Faculty of Engineering,
Alexandria University
Computer and Communications Engineering Department

Graduation Project Submitted In Partial
Fulfillment Of The B. SC. Degree

DrowsyGuard
(Drowsiness Detection and Music Recommender Mobile Based Application)

Presented by
Sarah Mohamed Fared
Nesreen Diaa Eldin
Salma Hamdy Hafez
Sophia Samir Arida
Nada Mostafa Saber
Amr Abdelaziz Attia

Defended on: **January, 2025**

Dr. Samia Hafez	Thesis Supervisor
Dr. Ahmed Kosba	Examiner

Academic Year: 2024/2025

Acknowledgements

First and foremost, we would like to express our heartfelt gratitude to Allah, whose infinite mercies and guidance crowned every step of this journey and made the completion of this project possible.

We would also like to express our profound debt of gratitude to Dr. Samia Hafez for the exceptional mentorship characterized by instinctive guidance and support throughout this work. Indeed, her dedication, constructive feedback, and encouragement to push harder have been very invaluable. Her expertise and commitment have played a very important role in shaping this thesis into its final form.

Equally, we would like to express our profound appreciation to our families for their endless love, patience, and sacrifice in providing us with the wherewithal and perseverance to keep going. The constant support and faith of our loved ones have formed a source of inspiration and motivation that has made it possible to achieve this accomplishment.

We would also like to thank all those who contributed to this effort, whether by active participation or by moral support, and enabled us to tide over difficulties that arose from time to time.

Abstract

Driver fatigue and/or drowsiness increase road-accident rates, even though the consequences are tremendous for individuals and society alike. This project presents the development of a mobile smartphone application that will contribute toward improving road safety by utilizing real-time driver monitoring interspersed with personalized interventions once early signs of drowsiness are detected.

By applying active computer vision and state-of-the-art artificial intelligence technologies, this driver assistance app detects early drivers' drowsiness events by analyzing facial expressions, such as blinking rates, and yawning frequency.

Apart from fatigue detection, the system includes a special music recommendation feature: providing curated playlists that better match the user's instant emotional state and personal taste. This system integrates Spotify's API and dynamically changes music suggestions according to real-time emotional analysis, user profile data, and cultural considerations. The application ensures that the user is engaged with it and focuses more on staying alert during long or late-night drives.

Explainable AI combined with federated learning reinforces the system in terms of transparency and security while maintaining user privacy. The application is designed for mobile platforms with consideration for low resource consumption and efficiency in user interaction. With this, a solution appropriate to meet critical safety needs with accessibility and personalization, yet maintaining driving pleasure, is provided.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 General

Road safety is a big issue all over the world. Every day there are so many accidents because of the exhaustion and drowsiness felt by drivers. According to previous studies, drowsy driving causes a large portion of accidents leading to injuries, deaths and significant financial losses. Working on this would help improve road safety and minimize the dangers of driving for a long time [18].

Advances in technology such as artificial intelligence (AI) and machine learning now allow for real-time drowsiness detection. AI can show signs of tiredness by evaluating features on our face like blinking patterns, closing of eyes and yawning. Music also is well known to influence feelings and improve focus. The use of these technologies provides an innovative approach to improve driver alertness [19].

The project is concerned with building a mobile application to detect driver drowsiness through facial recognition along with recommending personalized music playlist according to the user's mood and emotion. We aim at developing an application that keeps the driver safe but also makes driving fun and entertaining. With help of this system, drivers can get real-time alerts and an interesting playlist of music to keep them engaged while driving.

1.2 Motivation

Millions of people are driving around the world. However, driving has some serious risks. One of the biggest challenges drivers encounter is staying focused and awake, especially on long or late-night trips. Studies show that driver fatigue may be responsible for a lot of road accidents, and drowsy drivers are more than three times as likely as awake drivers to be involved in collisions. Unlike distractions like phone usage, drowsiness can be difficult to be self-detected, so the application gives the drivers little warning before they lose their ability to make safe decisions exposing themselves and other road users to danger leading to injuries and deaths [20].

To solve this problem, existing technologies are often integrated into high-end vehicles, such as luxury car drowsiness detection systems. However, most of the drivers cannot access these solutions because of the high cost and unavailability of such solutions. Even with technology, interventions often lack real-time personalization, leaving room for improvement. Systems currently available can recognize fatigue, but they don't provide immediate solutions to help the driver refocus [21].

For ages, music has been a mood lifter, a stress buster, and a powerful concentration enhancer. But traditional music systems do not change according to the driver's mood and emotions, which may not fight drowsiness up to the mark. Music recommendation system based on real-time emotion analysis could be a powerful tool in helping a driver stay awake and entertained during long drives [19].

The motivation of this application is to fill the gap in existing systems by providing an affordable, accessible and effective drowsiness detection solution by integrating a drowsiness detection system with music recommendation system based-on detected emotion. This approach improves road safety by offering a system that is both functional and entertaining.

1.3 Scope Of Work

The scope of this project is developing mobile application using deep learning and computer vision to detect drowsy drivers while driving along with an emotion-based music recommendation system. There will be no hardware implementation in our scope for this project, it will be left as future work.

1.4 Organization of Thesis

The report is organized into three chapters.

- In Chapter 1, a brief introduction about the addressed problem and the motivation are given.
- In Chapter 2, the essential background, survey of the Related work and applications, and feature definitions are provided.
- In Chapter 3, the proposed techniques, systems architectures and experimenting AI models are provided
- In Appendix, there is a literature review of some papers related to ours that helped us reform our idea.

Chapter 2

2.1 Background

2.1.1 Deep Learning

Deep learning is a specialized field within machine learning which is itself part of artificial intelligence (AI) [22] as shown in Figure 2.1. It uses architectures inspired by the structure of the human brain's neurons and how it operates [23] [24]. Just as the human brain can extract complex features from images, text, and voices. Deep learning models are designed to perform similar tasks that identify patterns and make sense of complex data in ways that imitate how humans process information.

