INTERNSHIP I / DISSERTATION I REPORT

ON

REAL TIME MULTIMODAL ENGAGEMENT DETECTION IN DIVERSE LEARNING

Submitted in partial fulfilment of the requirements for the degree of

M.Tech. Computer Science and Engineering

by

Tanmay Borse

24MCS0001

Under the Supervision of

Dr. Deepa D

Assistant Professor Senior Grade 1

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

November, 2025

DECLARATION

I, **Tanmay Borse (24MCS0001)** hereby declare that the thesis entitled **"Real Time Multimodal Engagement Detection in Diverse Learning"** submitted to Vellore Institute of Technology (VIT), Vellore for the award of the degree of **Master of Technology in Computer Science and Engineering** is a record of bonafide work carried out by me under the supervision of **Dr. Deepa D**, School of Computer Science and Technology, Vellore Institute of Technology, Vellore.

I further declare that the work reported in this project report has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Vellore

Date: 18-11-2025

Signature of the Candidate

CERTIFICATE

This is to certify that the thesis entitled **"Real Time Multimodal Engagement Detection in Diverse Learning"** submitted by **Tanmay Borse (24MCS0001)**, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore for the award of the degree of **Master of Technology in Computer Science and Engineering**, is a record of bonafide work carried out by him under my supervision during the period, **14-07-2025** to **11-11-2025** as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other Institute or University. The dissertation fulfills the requirements and regulations of the Institute and in my opinion meets the necessary standards for submission.

Place: Vellore,

Date: 18-11-2025

Name and signature of the guide

Internal Examiner

External Examiner

Head of the Department

Department of Quantum AI

ABSTRACT

The accurate and timely assessment of engagement is necessity of affective online session. User concentration a multifaceted construct encompassing behavioural, emotional and cognitive involvement, is a string predictor of success and retention. Our system utilizes a Convolutional Neural Network architecture to process facial features extracted from video streams. Engagement is quantified using a multimodal measure that fuses the Eye Aspect Ratio (EAR) and the Mouth Aspect Ratio (MAR). EAR provides a robust, non-intrusive indicator of attentiveness, while MAR offers insights into active participation and cognitive load. The CNN model is trained on a Facial Emotion Recognition (FER 2013) dataset and sample images captured through webcams. At end of session one report will be get generated that will tell how much percentage of time end user is attentive and out of focus. With EAR and MAR one buffering time window is introduced before the starting of non-attentive timer. For a dim light environment sitting candidate alarm window is created to notify them to sit in light source environment. Experimental results demonstrate that this fusion approach of dataset with new web images significantly improves the accuracy of model up to 85% from 44% and reliability of engagement detection compared to unimodal methods, enabling timely and targeted pedagogical interventions to optimize the learning experience.

Keywords: *Online attentiveness, Emotion detection, Distraction detection, Drowsiness detection.*

ACKNOWLEDGEMENTS

With immense pleasure and a deep sense of gratitude, I wish to express my sincere thanks to my supervisor **Deepa D**, School of Computer Science and Engineering, Vellore Institute of Technology (VIT), Vellore without her motivation and continuous encouragement, this research would not have been successfully completed.

I am grateful to the Chancellor of VIT, **Dr. G. Viswanathan**, the Vice Presidents, and the Vice Chancellor for motivating me to carry out research in the Vellore Institute of Technology.

It would be no exaggeration to say that the Dean in-charge of SCOPE, **Prof. Jaisankar N**, was always available to clarify any queries and clear the doubts I had during the course of my project. I would also like to acknowledge the role of the HoD, **Dr. Sathiya Kumar C**, who was instrumental in keeping me updated with all necessary formalities and posting all the required formats and document templates through the mail, which I was glad to have had.

Finally, I would like to thank **Vellore Institute of Technology** for providing me with infrastructural facilities, a flexible choice and for supporting my research and execution related to the dissertation work.

Place: Vellore, **Tanmay Borse (24MCS0001)**

Date: 18-11-2025

CONTENTS

LIST OF FIGURES

LIST OF TABLES

List of Abbreviations

CNN Convolutional Neural Network

EAR Eye Aspect Ratio

MAR Mouth Aspect Ratio

Chapter 1

INTRODUCTION

1.1 Overview

This paper presents a novel approach for Real-Time Multimodal Engagement Detection in diverse learning environments, addressing the critical need for automated tools to monitor user focus and participation. Traditional methods for monitoring engagement such as direct observation or self-report are often subjective, resource intensive and difficult to scale. The rise of multimodal data collection and computer vision techniques offers a powerful solution to automate this process (Voulodimos et al. (2018), Szeliski (2022)). Extended hours in front of a screen during remote meetings can severely impact user's focus. The relaxed home atmosphere often encourages learners to multitask, pulling their attention away from the important content with activities such as checking social media, using their phones, or listening to audio. Since these distractions result in poorer academic outcomes, evaluating users' attentiveness is essential for presenter to effectively manage and monitor progress within the virtual classroom or online corporate meetings.

1.2 Objectives

- Use facial analysis to detect the dominant emotion (happy, sad, drowsiness, distraction and neutral)
- Build a system that can monitor attendees live and display alertness/emotion feedback dynamically
- Instant feedback to Presenter to find which candidate is Non attentive in session
- Feedback to user at end of session to know how much time they were attentive in session.

Chapter 2

LITERATURE REVIEW

If we talk about previous works done in this field automated systems are developed with features and neural networks to detect active behavioural of attendees of meeting, their facial expression that based on eyes and mouth moment, head positioning have become key elements for analysing distraction signs, sleepiness, and varied emotions as mention by A-masiri and Kerdvibulvech (2023) and Azeez et al. (2023). This part will review previous works on these three signs of human distraction which are crucial features applied for the study of this experiment.

