.colab import drive.
- Frame Extraction: The uploaded .webm video is decoded using libraries such as OpenCV or FFmpeg. Frames are sampled at a predefined rate (e.g., 1 FPS or 5 FPS).
- Preprocessing: Frames are resized and normalized to match the input shape expected by the model (e.g., 224×224 for EfficientNet).
- Model Prediction: The pre-trained model (e.g., EfficientNet-B4, ViT, CNN-LSTM) is loaded and used to predict emotions on each frame.
- Aggregation: Predictions across frames are averaged or fed into an LSTM for temporal analysis to determine the final result.
- Output Writing: The final result is saved as a JSON file (result.json) back into the same Google Drive folder.

This layer is manually triggered but can be automated in future using scheduled APIs or Drive file listeners.

E. Datasets and Model Integration

To ensure robustness and generalization, the models are trained on a combination of three publicly available datasets:

1) FER2013: Contains 35,000 grayscale images classified into seven emotions.
2) AffectNet: A large-scale dataset with 1 million images annotated for eight basic emotions.
3) RAF-DB: Real-world dataset with high-resolution facial images and fine-grained emotion labels.

The training is performed on Kaggle using GPU runtime and model checkpoints are saved periodically. The final .h5 model files and architecture (.json) are stored in the backend or loaded from Drive.

Models Tried:

1) EfficientNet-B4: Offers high accuracy with fewer parameters.
2) Vision Transformer (ViT): Captures global context through self-attention mechanisms.
3) CNN-LSTM: Enables temporal modeling of video sequences.

F. Security and Authentication

1) Firebase Authentication is integrated into the frontend to manage user access.
2) Google Drive API access is limited to the service account credentials with specific permissions.
3) Data Privacy: No user data is stored permanently. All video files are automatically deleted after inference is completed.
4) CORS: Proper CORS headers are configured in Flask to enable secure cross-origin requests.

G. Advantages of the Proposed Architecture

V. Experimental Results & Evaluation

A. Evaluation Metrics

The model was evaluated on the combined test set using the following metrics:

Component	Role	Reason for Choice
React Frontend	User interface, webcam access	Lightweight, responsive, and has a strong ecosystem for building real-time UI with webcam and video processing support.
Flask Backend	Video upload, API endpoints	A simple yet powerful Python-based framework that integrates well with machine learning pipelines and allows for rapid prototyping.
Google Drive	File exchange medium	Serves as a cloud bridge between local backend and cloud notebook, removing the need for dedicated server-based storage.
Google Colab/Kaggle	Model execution	Provides free GPU access for running deep learning models with a flexible Python environment and seamless integration with Google Drive.
Firebase	Authentication	Offers secure, scalable user login and access management, which can be easily integrated into web applications.

TABLE I
ADVANTAGES OF THE PROPOSED ARCHITECTURE

- **Accuracy:** Overall correct predictions / total predictions.
- **Precision, Recall:** Per-class metrics computed for imbalance sensitivity.
- **F1-Score (Macro):** Average F1 across all classes.
- **Confusion Matrix:** Visual representation of class-level performance.

B. Model Performance Comparison

We compared four architectures across validation datasets:

Model	Dataset	Training Acc.	Validation Acc.
EfficientNet-B0	FER2013	65.52%	60.00%
EfficientNet-B4	AffectNet	52.55%	54.76%
ConvNeXt Small	AffectNet	66.96%	58.75%
ConvNeXt Base	AffectNet	73.39%	60.25%
ConvNeXt Large	AffectNet	73.36%	60.25%
ConvNeXt Large	RAF-DB	99.10%	85.82%
CNN-LSTM (static)	FER2013	62.50%	57.25%
ConvNeXt-Large	Combined	**89.65%**	**81.74%**

TABLE II
MODEL COMPARISON ACROSS DATASETS

ConvNeXt-Large outperformed all baselines in both accuracy and F1-score. The inclusion of TTA and SWA helped stabilize performance in minority classes like fear, disgust, and sadness.

C. Confusion Matrix

[width=0.48]convnext_large_epoch63_confusion_matrix63.png

Fig. 1. Confusion Matrix – ConvNeXt-Large on Combined Validation Set

Most confusion occurred between "neutral" and "sad" classes, highlighting the need for more samples and perhaps multi-label classification in future iterations.

D. Ablation Study

We conducted an ablation study to isolate the effect of:

- Removing SWA $\rightarrow$ performance dropped by 1.3%.
- Disabling Mixup $\rightarrow$ increased overfitting, especially in early epochs.
- Skipping TTA $\rightarrow$ test-time predictions fluctuated more across runs.

E. Discussion

These experiments confirm that emotion classification models benefit significantly from augmentation and stability strategies. ConvNeXt-Large, in particular, proved robust to dataset noise and imbalance. The model generalizes well across three datasets, suggesting potential for deployment in real-world scenarios.