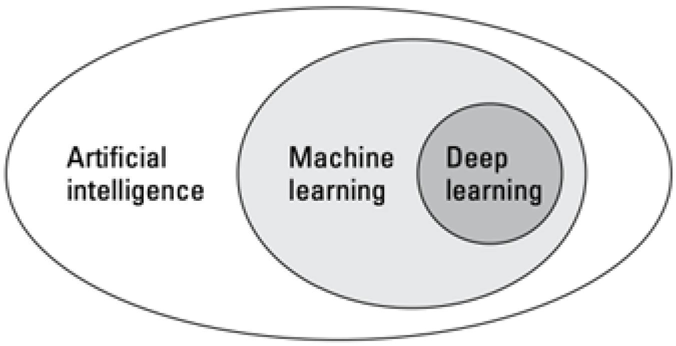

Figure 2.1: The Deep Learning is a subset of a subset of AI [1].

- **Artificial Neural Networks (ANNs)**

The core of deep learning is Artificial Neural Networks (ANNs), which are designed to simulate the biological neural networks of the human brain. [22] These networks consist of layers of interconnected nodes, known as neurons, which are represented by complex, non-linear functions. Figure 2.2 shows how the neurons are connected to one another through weighted links. ANNs Consist mainly of three layers: Input layer, Hidden layers and output layer [22].

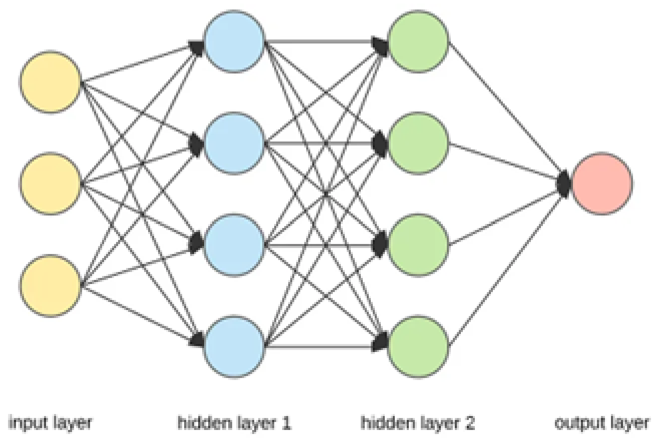

Figure 2.2: Layers of Artificial Neural Networks (ANN) [2].

- **Components of Artificial Neural Networks (ANNs)**

  1. **Neurons** are the building blocks of the artificial neural network [24]. It is responsible for the flow of information from the input layer to the output layer. Each neuron receives input and processes it to produce output.

  2. **Received input** is represented by features such as text, data images, and signals.

  3. **Weights** are numerical values associated with the connections between neurons to determine the strength of these connections.

  4. **Bias** is the numerical value associated with the layer added to the sum of the input to provide a better-fitted model.

  5. **Activation function** is a mathematical function applied to the sum of the weighted inputs of each neuron. This function introduces non-linearity to the network [24], enabling the model to capture complex patterns in the data. It determines whether the data flowing through should be allowed to contribute to the output based on evaluating the weighted sum of its input. Common activation functions include:

     - **Sigmoid**: It compresses input values into a range between 0 and 1. It is commonly used in binary classification problems to represent probabilities. The output of the Logistic Sigmoid function is saturated for higher and lower inputs, which leads to a vanishing gradient problem [24] [25].

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$

- **Tanh**: The Tanh function also ranges the inputs in $[-1, 1]$. The problems of Logistic Sigmoid function such as vanishing gradient and computational complexity also exist with Tanh function [24] [25].

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.2}$$

- **ReLU (Rectified Linear Unit)**: Its output is equal to the input if the input is a positive value. For negative values, the output is zero. ReLU activation functions are better in practice but can face the "dying ReLU" issue, where neurons stop learning due to a zero gradient. Despite this, ReLU is widely used in hidden layers and output layers, especially when the response variable is continuous and positive [24].

$$f(x) = \begin{cases} x, & \text{if } x > 0, \\ 0 & \text{if } x \leq 0 \end{cases} \tag{2.3}$$

- **Softmax**: It provides a probability distribution over mutually exclusive output classes [24] , making it suitable for multi-class classification problems.

$$f(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \tag{2.4}$$

The following figure 2.3 shows the building unit of ANNs.

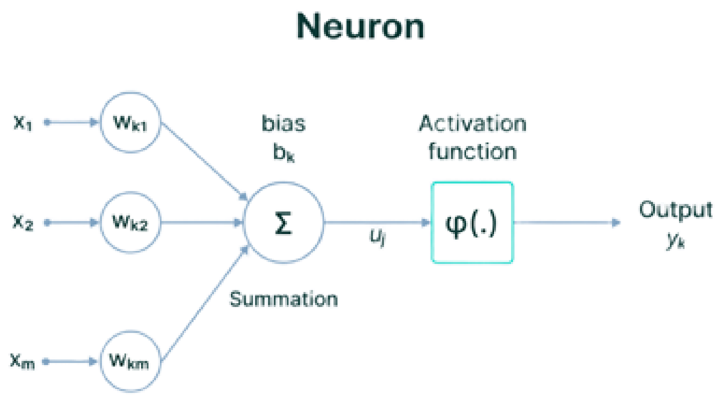

Figure 2.3: Building blocks of the Artificial Neural Network [3].

6. **Layers of ANNs**

   - Input Layer:

   The input layer receives the raw data features as input. The number of neurons in this layer corresponds to the size of the data, if it's a numerical dataset, it matches the number of features. For an image dataset, it corresponds to the flattened size of the image. This layer acts as a gateway, allowing information to flow into the neural network.

   - Hidden Layers:

   The hidden layer is a set of interconnected layers that connects the input layer to the output layer. It is responsible for extracting complex features from the data and processing output.

   - Output Layer:

   The output layer contains a number of neurons based on the type of problem being solved. For classification problems, the number of neurons is equal to the number of classes. It produces the final result of the network [24].

7. **Simple ANN structure can be represented by the following equation**

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{2.5}$$

   Where:

   . x_i: Input raw data.
   . w_i: Weights associated with each input.
   . b: Bias term.
   . f: activation function.
   . y: output.