2.1 Emotion detection

In emotion-based assessment, According to Sharma et al. (2022) He utilized the state-of-the-art Mini-Xception CNN which is also used by Fatima et al. (2021), trained on the grayscale FER-2013 dataset, to detect seven distinct emotion types (e.g., angry, happy, sad, surprise). The model achieved 66% accuracy in emotion classification on the FER-2013 dataset. The study then used the resulting emotion probabilities to calculate a concentration index by multiplying the probability of the dominant feeling by a corresponding emotional weight. This index serves to classify the user's engagement into three tiers: very engaged, nominally engaged, and not engaged at all. The research by Shamika et al. (2021) compared a Convolutional Neural Network (CNN) against a Long Short-Term Memory (LSTM) model to gauge user involvement through emotion detection. Using the DAiSEE dataset—which captures various feelings during e-learning—the models were trained to categorize four levels of emotional affect into three engagement labels: attentive, partially attentive, and inattentive. The study give result that CNN model perform well than LSTM model.

2.2 Distraction detection

Distraction detection is a technique of determining behaviours denoting how attentive a candidate in a meeting. The Sharma et al. (2022) utilized a two-stage approach: Haar Cascades (Viola

and Jones (2001)) were used for initial facial and ocular detection, and the output from this detection was subsequently processed by a dedicated Convolutional Neural Network (CNN) to categorize the user's behavioural state into one of two classes: focused or distracted. Khan and Debnath (2020) proposes a distraction detection system that utilizes head pose (measured by roll, pitch, and yaw Euler angles) and eye direction (determined via binary thresholding) based on Dlib facial landmarks. This approach is effective for video analysis and extends to recognizing drowsiness in multiple learners.

2.3 Sleepiness detection

While talking about sleepiness detection in previous studies the movement of the eyes and lips provides key data for drowsiness detection, often quantified using the Eye Aspect Ratio (EAR) (Mehta et al. (2019)) and Yawn Aspect Ratio (YAR) (Shah et al. (2021)). While Shah et al. (2021) uses these ratios primarily to filter frames where the eyes are closed or a yawn occurs, Khan and Debnath (2020) advances this by applying a statistical threshold to monitor the duration of drowsiness. Recognizing the limitations of relying solely on eye closure for early detection, Ghoddoosian et al. (2019) introduced a more robust method using Hierarchical Multiscale Long Short-Term Memory (HM-LSTM) to achieve higher-than-human accuracy in detecting drowsiness across all stages, accompanied by the release of the Real-Life Drowsiness Dataset (RLDD). Subsequently, Gapi et al. (2021) and Khandare et al. (2023) utilized this RLDD dataset to train VGG16-based CNN models designed to predict facial fatigue through the detection of indicators like hanging eyelids and drooping mouth corners.

Chapter 3

PROBLEM DESCRIPTION

3.1 Problem Description

Student participation is a critical factor and a strong predictor of academic success, but its reliable and scalable assessment in modern educational settings presents a significant challenge. Traditional methods—like manual observation—are subjective and resource-intensive and do not capture the subtle shifts from moment-to-moment in attention necessary for timely pedagogical intervention. The core problem addressed by this project is the lack of a robust, real-time, and generalized system capable of accurately monitoring learner attentiveness and participation across diverse, unconstrained environments such as varied lighting, differing camera angles, yawning and attentiveness issues.

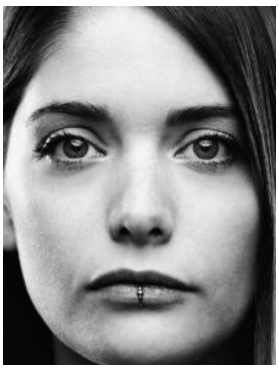
Learners attending online classes often spend long hours in comfortable home environments, which facilitates multitasking, for example, mobile phone use, social media, and significantly reduces focus on study content, leading to lower academic performance. Assessing this subtle lack of focus requires continuous objective tracking. Unimodal Limitations for systems are relying on single cues that are often unreliable. Detecting early-stage drowsiness, for instance, requires a more advanced temporal analysis than simple visual feature checks. Furthermore, these systems often fail to distinguish between active participation (speaking) and passive attention. However, state-of-the-art CNNs can classify basic emotions, limiting the model's ability to accurately infer true engagement.

To solve this, we require a multimodal framework that integrates highly efficient physiological features with deep learning. Specifically, the challenge is to design a system that can effectively fuse the low-latency indicators of Eye Aspect Ratio (EAR) and Mouth Aspect Ratio (MAR) with the rich, contextual information provided by a CNN-based emotion classifier to generate an accurate and actionable index of learner engagement in real-time.s

3.1.1 Dataset Description

The original FER2013 dataset consists of approximately 35,887 grayscale images of 48×48 pixels, categorized into seven emotions: Angry, Disgust, Fear, Happy, Sad, Surprise, and Neutral. Initially, some emotion classes within the FER2013 dataset lacked a sufficient number of images for optimal model training. To enhance the model's performance and robustness in unconstrained real-world conditions (such as varying lighting and head poses), we augmented the dataset. This involved collecting and pre-processing additional images captured specifically from typical webcam setups. These new samples were integrated into the FER2013 classes that had fewer original images, effectively increasing the sample size for these categories.

(a) FER2013 Image



(b) Sample Image

Figure 3.1: FER2013 and Added sample Neutral Expression

Chapter 4

PROPOSED APPROACH

This work details a system engineered to evaluate an individual's attentiveness or non-attentiveness during online sessions. The system mirrors a presenter's ability to gauge participant interest in an in-person setting by analyzing facial expressions, yawning, and eye movements. It operates by capturing video frames from a webcam and subjecting them to initial steps like normalization and feature engineering. It also assesses the environmental lighting to optimize frame clarity. A facial detection model first identifies the face within the captured frame. Next, the system uses the Dlib library to extract 68 facial landmarks. These landmarks are used to compute the Eye Aspect Ratio and Mouth Aspect Ratio to detect sleepiness and yawning. A trained Convolutional Neural Network (CNN) then processes these features to ultimately predict the user's state of attentiveness and also determine the surrounding light environment.

4.1 Data pre-processing and Augmentation

4.1.1 Data pre-processing and Normalization

Before training the CNN model using the combined FER2013 and webcam images, the data required pre-processing. Any additional coloured images captured from webcams were first converted to grayscale to maintain consistency with the FER2013 dataset, which also helped reduce computational load. The images then underwent a normalization step where all pixel values, originally in the range [0, 255], were scaled to the range [0, 1] by dividing by 255.0; this standardization was performed to accelerate convergence during the training phase. Finally, every image was resized to a fixed input dimension of 48 x 48 pixels to match the requirements of the selected CNN architecture.