VI. CONCLUSION AND FUTURE WORK

A. Summary of Findings

This study demonstrates that deep learning models—particularly ConvNeXt-Large—can be effectively used to perform real-time emotion classification using facial expressions. By leveraging benchmark datasets (FER2013, RAF-DB, AffectNet) and combining them with modern training strategies like Mixup, SWA, and TTA, we achieved over 80% validation accuracy. The React-Flask-Firebase architecture facilitated seamless deployment and interaction.

B. Limitations and Future Enhancements

Despite promising results, several limitations remain:

- Manual Execution of Inference: Currently, the user must manually trigger model inference on Google Colab. In the future, this will be automated using Google Cloud Functions or Drive event listeners.
- Latency: Uploading videos and waiting for Colab results may introduce a delay. Real-time inference using a cloud-hosted API (e.g., GCP or AWS Lambda) is a potential upgrade.
- Data Storage: Although data is not stored permanently, a secure logging mechanism may be introduced for clinical use.
- Multi-Modal Expansion: Future iterations will integrate text (e.g., from chatbots) and voice to enhance prediction using multimodal learning.
- Lack of demographic balance in datasets may affect fairness.
- Current model does not yet support temporal emotion tracking or multi-label prediction.

C. Future Directions

- Integrate text and voice sentiment analysis for multimodal emotion detection.
- Explore CNN-LSTM or ConvLSTM to model facial dynamics over time.
- Implement edge-device deployment via ONNX/TensorRT.
- Add explainability tools (e.g., Grad-CAM, LIME) for clinical transparency.
- Collaborate with mental health professionals to validate system reliability in real environments.

This system offers a scalable foundation for preventive mental health screening, bridging technology and psychology through ethical and accessible AI.

ACKNOWLEDGMENT

We would like to sincerely thank our faculty mentor, **Professor Ihab Darwish** <ihab.darwish60@login.cuny.edu>, for his invaluable guidance throughout the semester. We also acknowledge the support provided by The City College of New York, our department faculty, and compute resources via Kaggle and Google Colab Pro. Tools such as GitHub, Firebase, and Overleaf contributed significantly to our development and collaboration workflows.

The source code and training pipeline are available at: https://github.com/Capstone-Project-CCNY/AI-Driven-Mental-Disorder-Detection-

REFERENCES

[1] Z. Liu, H. Mao, C.-Y. Wu, et al., "A ConvNet for the 2020s," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022, pp. 11976–11986.

[2] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 6105–6114.

[3] A. Mollahosseini, D. Chan, and M. H. Mahoor, "AffectNet: A database for facial expression, valence, and arousal computing in the wild," *IEEE Trans. Affective Comput.*, vol. 10, no. 1, pp. 18–31, 2019.

[4] S. Li and W. Deng, "Reliable crowdsourcing and deep locality-preserving learning for expression recognition in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 2584–2593.

[5] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond Empirical Risk Minimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.

[6] P. Izmailov et al., "Averaging weights leads to wider optima and better generalization," in *Proc. Conf. Uncertainty Artif. Intell. (UAI)*, 2018.

[7] D. Wang et al., "Test-Time Training with Self-Supervision for Generalization under Distribution Shifts," in *Proc. NeurIPS*, 2020.

[8] A. Ahmed et al., "AI-Driven Mental Disorders Categorization from Social Media: A Deep Learning Pre-Screening Framework," in *Proc. Int. Conf. Mach. Intell. Smart Innov. (ICMISI)*, 2024, doi: 10.1109/ICMISI61517.2024.10580665.

[9] B. Singh et al., "A Hybrid Learning Architecture for Mental Disorder Detection Using Emotion Recognition," *IEEE Access*, vol. 12, pp. 91410–91425, 2024.

[10] S. Ramesh et al., "Machine Learning-Based Detection of PTSD in Mental Health," in *Proc. Int. Conf. Electron. Sustain. Commun. Syst. (ICESC)*, 2023.

[11] M. Kaya et al., "Multi-task Learning on Mental Disorder Detection Using Social Media Posts," in *Proc. Int. Symp. Innov. Approaches Smart Technol. (ISAS)*, 2024.

[12] V. Patel et al., "Improving Mental Health Assessments: CNN-SVM for Depression Detection," in *Proc. Int. Conf. Adv. Technol. (ICONAT)*, 2024.

APPENDIX A
CODE SNIPPETS

A. Model Loading and Inference (Flask API)

```
import torch
from torchvision import transforms
from model import ConvNeXtLarge

model = ConvNeXtLarge(pretrained=False)
model.load_state_dict(torch.load("model_final.pt",
model.eval()

def predict(image_tensor):
    with torch.no_grad():
        outputs = model(image_tensor.unsqueeze(0))
        _, predicted = torch.max(outputs, 1)
        return predicted.item()
```

APPENDIX B
DATASET SAMPLES / EXTRA FIGURES

Include images and annotated samples from each dataset with emotion labels.