8. **Feedforward Network**

   Feedforward Network is a type of artificial neural network where information flows in one direction: from the input layer, through one or more processing layers, to the output layer. It has no connections within the same layer (no intralayer connections) and no backward connections between layers (no supralayer connections) [24].

9. **Training Artificial Neural Network (Backpropagation):**

The goal of Backpropagation is to optimize the selected loss function. During each epoch, the model "learns" by adjusting its weights and biases to reduce the loss [26].

10. **Cost/Loss Function**:

    It is a mathematical function applied to the network's output to measure the overall error. The loss function tells the machine how far away the combination of weights and biases is from the optimal solution [26], summing up the errors across all data points. By minimizing the cost function with respect to the weights and biases, the network adjusts its parameters to reduce the error and improve performance. Common cost functions include:

    - Mean Squared Error (MSE): Used for regression tasks.

$$J(W,b) = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y_i})^2 \tag{2.6}$$

    - Cross-Entropy Loss: Common in classification tasks, it measures the divergence between predicted probabilities and actual labels [24].

$$J(W,b) = -\frac{1}{m}\sum_{i=1}^{m}\left[y_i \log(\hat{y_i}) + (1-y_i)\log(1-\hat{y_i})\right] \tag{2.7}$$

11. **Optimization Techniques**:

    Optimization in neural network training aims to find the best parameters for performance. It involves adjusting the weights of the network to minimize the cost function [27]. The most common method is gradient descent.

    Popular techniques include:

    - Stochastic Gradient Descent (SGD): Updates weights iteratively based on a subset of the training data. It needs a large number of iterations in order to achieve the best parameters. The gradient descent may not be a good choice if the training sample is too large [27].

    The weight update rule in gradient descent is given by:

$$w := w - \eta\frac{\partial J}{\partial w} \tag{2.8}$$

    Where:

- η: is the learning rate.
- $\frac{\partial J}{\partial w}$: is the gradient of the cost function.

- Adam Optimizer (*Adaptive Momentum)*:
Enhances the performance by adjusting the learning rate and using past gradients, making it more efficient than traditional methods [27].

2.1.2 Convolutional Neural Network (CNN)

CNN is a type of deep learning model and a most impressive form of an artificial neural network (ANN) that processes grid-like data, such as images and videos [23]. It is widely used for computer vision tasks, including image recognition, object detection, and classification [28]. Its architecture is specifically designed to learn visual features from large datasets and capture the spatial hierarchies in images [28].

• CNN Architecture

The CNN architecture uses a special technique called Convolution instead of relying solely on matrix multiplications like traditional neural networks. Convolutional networks use a process called convolution, which combines two functions to show how one changes the shape of the other. Its architecture model consists of convolutional, pooling operation, and fully connected layers [23]. It is designed to learn spatial hierarchies of features through a backpropagation algorithm [28]. A simple CNN architecture is illustrated in Figure 2.4.

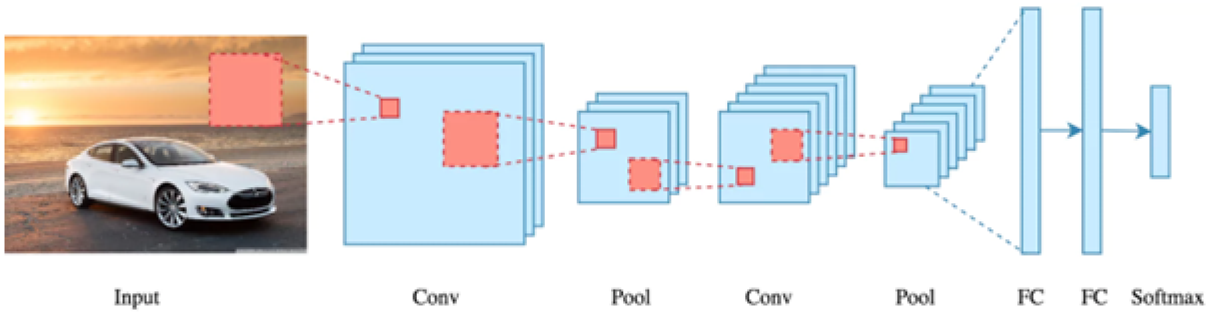

Figure 2.4: A simple CNN architecture [4].

• How Does CNNs Work?

The processing of an image in a CNN involves a series of computational steps where the image is gradually reduced in size as we move through the network. Initially, the image is scanned using convolutional filters/kernels that detect patterns and features. As these filters move across the image, they perform a convolution operation, which reduces the size of the output feature map while capturing spatial hierarchies and patterns. Despite the reduction, the final feature map retains all the information processed from the previous steps, maintaining the context and relationships between features throughout the network.

Example on a RGB image:

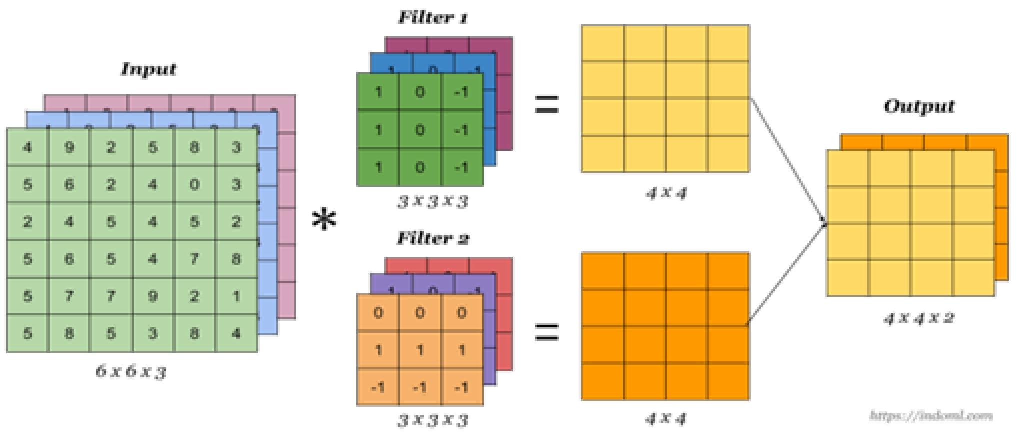

Figure 2.5: An Example of RGB Image [5].