4.1.2 Data Augmentation

To prevent overfitting and substantially enhance the model's ability to generalize to real-time video feeds, we utilized data augmentation. This crucial step involved applying various geometric

and pixel-level transformations to the training images. For instance, we used rotation by various angles to account for user head movement, and adjusted brightness and contrast to simulate a diverse range of lighting conditions. Additionally, techniques like zoom and shearing were applied to mimic minor camera movements or changes in distance from the user.

4.2 Model

This section details the specific architecture, configuration, and training regimen of the Convolutional Neural Network (CNN) used for the initial feature extraction and emotion classification component of the multimodal engagement detection system.

4.2.1 CNN Model Architecture

The model utilizes a sequential architecture comprising four convolutional blocks, followed by fully connected layers. The design emphasizes depth and the inclusion of regularization techniques to ensure robust feature learning and prevent overfitting. They work sequentially to transform the raw pixel data of the input image (48×48×1) into high-level, abstract features that the final dense layers can use for classification.

The network employs a standard pattern of Convolution → Batch Normalization → ReLU Activation → Pooling → Dropout.

Block	Input Shape	Filters	Output Operation	Final Output Shape
Initial	(48,48,1)	32	Conv → Norm → ReLU	(46,46,32)
Block 1	(46,46,32)	64	Conv → Norm → ReLU → Pool → Drop	(22,22,64)
Block 2	(22,22,64)	128	Conv → Norm → ReLU → Pool → Drop	(10,10,128)
Block 3	(10,10,128)	256	Conv → Norm → ReLU → Pool → Drop	(4,4,256)

Figure 4.1: CNN layer description

The Convolutional (Conv2D) layers initiate feature extraction by applying a learned set of filters across the input image. These filters progressively increase in number (from 32 to 256), enabling deeper layers to recognize increasingly complex patterns. Batch Normalization stabilizes the learning process by standardizing layer activations. The ReLU (Rectified Linear Unit) function then introduces essential non-linearity, which allows the network to model the complex, non-linear relationships inherent in facial expressions, something linear models cannot achieve. For efficiency, a 2×2 Max Pooling layer reduces both the computation required and the overall parameter count. Finally, 25% Dropout acts as a powerful regularize, randomly silencing units during training to force the network to develop a distributed and more robust feature representation, thereby effectively combating overfitting.

4.2.2 Model compilation and Training configuration

The *Adam* optimizer was chosen for its adaptive learning rate functionality, and it was initially set to a cautious value of 0.0001. This slow rate was deliberate, aiming to allow the deep network to fine-tune its weights and prevent the *Categorical Cross-Entropy* loss (appropriate for the seven-class emotion classification) from experiencing large, destabilizing oscillations. Training was conducted using an augmented and resampled dataset ($\mathbf{X}_{\mathbf{resampled}}$, $\mathbf{Y}_{\mathbf{resampled}}$) to correct the class imbalance inherent in the dataset. A data generator was utilized to efficiently stream data and apply real-time augmentation during training. Model performance was consistently assessed using a separate, static validation set ($\{X_{\text{val}}$, $\{Y_{\text{val}}$). Furthermore, a *ModelCheckpoint* callback was implemented to save the model weights to `best_model_v2.h5` only when an improvement in validation accuracy was observed.

4.3 Facial Landmark Extraction and Aspect Ratio Calculation

4.3.1 Dlib for facial landmarks

For real-time engagement detection, the core features Eye Aspect Ratio (EAR) and Mouth Aspect Ratio (MAR)—are calculated directly from video frames using facial landmarks, which is significantly faster than running the full image through the CNN for every frame. The Dlib library, along with its pre-trained 68-point shape predictor, is employed for real-time face and landmark detection. Specifically, the Dlib Histogram of Oriented Gradients (HOG)-based face detector first identifies the face's bounding box in each video frame. Then, the 68-point predictor is applied to the detected face area, mapping specific points to key facial features, including the eyes and mouth, from which the EAR and MAR are computed.

4.3.2 Sleepiness detection

To detect the sleepiness of user capture through frames we are using the Eye Aspect Ratio (EAR). It is a scalar metric used to determine state of an eye either close or open. It is calculated using the distances between specific sets of six landmarks around each eye by taking ratio between vertical distances to horizontal distances.

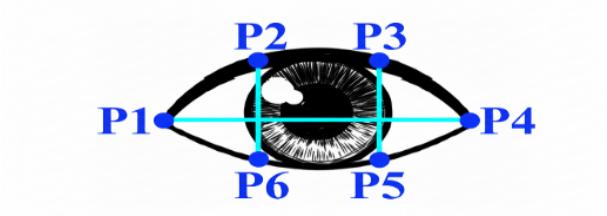


Figure 4.2: Eye Aspect Ratio

For a set of eye landmarks P_1, P_2, P_3, P_4, P_5 and P_6. Where P_1 and P_4 are horizontal extremes, and P_2, P_3, P_5, P_6 are vertical as shown in fig 4.2, the EAR is defined as:

$$\text{EAR} = \frac{\|P_2 - P_6\| + \|P_3 - P_5\|}{2 \times \|P_1 - P_4\|}$$

The Eye Aspect Ratio (EAR) is used as an indicator of an attendee's attentiveness. As seen in the figure (which describes the EAR calculation), when the eye is closed, the vertical distances between the landmarks (P_2 and P_6, and P_3 and P_5) decrease. This results in a lower numerator compared to the denominator in the EAR formula, yielding a low EAR value. A low EAR thus indicates a closed eye, which is interpreted as non-attentiveness or drowsiness. Conversely, a high EAR signifies an open eye and suggests attentiveness. To make a final decision, the EAR is calculated for both eyes, and the average of the two values is compared against a defined threshold, typically set in the range of 0.2 to 0.3, which can vary slightly depending on individual eye size.

4.3.3 Drowsiness Detection

Mouth Aspect Ratio is used to detect the drowsiness of a user by calculating its MAR value. It identifies the opening of the mouth, serving as an indicator of activeness of an end user, either he is speaking or yawning. MAR is given by the equation:

$$\text{MAR} = \frac{\|P_2 - P_4\| + \|P_6 - P_8\|}{\|P_1 - P_3\| + \|P_5 - P_7\|}$$

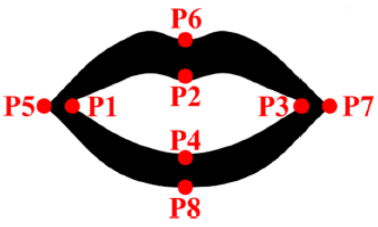


Figure 4.3: Mouth Aspect Ratio

A high MAR value indicates the mouth is significantly open (e.g., speaking or yawning), the value comes high if P_6 and P_8, and P_2 and P_4 values are large as compared to P_1, P_3 and P_5, P_7. If distance in vertical points is high resulting high value in numerator. And hence horizontal points comes close then giving low value at denominator. The result in compared with Threshold value. This situation indicating yawning providing a valuable cue for active or fatigued states.

These two real-time features, EAR and MAR, are then fused with the contextual emotional features extracted by the trained CNN to create the final multimodal engagement prediction model.

Chapter 5

EXPERIMENTAL SETUP

The system operates in a continuous, real-time loop for multimodal engagement detection. Initially, the webcam feed is captured frame-by-frame. Before processing a frame, we will check the user weather user is seated in a light environment to video capture can generate clear images, if user is in low light environment then create a alter window on screen to request the user to seat in better light environment. Then each frame is simultaneously processed by two parallel pipelines. In the Computer Vision Pipeline, the Dlib library is used to detect the user's face and locate 68 facial landmarks. These landmarks are immediately used to calculate the Eye Aspect Ratio (EAR) and the Mouth Aspect Ratio (MAR) using geometric formulas, providing instantaneous, low-latency indicators of attentiveness and active participation or drowsiness.

Simultaneously, the Emotion Recognition Pipeline extracts the detected face region, converts it to grayscale, normalizes, and resizes it to 48×48 pixels. This processed image is then fed into the pre-trained CNN model, which outputs a probability distribution across seven emotion classes. Finally, the system executes the Fusion and Decision Module, where the real-time EAR and MAR values are combined with the CNN's emotion probabilities. This integration allows for a robust, multimodal assessment: for example, low EAR combined with a 'Sad' emotion probability may indicate disengagement or confusion, while a moderate EAR and high MAR (indicating speech) suggests high active engagement.

The experimental setup is designed to validate the performance of the proposed multimodal system in two stages: first, the accuracy of the baseline CNN model (emotion classification), and second, the efficacy of the complete real-time EAR/MAR and emotion fusion system.

5.1 Dataset Configuration

5.1.1 Emotion Classification Dataset (CNN Training & Testing)

Primary Data FER-2013 Dataset (Facial Expression Recognition): Content 48×48 grayscale images of faces labeled into 7 emotion categories. Supplementary Data Webcam Images: Content Additional images collected and manually labeled to improve generalization to real-world, diverse lighting and pose variations typical of a learning environment.

The combined dataset is split into 80% Training, 10% Validation, and 10% Testing sets. Resampling is applied to the training set to mitigate class imbalance. Real-time data augmentation (rotation, shifting, flipping, contrast adjustment) is applied only to the training data via the Keras 'ImageDataGenerator'.

5.1.2 Multimodal Testing Dataset (Engagement Ground Truth)

A custom-recorded dataset comprising short video clips of student surrogates performing typical online learning behaviors. Videos are manually annotated frame-by-frame with a Ground Truth Engagement Label (e.g., Engaged, Disengaged).

Annotations are based on observed behaviors: looking at screen (Engaged), looking away/using phone (Disengaged), yawning/drowsy signs (Disengaged), or actively speaking (Engaged).

5.2 Hardware and Software Environment

Table 5.1: Technology Stack

Category	Component/Tool	Details
Hardware	GPU	NVIDIA CUDA-enabled GPU for accelerated training and real-time processing.
Frameworks	Deep Learning	TensorFlow and Keras for CNN model definition and training.
Computer Vision	Facial Landmarks	Dlib Library (using the 68-point shape predictor).
Libraries	Processing	OpenCV for real-time video stream handling and image pre-processing.
Optimizer	Training	Adam Optimizer with Learning Rate: 0.0001.

5.3 Training and Evaluation Protocol

5.3.1 CNN Training Protocol

Model The defined 4-block CNN architecture (Conv2D $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ Pool $\rightarrow$ Dropout). Loss Function Categorical Cross-Entropy Epochs 200 (with an emphasis on early

stopping). Callback ModelCheckpoint to save the model based on the highest achieved validation accuracy.

5.3.2 Evaluation Metrics for Multimodal System

The real-time system is evaluated against the manually annotated Multimodal Testing Dataset. Prediction: The final output (EAR + MAR + Emotion Fusion) is classified into two engagement states. **Metric**: Accuracy in classifying the three ground truth engagement labels. **Comparison**: A crucial part of the experiment is comparing the final multimodal system's accuracy against unimodal baselines (e.g., using EAR only or Emotion only to demonstrate the superior performance gained through fusion.

This module processes the combined features to classify the user's state as either attentive or non-attentive. If the user is classified as attentive, their accumulated attentiveness time is increased. If classified as non-attentive, a 15-second buffer time begins. If the user remains non-attentive after this buffer, the non-attentive timer starts incrementing. At the conclusion of the session, the system generates a report detailing the percentage of total time the user was actively concentrating versus the percentage of time they were not.