Example on a Grayscale image:

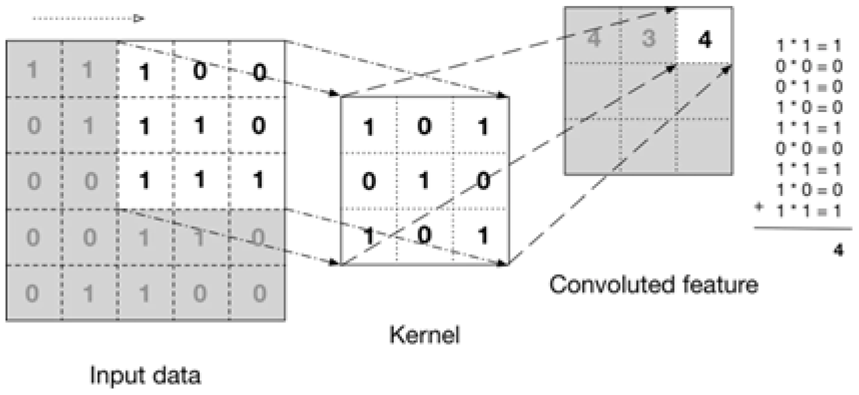

Figure 2.6: An Example of Grayscale Image [6].

- **Components of CNN Architecture**

1. **Convolutional Layers:**

   These layers scan the input image using filters (small grids) to detect patterns like edges or corners. Each filter captures specific features, and the output is called a feature map, which detects these details. It carries out feature extraction by applying convolution for linear processing and activation functions for nonlinear transformations [28]. Parameters like stride control how far the filter moves with each step, and padding ensures the image edges are included in the process. A convolutional layer computes the output of neurons by calculating the scalar product between their weights and the corresponding local regions of the input volume [23].

2. **Pooling Layers:**

   Pooling comes after convolution to downsample and simplify the feature maps [28]. It reduces their size by summarizing the information, making the network less computationally expensive. Additionally, it facilitates downsampling [23].

   - Max Pooling selects the largest value in a small area (e.g., 2x2) [28].
   - Average Pooling takes the average of values in that area [28].

3. **Fully Connected Layers:**

   These layers will perform the same function as ANNs [23]. They take the simplified feature maps, flatten them into a single vector, and pass them through dense connections to make predictions [28]. They're where the model decides what category an image belongs to. Using the ReLu activation function between layers will achieve higher performance [23].

4. **Activation Function:**

   The activation function for the final fully connected layer differs based on the task. For multiclass classification, the softmax function is typically used. It normalizes the real-valued outputs of the layer into probabilities, ensuring each value lies between 0 and 1 and the total sums to 1, representing the likelihood of each target class [28].

   - **ReLU:** Allows only positive values, making the network efficient by ignoring negative activations.
   - **Softmax:** Converts the final layer's outputs into probabilities, showing how likely each class is for the given input.

- **CNN vs. ANN**

- ANN is a general-purpose neural network that can be used for a wide range of tasks, including classification, regression, and pattern recognition.
- CNN is a type of neural network that is commonly used for image recognition and computer vision tasks.

Advantages of CNN over ANN [23]:

1. Better in solving Computer-Vision tasks [28].
2. Specialized for image data [23].

3. Reduced computational complexity: There are no neurons or weights in CNN ,Unlike ANN that depend on neurons and weights [23].
4. Spatial hierarchies: CNNs can understand the hierarchical structure of image data (e.g., edges, textures, and patterns), which is crucial for image recognition tasks.

2.1.3 Computer Vision

- **What is Computer Vision?**

Computer Vision is a branch of Artificial Intelligence that focuses on the capabilities of machines to interpret and analyze visual data, like images and videos, and make decisions based on this data. It emulates human vision but focuses on the automation of tasks such as object detection, recognition, and image analysis, thus allowing devices to 'see' and understand the visual world [29].

- **How Does Computer Vision Work?**

As computers only understand arrays of zeros and ones, computer vision converts images and videos into a stream of binary bits. Computer Vision begins with capturing visuals through cameras or sensors. These raw visual inputs are then subjected to preprocessing techniques designed to enhance the overall quality and reliability of the data [8].

- **Representing images:** The image consists of tiny dots called pixels where each pixel has a color represented by numerical values as shown in Figure 2.7 . If the image is grayscale each pixel has a value ranging between 0 representing black and 255 representing white and if the image is colored each pixel has 3 values: Red, Green, Blue [7].

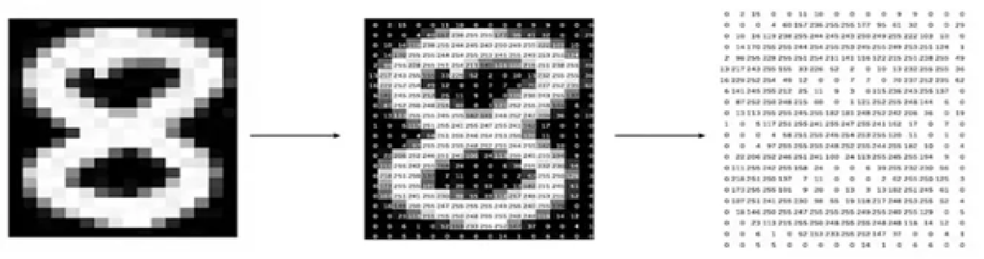

Figure 2.7: Images Representation in Computer Vision [7].

- **Preprocessing images:** Images need to be cleaned before being analyzed through resizing by adjusting dimensions to ensure uniform input dimensions for machine learning models, grayscale conversion to simplify the data and normalization that scales pixel values to a consistent range to improve training efficiency and stability [8].

- **Computer Vision Tasks**

  1. **Feature extraction:** Image is identified by key patterns and features such as corners which are the meet point of 2 edges, edges where sharp intensity changes occur, and shapes of objects. Different techniques can be used such as Canny Edge Detection which is determining boundaries between objects, Hough Transform which is detecting geometric shapes, and Histogram of Oriented Gradients which is describing the shape of objects by using gradient orientation distributions. These features are important because they serve as the building blocks for the upcoming analysis' stages. These identified features are translated into numerical representations, effectively converting the visual information into a format that machines can comprehend [8].