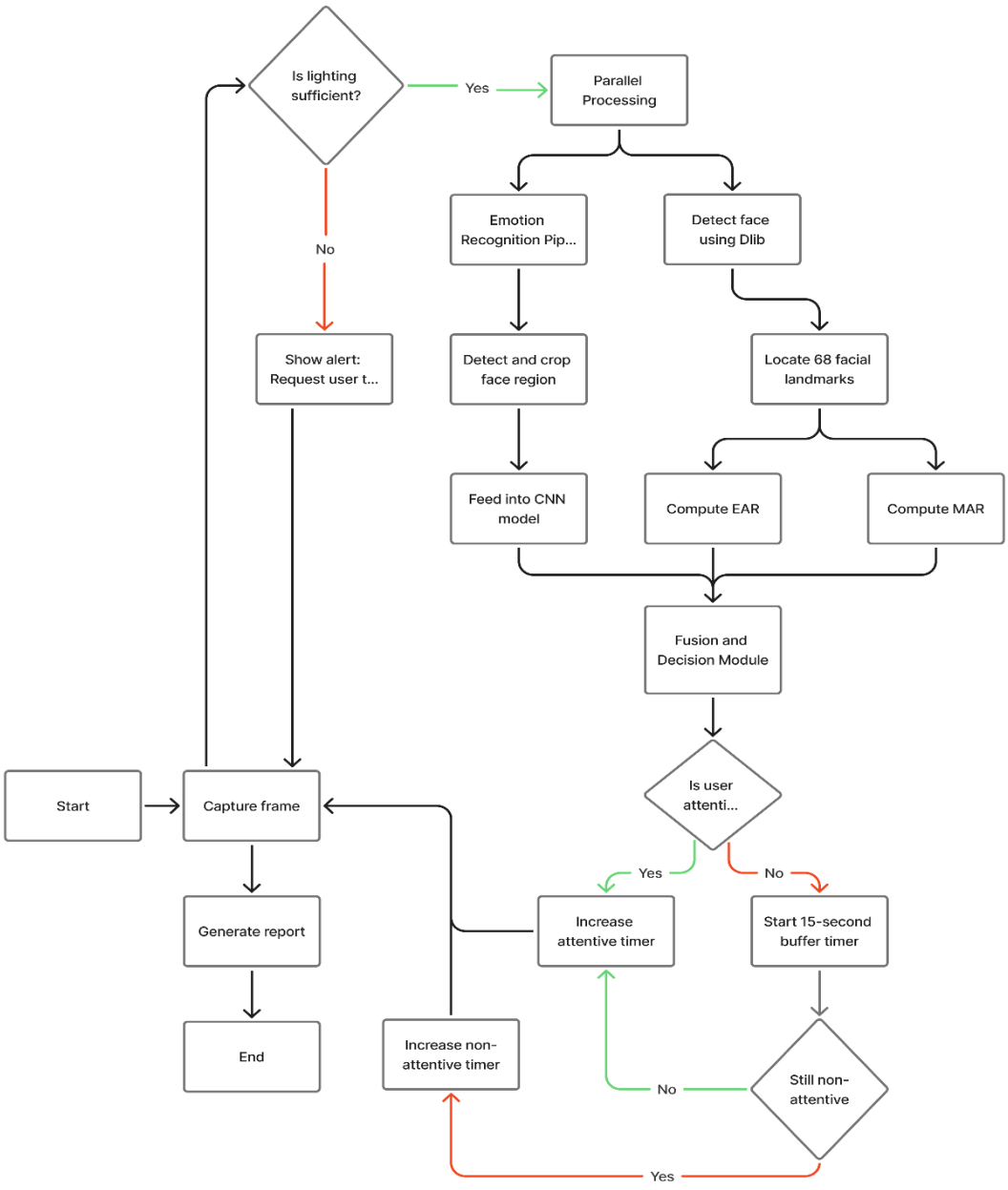


Figure 5.1: Flow Diagram

Chapter 6

RESULT ANALYSIS AND DISCUSSION

The CNN model's performance in classifying the seven basic emotions was rigorously evaluated, achieving an overall Training accuracy of 0.84996 (85%) on the training dataset and 0.6906 (70%) on test data of 7,178 samples. The model demonstrates high proficiency in recognizing certain states, most notably Happy, which achieved the highest scores with a precision of 0.86, recall of 0.84, and an f1-score of 0.85. Similarly, Surprise (f1=0.79) and Disgust (f1=0.73) were classified with high reliability.

Table 6.1: Classification Report

Class	**Precision**	**Recall**	**f1-score**	**Support**
0	0.66	0.62	0.63	985
1	0.72	0.75	0.73	102
2	0.63	0.51	0.57	1043
3	0.85	0.85	0.85	1765
4	0.59	0.53	0.56	1210
5	0.78	0.80	0.79	795
6	0.58	0.74	0.65	1278
accuracy				**0.69**
macro avg	0.69	0.69	0.68	7178
weighted avg	0.69	0.69	0.69	7178

We are generating an end-of-session pie chart report for each attendee to visualize their engagement. This chart breaks down the total meeting time into three segments: time spent actively attentive, time spent non-attentive, and the duration their image was successfully captured as shown in fig 6.1.

However, the model struggled a bit with negative and ambiguous emotions. Sad exhibited the lowest performance, with a low recall of 0.53 and an f1-score of only 0.56. Fear and Angry also showed comparatively lower f1-scores (0.57 and 0.63, respectively), suggesting difficulty distinguishing between these subtle negative expressions. The Neutral class, despite having a high support count (1,278), also shows a notable imbalance between precision (0.58) and recall (0.74).