  2. **Object Detection and Recognition:** It is used in identifying objects in an image such as face recognition. This enables computers to not only detect the presence of objects but also understand what those objects are. There are classical methods such as HOG and SIFT and there are deep learning methods which we are concerned about using neural networks like CNNs to automatically learn patterns from the detected features [8] [29].

  3. **Image Classification:** It categorizes entire images into predefined classes or categories as visualized in Figure 2.8. This is where Convolutional Neural Networks do their role in learning complex hierarchies of features, which allows them to identify complex patterns and make highly accurate image classifications [8] [29].

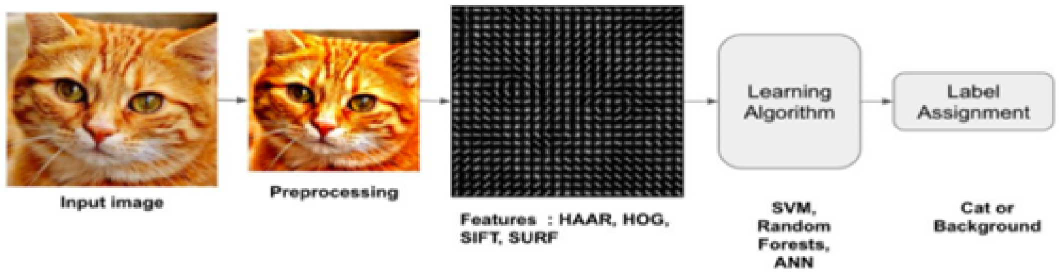

Figure 2.8: Image Classification in Computer Vision [8].

  4. **Object Tracking:** Refers, as illustrated by Figure 2.9, to the ability

of tracking object motion as objects cross consecutively through frames within a video, such as pose estimation [8] [29].

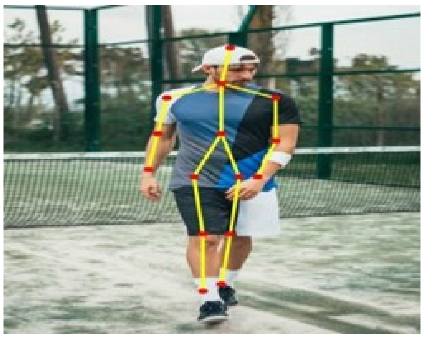
Figure 2.9: Example for Tracking Objects In Computer Vision [8].

5. **Image Segmentation:** It is the process of dividing an image into small meaningful regions, such as separating a face from the background. This technique helps in labeling each pixel within the image with its respective category and understanding the boundaries and categories of each pixel within the identified objects, as seen in Figure 2.10 [8].

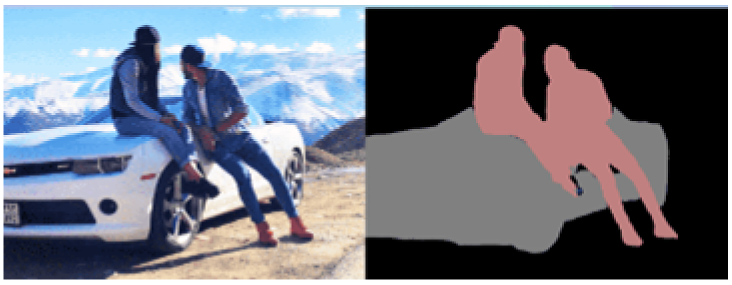
Figure 2.10: Example for Image Segmentation In Computer Vision [8].

6. **Facial Analysis:** This may be used to detect eyes, mouth, or face shape for different applications using landmarks for the purpose of precise analysis [8].

- **Convolution**

Convolution is an important operation in computer vision, particularly in image processing and deep learning. It involves applying a small matrix (called a kernel/filter) to an image to recognize objects through extracting features such as edges, corners, or textures. The convolution operation works by sliding the kernel across the image, computing the dot product between the kernel values and the pixel values it overlaps. This produces a new matrix (called a feature map), which highlights specific features in the image. While convolution involves flipping the kernel before applying it, correlation uses the kernel as it is. Convolution is preferred in deep learning because it more accurately reflects mathematical properties relevant to feature extraction. The results of convolution are combined hierarchically in CNNs, where earlier layers detect simple features like edges, and deeper layers recognize more complex patterns like faces or objects [7].

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i-u, j-v] \tag{2.9}$$
$$G = H * F$$

Computer Vision has made great progress, which can be due to two factors: advancements in deep learning and the availability of large amounts of visual data. These improvements have boosted vision systems from just 50% accuracy to 99% accuracy in less than a decade [8].

2.1.4 Transfer Learning

- **What is transfer learning?**

It is a machine learning technique when we have a limited available dataset for a new problem, we use a pre-trained model that makes the similar task we need to do (related tasks).
Rather than starting from scratch, we use the pre-trained model which is already learned that will solve the new challenge more effectively and efficiently and it will accelerate the learning and improve the performance.
It also reduces the overfitting as the model contains generalizable features useful for the new task [9].

- **Why is transfer learning important?**

Transfer learning offers some solutions to some main challenges [9]:

1. **Limited data:** It is hard and costly to get dataset that are labeled and huge. So, transfer learning helps us to use another pre-trained model that will reduce the dependency on large datasets.
2. **Enhanced performance:** When we have applications that need high accuracy and efficiency, we start with a pre-trained model that has already learned from huge datasets which will provide us faster and more accurate results on our task.
3. **Time and cost efficiency:** Transfer learning will reduce the training time and conserve resources by utilizing existing models, eliminating the need for training from scratch.

4. **Adaptability:** We can fine-tune the pre-trained model which makes one task to make related tasks, which make transfer learning able to make various applications.