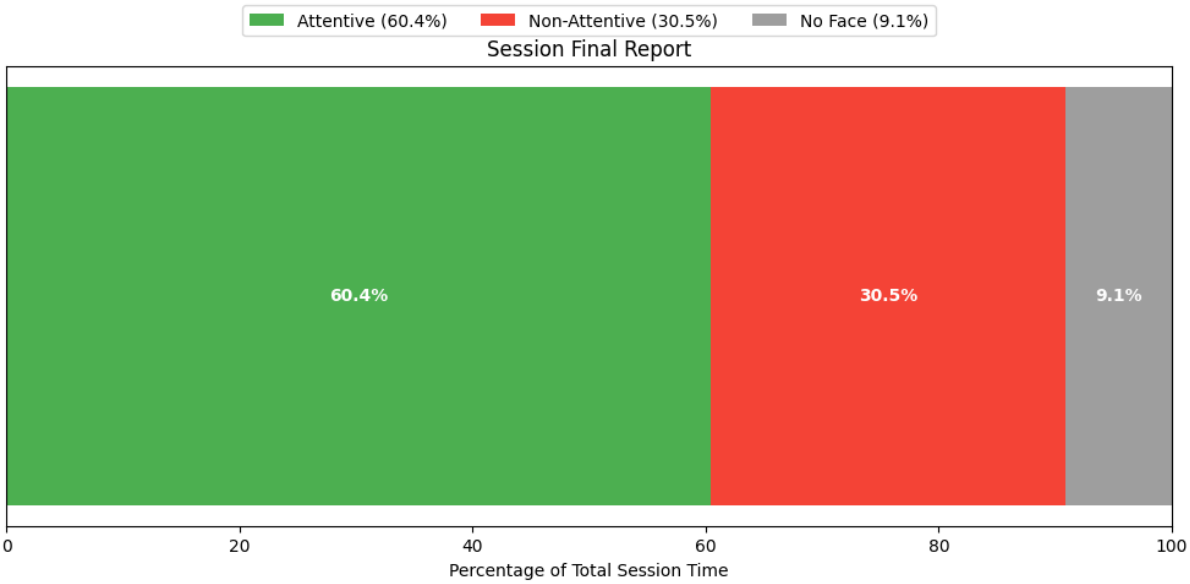


Figure 6.1: Pie chart of final result of an attendee

Chapter 7

CONCLUSIONS AND FUTURE ENHANCEMENTS

This paper presented a robust, real-time multimodal approach for engagement detection tailored for diverse learning environments. By fusing the power of a Convolutional Neural Network for emotion classification (trained on FER-2013 and supplementary data) with the computationally efficient Eye Aspect Ratio and Mouth Aspect Ratio features, our system successfully quantifies student attentiveness and participation. The architecture demonstrated solid performance on the emotion component, achieving 70% overall accuracy and particularly strong results for the 'Happy' state (f1=0.85). By integrating EAR and MAR derived from Dlib facial landmarks we successfully established a low-latency, generalized framework capable of distinguishing between focused attention, distraction, and active engagement.

A primary limitation is the model's reduced accuracy for subtle negative emotions, even with dataset resampling. Addressing this requires a future focus on creating and refining more balanced training datasets.

Additionally, the current approach relies solely on visual analysis of the end-user. The model's utility could be greatly enhanced by incorporating the audio modality, which provides essential context on user engagement, differentiating between distracting ambient noise and relevant, active speech.

BIBLIOGRAPHY

A-masiri, P. and Kerdvibulvech, C. (2023). Anime face recognition to create awareness. *International Journal of Information Technology*, 15(7):3507–3512.

Azeez, R. A., Jamil, A. S., and Mahdi, M. S. (2023). A partial face encryption in real world experiences based on features extraction from edge detection. *Int. J. Interact. Mob. Technol.*, 17(7):69–81.

Fatima, S. A., Kumar, A., and Raoof, S. S. (2021). Real time emotion detection of humans using mini-xception algorithm. In *IOP conference series: materials science and engineering*, volume 1042, page 012027. IOP Publishing.

Gapi, T., Matthew G. Magbitang, R., and F. Villaverde, J. (2021). Classification of attentiveness on virtual classrooms using deep learning for computer vision. In *Proceedings of the 2021 11th International Conference on Biomedical Engineering and Technology*, pages 34–39.

Ghoddoosian, R., Galib, M., and Athitsos, V. (2019). A realistic dataset and baseline temporal model for early drowsiness detection. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition workshops*, pages 0–0.

Khan, R. and Debnath, R. (2020). Human distraction detection from video stream using artificial emotional intelligence. *International Journal of Image, Graphics and Signal Processing*, 14(2):19.

Khandare, H., Rajak, A., and Tripathi, R. (2023). A deep learning framework for non-intrusive driver drowsiness detection. In *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, pages 1–5. IEEE.

Mehta, S., Dadhich, S., Gumber, S., and Jadhav Bhatt, A. (2019). Real-time driver drowsiness detection system using eye aspect ratio and eye closure ratio. In *Proceedings of international conference on sustainable computing in science, technology and management (SUSCOM), Amity University Rajasthan, Jaipur-India*.

Shah, N. A., Meenakshi, K., Agarwal, A., and Sivasubramanian, S. (2021). Assessment of student attentiveness to e-learning by monitoring behavioural elements. In *2021 International conference on computer communication and informatics (ICCCI)*, pages 1–7. IEEE.

Shamika, U. B. P., Weerakoon, W. A. C., Panduwawala, P. K. P. G., and Dilanka, K. A. P. (2021). Student concentration level monitoring system based on deep convolutional neural network. In *2021 International Research Conference on Smart Computing and Systems Engineering (SCSE)*, volume 4, pages 119–123. IEEE.

Sharma, P., Joshi, S., Gautam, S., Maharjan, S., Khanal, S. R., Reis, M. C., Barroso, J., and de Jesus Filipe, V. M. (2022). Student engagement detection using emotion analysis, eye tracking and head movement with machine learning. In *International conference on technology and innovation in learning, teaching and education*, pages 52–68. Springer Nature Switzerland.

Szeliski, R. (2022). *Computer vision: algorithms and applications*. Springer Nature.

Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages I–I. IEEE.

Voulodimos, A., Doulamis, N., Doulamis, A., and Protopapadakis, E. (2018). Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018(1):7068349.

APPENDICES

Appendix A

CNN Model Code

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from imblearn.over_sampling import RandomOverSampler
import cv2

data = pd.read_csv('fer2013.csv')

# The emotion labels are integers 0-6
# 0: Angry, 1: Disgust, 2: Fear, 3: Happy, 4: Sad,
5: Surprise, 6: Neutral
emotion_labels = {0: 'Angry', 1: 'Disgust', 2: 'Fear', 3: 'Happy',
4: 'Sad', 5: 'Surprise', 6: 'Neutral'}



pixels = data['pixels'].tolist()
emotions = data['emotion'].values



images = []
for pixel_sequence in pixels:
    image_data = [int(p) for p in pixel_sequence.split(' ')]
    image_data = np.asarray(image_data).reshape(48, 48)
    images.append(image_data)

images = np.asarray(images)
images = np.expand_dims(images, -1)



emotions_one_hot = to_categorical(emotions, num_classes=7)
```

```
X_train, X_val, y_train, y_val = train_test_split
(images, emotions_one_hot, test_size=0.2, random_state=42)


datagen = ImageDataGenerator(
    rotation_range=10,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.1,
    zoom_range=0.1,
    fill_mode='nearest'
)


ros = RandomOverSampler(random_state=42)
X_train_flat = X_train.reshape(X_train.shape[0], -1)
y_train_idx = np.argmax(y_train, axis=1)

X_resampled, y_resampled = ros.fit_resample(X_train_flat, y_train_idx)
X_resampled = X_resampled.reshape(-1, 48, 48, 1)
y_resampled = to_categorical(y_resampled, num_classes=7)
print("Original training set shape:", X_train.shape)
print("Resampled training set shape:", X_resampled.shape)

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten,
    Dense, Dropout, BatchNormalization
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import ModelCheckpoint


model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(48, 48, 1)),
```

```
    BatchNormalization(),
    Conv2D(64, (3, 3), activation='relu'),
    BatchNormalization(),
    MaxPooling2D(pool_size=(2, 2)),
    Dropout(0.25),

    Conv2D(128, (3, 3), activation='relu'),
    BatchNormalization(),
    MaxPooling2D(pool_size=(2, 2)),
    Dropout(0.25),

    Conv2D(256, (3, 3), activation='relu'),
    BatchNormalization(),
    MaxPooling2D(pool_size=(2, 2)),
    Dropout(0.25),

    Flatten(),
    Dense(512, activation='relu'),
    BatchNormalization(),
    Dropout(0.5),
    Dense(7, activation='softmax')
])

model.compile(optimizer=Adam(learning_rate=0.0001), loss='categorical_cr
model.summary()

checkpoint = ModelCheckpoint("best_model_v2.h5", monitor='val_accuracy'

history = model.fit(
    datagen.flow(X_resampled, y_resampled, batch_size=64),
    validation_data=(X_val, y_val),
    epochs=200,
    callbacks=[checkpoint]
)
```

```
from sklearn.metrics import classification_report

print(classification_report(
    y_true_classes,
    y_pred_classes,
    target_names=['Angry','Disgust','Fear','Happy','Sad',
        'Surprise','Neutral']
))

import numpy as np
from sklearn.metrics import confusion_matrix, classification_report
import seaborn as sns
import matplotlib.pyplot as plt

# Predictions
y_pred = model.predict(X_val)
y_pred_classes = np.argmax(y_pred, axis=1)
y_true = np.argmax(y_val, axis=1)

emotionclasses = ['Angry','Disgust','Fear','Happy','Sad',
    'Surprise','Neutral']
cm = confusion_matrix(y_true, y_pred_classes)

plt.figure(figsize=(8,6))
sns.heatmap(cm, annot=True, fmt='d', xticklabels= emotionclasses,
        yticklabels=emotionclasses, cmap='Blues')
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()

# Classification Report
print(classification_report(y_true, y_pred_classes))
```

Appendix A

Function Code

```
import cv2
import dlib
import numpy as np
from tensorflow.keras.models import load_model
from scipy.spatial import distance as dist
import time
import matplotlib.pyplot as plt

# 1. Constants and Model Loading (Assuming previous setup is available)

MODEL_PATH = 'best_model.h5'
PREDICTOR_PATH = "shape_predictor_68_face_landmarks.dat"

try:
    emotion_model = load_model(MODEL_PATH)
    detector = dlib.get_frontal_face_detector()
    predictor = dlib.shape_predictor(PREDICTOR_PATH)
except Exception as e:
    print(f"FATAL ERROR: Could not initialize models or dlib: {e}")
    exit()

# Define landmark indices
L_EYE_START, L_EYE_END = 36, 42
R_EYE_START, R_EYE_END = 42, 48
MOUTH_START, MOUTH_END = 60, 68
EMOTION_CLASSES = ['Angry', 'Disgust', 'Fear', 'Happy', 'Sad',
'Surprise', 'Neutral']

# Thresholds
EAR_THRESHOLD = 0.25
```

```
MAR_THRESHOLD = 0.70
LOW_LIGHT_THRESHOLD = 50
BUFFER_TIME_SECONDS = 15.0

#  2. Geometric Feature Functions (EAR & MAR)
def eye_aspect_ratio(eye):
    A = dist.euclidean(eye[1], eye[5])
    B = dist.euclidean(eye[2], eye[4])
    C = dist.euclidean(eye[0], eye[3])
    return (A + B) / (2.0 * C)

def mouth_aspect_ratio(mouth):
    A = dist.euclidean(mouth[1], mouth[7])
    B = dist.euclidean(mouth[2], mouth[6])
    C = dist.euclidean(mouth[3], mouth[5])
    D = dist.euclidean(mouth[0], mouth[4])
    return (A + B + C) / (3.0 * D)

# 3. Final Summary and Chart Function
def display_summary_chart(attentive_time, unattentive_time, no_face_time
    total_time = attentive_time + unattentive_time + no_face_time
    if total_time == 0:
        print("\nSession too short for analysis.")
        return

    # Calculate percentages
    attentive_percent = (attentive_time / total_time) * 100
    unattentive_percent = (unattentive_time / total_time) * 100
    no_face_percent = (no_face_time / total_time) * 100

    # Data for the chart
    labels = ['Engagement Breakdown']
    attentive_data = [attentive_percent]
    unattentive_data = [unattentive_percent]
    no_face_data = [no_face_percent]
```

```
# Colors for states
colors = { 'Attentive': '#4CAF50', 'Non-Attentive': '#F44336',
'No Face': '#9E9E9E' }

# Console Output (Kept for detailed numbers)
print("\n" + "="*50)
print("FINAL SESSION ENGAGEMENT SUMMARY")
print("="*50)
print(f"Total Session Time: {total_time:.2f} seconds")
print(f"Attentive Time:       {attentive_time:.2f}
s ({attentive_percent:.2f}%)")
print(f"Non-Attentive Time:
{unattentive_time:.2f} s ({unattentive_percent:.2f}%)")
print(f"No Face Detected Time: {no_face_time:.2f}
s ({no_face_percent:.2f}%)")
print("="*50)

# Generate Stacked Horizontal Bar Chart

plt.figure(figsize=(10, 5))
# 1. Attentive Bar
plt.barh(labels, attentive_data, color=colors['Attentive'],
         label=f'Attentive ({attentive_percent:.1f}%)', height=0.5)