- **How does transfer learning work?**

Transfer learning has structured process to gain knowledge from pre-trained model for new task [9]:

1. **Pre-trained model (Base model):** We start with a pre-trained model that has already learned from a large dataset general features and patterns that are related to our task. The pre-trained model consists of layers that have processed data to learn hierarchical representations, capturing low-level to complex features.
2. **Transfer layers:** Knowing the layers in the base model that contain generic information that can be used in both original and new tasks. These layers are often near the top of the network.
3. **Fine-tuning:** Fine-tune these layers by the dataset from the new task, this helps restore the pre-trained knowledge with adjusting the parameters to meet the requirements of the new task.

Figure 2.11 represents flow of Transfer Learning.

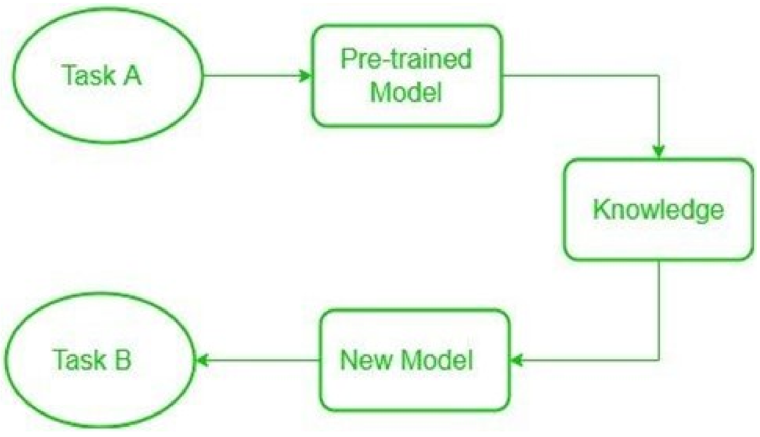

Figure 2.11: The Flow of Transfer Learning [9].

In transfer learning, we have 2 main types of layers that help in adapting the model effectively [9]:

1. **Frozen layers:** We keep those layers as it is from the pre-trained model during fine-tuning. They restore the general features the model learned from the original task, extracting the universal patterns from input data.

2. **Modifiable layers (Trained):** These layers are adjusted during fine-tuning to learn the model specific features from the new dataset. This allows the model to meet the new task requirements.

Figure 2.12 represents types of layers.

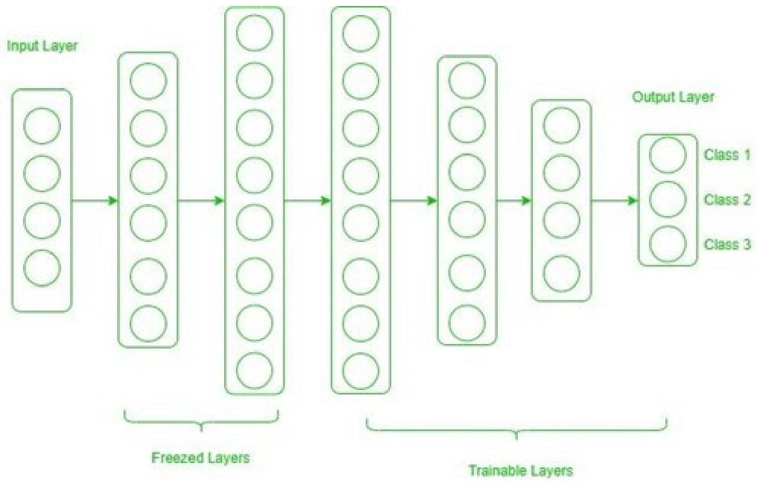

Figure 2.12: Representation to Types of Layers [9].

- **How to decide which layers to freeze or modify?**

As you want to take features from the pre-trained model, the more you have to freeze layers. The extent to which you need to take features from the pre-trained model depends on the size and the similarity of the target dataset to the original dataset [9].

- **Small and similar target dataset:** When the target dataset is small but similar to the original dataset, fine-tuning only a few layers will lead to overfitting, so we remove the last one or two fully connected layers and replace them by new layers (trained layers) that fits the target classes and the rest of layers of the model are frozen.
- **Large and similar target dataset:** When the target dataset is large and similar, overfitting is less likely to occur. So, we remove the last fully connected layer and replace it with another layer that matches the new classes, and the full model is fine-tuned on the new dataset.
- **Small and different Target Dataset:** when the dataset is small and different, the high-level features in the top layers from the original model are less useful. So, removing most of the top layers, and add new layers to fit the target classes and train from these lower layers onward to fit the unique dataset
- **Large and different target dataset:** when the dataset is large and

different, the most benefit for the model is by removing the final layers and adding new layers made specifically for the new task. Then the model is retrained without frozen layers to ensure complete adaptation.

2.1.5 Explainable AI

Explainable AI are techniques like heatmaps that provide insights primarily for developers and researchers to understand model behaviour without affecting model accuracy. They make AI model decisions transparent and interpretable by providing visualizations of which input features that influenced predictions the most.

For facial emotion recognition (FER), explainable AI improves reliability and trustworthiness by highlighting the specific image regions or features (e.g., eye region) that impact the model's classification decisions. This process happens after the prediction and is not part of the normal emotion prediction pipeline [30].

Techniques like Layer-wise Relevance Propagation (LRP) and Gradient-weighted Class Activation Mapping (Grad-CAM) help to pinpoint influential pixels or regions that aid in processing emotion predictions, which is very useful especially in sensitive applications. LRP gives pixel-level explanation while Grad-CAM provides a higher-level region visualization. LRP is typically preferred for its ease of interpretation at a granular pixel level, while Grad-CAM is chosen for tasks that need general spatial insights.

- **Techniques**

- LRP(Layer-wise Relevance Propagation):

- It explains model predictions by redistributing the output score back to the input features, assigning a relevance value to each pixel or neuron.
- Then a heat map is generated that visualizes the regions of the input image that were most relevant to the predicted emotion. For example, if the emotion predicted is "happy," the heat map will highlight the parts of the face that contributed most to this prediction (e.g., the smile).
- After generating the heat map, an Individual relevance score (IRS) is computed for each emotion. This score helps determine how much each part of the input image contributed to the prediction of each emotion.
- Based on the calculated IRS, the input image is categorized into one of 10 groups, representing different levels of relevance, ranging from no contribution to *strong positive contribution*. The score range is divided into three

intervals: 0.0-0.5 (minimal contribution), 0.5-0.7 (average contribution), and 0.7-1.0 (strong contribution).