# 2. Non-Attentive Bar (Stacked on Attentive)
plt.barh(labels, unattentive_data, left=attentive_data,
color=colors['Non-Attentive'],
         label=f'Non-Attentive ({unattentive_percent:.1f}%)', heigh

# 3. No Face Bar (Stacked on Attentive + Non-Attentive)
plt.barh(labels, no_face_data, left=[a + u
for a, u in zip(attentive_data, unattentive_data)],
color=colors['No Face'],
label=f'No Face ({no_face_percent:.1f}%)', height=0.5)
```

```
    # Adding Text Labels (inside the bars)
    current_offset = 0
    for percentage, label, color in zip([attentive_percent, unattentive
        ['Attentive', 'Non-Attentive', 'No Face'],
        [colors['Attentive'], colors['Non-Attentive'],
        colors['No Face']]):
        if percentage > 1.0:
            plt.text(current_offset + percentage / 2, 0, f'{percentage:
                    color='white', fontweight='bold', fontsize=10)
        current_offset += percentage


    plt.xlim(0, 100)
    plt.xlabel("Percentage of Total Session Time")
    plt.title("Session Final Report")
    plt.legend(loc='upper center', bbox_to_anchor=(0.5, 1.15), ncol=3)
    plt.yticks([])
    plt.tight_layout()
    plt.show()

# 4. Main Real-Time Detection Loop

def start_realtime_detection():
    cap = cv2.VideoCapture(0)
    if not cap.isOpened():
        print("Error: Cannot open webcam.")
        return

    attentive_time = 0.0
    unattentive_time = 0.0
    no_face_time = 0.0
    is_attentive = True
    buffer_start_time = None
    last_frame_time = time.time()
```

```
print("Starting real-time detection with 15s non-attentive buffer.
Press 'q' to quit and summarize.")

while True:
    ret, frame = cap.read()
    if not ret:
        break

    current_time = time.time()
    dt = current_time - last_frame_time
    last_frame_time = current_time

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = detector(gray, 0)

    avg_brightness = np.mean(gray)
    light_warning = avg_brightness < LOW_LIGHT_THRESHOLD

    current_frame_is_attentive = False
    if len(faces) == 0:
        state_display = "WARNING: No face detected!"
        text_color = (158, 158, 158)
        no_face_time += dt

        buffer_start_time = None
        is_attentive = False

    else:
        face = faces[0]

        # 4.1 Feature Calculation
        landmarks = predictor(gray, face)
        landmarks = np.array([[p.x, p.y] for p in landmarks.parts()
        ear = (eye_aspect_ratio(landmarks[L_EYE_START:L_EYE_END]) +
```

```
        eye_aspect_ratio(landmarks[R_EYE_START:R_EYE_END]))
mar = mouth_aspect_ratio(landmarks[MOUTH_START:MOUTH_END])

# 4.2 Emotion Classification
(x, y, w, h) = (face.left(), face.top(), face.width(), face
face_img = cv2.resize(gray[y:y+h, x:x+w], (48, 48))
face_img = np.expand_dims(face_img.astype("float") / 255.0,
axis=(0, -1))
emotion_preds = emotion_model.predict(face_img, verbose=0)[
emotion_label = EMOTION_CLASSES[np.argmax(emotion_preds)]

# 4.3 Classification Logic (Attentive vs. Non-Attentive)
is_eyes_open = ear > EAR_THRESHOLD
is_yawning = mar > MAR_THRESHOLD
is_positive_or_neutral = emotion_label in
['Happy', 'Neutral', 'Surprise']

# Direct Non-Attentive Triggers
is_clear_distraction = False
if not is_eyes_open:
    # Eyes closed (very low EAR) -> Immediate Non-Attentive
    is_clear_distraction = True
    reason = "Eyes Closed/Drowsy"
elif is_yawning:
    # High MAR (Yawn) -> Immediate Non-Attentive
    is_clear_distraction = True
    reason = "Yawning/High MAR"
elif emotion_label in ['Sad', 'Fear', 'Angry']:
    # Strong negative emotion -> Immediate Non-Attentive
    is_clear_distraction = True
    reason = f"Negative Emotion ({emotion_label})"

# Determine the final attentive state for the current frame
current_frame_is_attentive = is_eyes_open and is_positive_o
```

```
#  4.4 State Management and Timer Logic

is_ears_low = ear < EAR_THRESHOLD
is_maring = mar > MAR_THRESHOLD

is_immediate_unattentive = (is_ears_low or is_maring) and lo

if is_immediate_unattentive:
    REQUIRED_BUFFER_TIME = 1.0
else:
    REQUIRED_BUFFER_TIME = BUFFER_TIME_SECONDS

#  Display Warnings and State
if light_warning:
    state_display = "LOW LIGHT WARNING: Find a brighter spo
    text_color = (0, 0, 255)

elif current_frame_is_attentive:
    state_display = "ATTENTIVE"
    text_color = (0, 255, 0)
    attentive_time += dt
    buffer_start_time = None
    is_attentive = True
else:

    if is_attentive:
        buffer_start_time = current_time
        is_attentive = False

    buffer_time_elapsed = current_time - buffer_start_time

    if buffer_time_elapsed >= REQUIRED_BUFFER_TIME:
        state_display = "UNATTENTIVE (Focus Required!)"
        text_color = (0, 0, 255) # Red
```

```
                    unattentive_time += dt
                else:
                    time_remaining = REQUIRED_BUFFER_TIME - buffer_time
                    state_display = f"NON-ATTENTIVE (Buffer: {time_rema
                    text_color = (0, 165, 255)

            cv2.putText(frame, state_display, (10, 30), cv2.FONT_HERSHE
            cv2.putText(frame, f"ATTN: {attentive_time:.1f}s |
            NON-ATTN: {unattentive_time:.1f}s",
                        (10, frame.shape[0] - 10),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.6,
                        (255, 255, 255), 2)

        cv2.imshow('Real-Time Engagement Monitor (Press Q to quit)', fra

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

    cap.release()
    cv2.destroyAllWindows()

    display_summary_chart(attentive_time, unattentive_time,
    no_face_time)

if __name__ == "__main__":
    start_realtime_detection()
```