- The pixel-level relevance mapping is highly detailed, making it ideal for tasks that require fine-grained interpretability, such as medical imaging or applications where every pixel's contribution matters [30].

- Grad-CAM(Gradient-weighted Class Activation Mapping):

- Grad-CAM creates a heatmap by using the gradients of the target class to highlight image regions that contribute to the prediction.
- Grad-CAM works by identifying the target layer focusing on the final convolutional layers of a CNN
- Then it calculates the gradient of the target class according to the feature maps of the selected layer. These gradients indicate the importance of each feature map for the target class. Using the gradients weights for the importance of each feature map are calculated.
- Lastly, the weighted feature maps are combined into a heat map which highlights the areas in the input image that were most important for the prediction
- It provides a more general, region-based interpretation and is preferred when understanding of broad areas influencing the decision is sufficient, such as in emotion recognition or object detection [31].

2.1.6 Federated Learning

Federated learning is a way to train AI models while protecting user's privacy. It enables multiple decentralized systems (devices or organizations) to train a shared model collaboratively without transferring their data to a central server [32].

- **How does it work**

As shown in Figure 2.13 multiple devices remotely share their data to collaboratively train a single deep learning model to improve it iteratively. This takes place in 4 steps:

1. Model Initialization: Each device downloads the central model from a server in the cloud.
2. Local Training: Each device trains the model on their private data.
3. Sharing Updates: Instead of sharing the raw data each device summarizes, encrypts and sends their local model updates to the server.

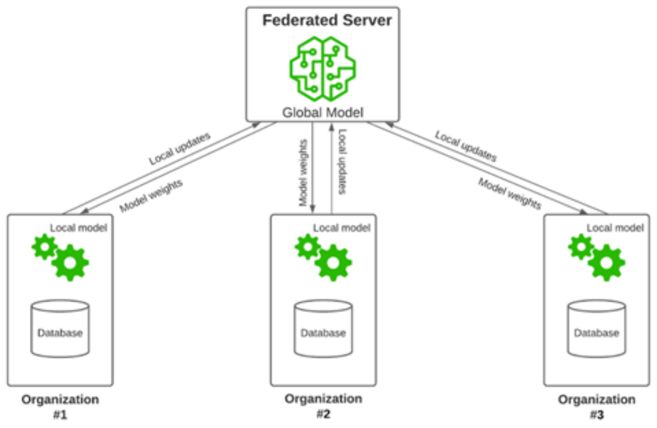

Figure 2.13: Representation of How Federated Learning Works [10].

4. Model Aggregation: The updates are decrypted, averaged, and integrated into the centralized model.

- **Its Benefits**

- **Personalization:** The global model trains on different and diverse data making it more efficient.
- **Maintain Users' Privacy:** User's data remain on his device not transferred to the global model.
- **Reduce Bandwidth Consumption:** Only model Updates are sent, not actual data.

2.2 Survey and Related Work

2.2.1 Literature Review

For detailed literature Review please check appendix A.1

	Features	**Papers**									**Our work**
		1 [33]	**2 [31]**	**3 [30]**	**4 [30]**	**5 [34]**	**6 [35]**	**7 [36]**	**8 [16]**	**9 [37]**	
1	Eye closure detection in real time	✓	X	X	X	X	X	X	X	✓	✓
2	Yawn detection	X	X	X	X	X	X	X	X	✓	✓
3	Explainable AI	X	✓	X	✓	✓	X	X	X	X	✓
4	Music recommendation system based on facial emotion (or eyes), age, and ethnicity.	X	X	✓	X	✓	✓	X	X	X	✓
5	Distinguish between blinking and sleeping	X	X	X	X	X	X	✓	✓	X	✓

Table 2.1: Comparison of Features in Papers and Our Work

2.2.2 Application Review

Features	Applications				Our work
	Beep (iOS)	Driver Sleep Alert (Android)	DrowsyAlert (iOS)	Drowsy Driving Alert (Android)	
Notify with sounds when getting sleepy	✓	✓	✓	✓	✓
Notify with a flashy screen when getting sleepy	✓	✓	✓	✓	✓
Track head movements by using sensors in the Airpods	✓	-	-	-	-
Setup working days and shift-end time to receive a notification to remind starting the session	-	-	-	✓	✓
Distinguish between blinking and sleeping	-	-	-	✓	✓
Adjustable alert volume	-	-	-	✓	✓
Adjustable eye size sensitivity	-	-	-	✓	✓
Low battery consumption	✓	-	-	✓	✓
Low resource consumption	-	-	-	✓	✓
Keep it on-screen	✓	✓	✓	✓	✓
Periodically trigger short alarm	-	✓	✓	-	✓
Trigger alert when eyes start to close in real time	-	✓	✓	-	✓
Works in background in case user wants to navigate to Google Maps	-	-	-	✓	✓
Music recommendation system based on facial emotion (or eyes), age, and ethnicity	-	-	-	-	✓
Explainable AI	-	-	-	-	✓
Trigger alert at yawn detection	-	-	-	-	✓

Table 2.2: Application Review

2.3 Need For Extension

After doing the Application and literature reviews we have gained insight into the most important features, and we have found that no application implements all of them simultaneously and efficiently in addition to some features are not implemented at all like:

1. Integration between Music recommendation system and Drowsiness Detection
2. Usage of explainable AI for a reliable system
3. Triggering alert at yawn detection
4. Distinguishing between blinking and sleeping
5. Adjustable eye size and eyeglass sensitivity

6. Usage of Federated Learning
7. Notify with sounds / flashy screen when getting sleepy.
8. Keep it on-screen
9. Low battery / resource consumption
10. Adjustable alert volume
11. Setup working schedule
12. Periodically trigger a short alarm
13. User Authentication
14. Data Encryption
15. Drowsiness Detection Data Storage
16. Personalized Model Storage
17. Spotify Linking
18. Track head movements by using sensors in the user's Airpods.
19. Integrating the system with special hardware to make a stand alone system

2.4 Scope of work and future work

Our Scope of work will be features 1-17 in the need for extension the rest will be left for future work

1. Integration between Music recommendation system and Drowsiness Detection
2. Usage of explainable AI for a reliable system
3. Triggering alert at yawn detection
4. Distinguishing between blinking and sleeping
5. Adjustable eye size and eyeglass sensitivity
6. Usage of Federated Learning
7. Notify with sounds / flashy screen when getting sleepy.
8. Keep it on-screen
9. Low battery/resource consumption

10. Adjustable alert volume
11. Setup working schedule
12. Periodically trigger a short alarm
13. User Authentication
14. Data Encryption
15. Drowsiness Detection Data Storage
16. Personalized Model Storage
17. Spotify Linking

2.4.1 Features Definition

No.	Feature Name	Feature Definition	Datasets	Models
		AI Features		
1	Eye closure detection in real time	Trigger alert when eyes start to close in real-time. Further explanation in AI features section 2.4.2 below.	• LFW (Labeled Faces in the Wild) • WIDER FACE dataset	• Haar Cascade • YOLO-face
2	Yawn Detection	Yawn detection is a method for identifying drowsiness by analyzing facial landmarks, specifically those around the mouth, to detect mouth opening.	• IBUG 300-W • LFW (Labeled Faces in the Wild)	• Dlib's Face Detection (HOG-based model) • Dlib's 68 Landmark Model on the IBUG 300-W dataset. • Haar Cascade Classifiers: `haarcascade_frontalface_default.xml` trained on LFW
3	Explainable AI	Techniques that help developers understand the decisions and predictions made by AI systems. It provides transparency and trust in AI systems, enabling developers to understand why a system made a specific decision.	-	• Grad-CAM • Layer-wise Relevance Propagation
4	Music recommendation system based on facial emotion (or eyes), age, and ethnicity	This feature enables a personalized music recommendation system that analyzes user attributes, facial emotion (or eye region expressions), age, and ethnicity, to suggest music playlists that resonate with the user's mood and personal characteristics.	• FER2013 • JAFFE, CKPlus, KDEF, AffectNet • Music Database • Pre-trained ImageNet	• Multi-layer Convolutional Neural Network • VGG-16 • ResNet50 • Haar Cascade

No.	Feature Name	Feature Definition	Datasets	Models
5	Distinguish between blinking and sleeping	• Blinking is a short, frequent, and involuntary closure of the eyelids. It typically lasts for a fraction of a second and occurs regularly while a person is awake. • Sleeping refers to a longer-term closure of the eyes, often accompanied by other indicators.	• IBUG 300-W	• Histogram of Oriented Gradients (HOG) + linear SVD • Built-in Haar cascade algorithm in OpenCV • Dlib Facial Landmark Detector - 68 points model - 194 points model • Multitask cascaded convolutional networks (MTCNN)
User (Application Features)				
6	Notify with sounds when getting sleepy	When drowsiness is detected, the application will alert the user either with upbeat music (if music recommender mode is on) or with a regular alert sound (if music recommender mode is off).	-	-
7	Notify with a flashy screen when getting sleepy	When drowsiness is detected, the application will alert the user with an on-off flashy screen or phone's flashlight.	-	-
8	Keep it on-screen	Minimizes the application screen if the user decides to navigate to other applications (e.g., Google Maps) while using it.	-	-
9	Low battery/resource consumption	Stores the data on a server to use minimal power, extend battery life, and reduce resource consumption.	-	-
10	Adjustable alert volume	Allows the user to control the alert volume with a minimum volume threshold for safety.	-	-
11	Setup working schedule	Enables the user to set working days and shift-end times to receive notifications for session reminders.	-	-
12	Periodically trigger a short alarm	Allows the user to have the application trigger a small alert at set intervals, even if drowsiness is not detected.	-	-
13	User Authentication	Ensures secure user login and prevents unauthorized access. Supports email, password, and biometric authentication for added security.	-	-
14	Data Encryption	Encrypts sensitive data stored on-device and during transit to secure user privacy.	-	-

No.	Feature Name	Feature Definition	Datasets	Models
15	Drowsiness Detection Data Storage	Stores model data and user-specific information securely on the server.	-	-
16	Personalized Model Storage	Creates and stores user-specific models for drowsiness detection based on individual data. Models are trained and stored on the server per user using federated learning to improve models without data sharing.	-	-
17	Spotify Linking	Enables the app to access Spotify's music library, user data, and playback controls, allowing personalized recommendations and device management. Requires authorization with specific permissions for user-related actions.	-	-

Table 2.3: Features Definition Summary Table

2.4.2 AI Features

1. **Eye closure detection:**

   The application triggers an alert when eyes start to close in real-time. It monitors eye movement using the Eye Aspect Ratio (EAR). When the eyes begin to close and the EAR falls below a threshold, the system detects potential drowsiness. If the eyes remain closed for a set number of frames, an alert is triggered to warn the user of fatigue or drowsiness, helping prevent accidents or maintain alertness.

2. **Yawn Detection:**

   Yawn detection is a method for identifying drowsiness by analyzing facial landmarks, specifically those around the mouth, to detect mouth opening. This approach serves as an alternative when eye-based drowsiness detection faces challenges, such as when the eyes are not clearly visible, or the user is wearing glasses or other obstructions. In yawning detection, the key feature analyzed is the distance between the upper and lower lip. When the mouth opens beyond a specific threshold, it is flagged as a yawn. This threshold is crucial for distinguishing between normal mouth movements and yawns, with values typically set based on empirical testing.

3. **Music recommendation:**

   Music recommendation system based on facial emotion (or eyes), age, and ethnicity. This feature enables a personalized music recommendation system that analyzes user attributes, facial emotion (or eye region expressions), age, and ethnicity, to suggest music playlists that resonate with the user's mood and personal characteristics. By using advanced deep learning models, the system recognizes emotions through facial expressions, as well as factors like age and ethnicity, to generate a targeted music recommendation. Integrating emotion detection, this system leverages visual data from the user's face to infer their current emotional state (such as happiness, sadness, or anger). Recognizing age and ethnicity, the model adjusts recommendations to match culturally and demographically relevant music genres.

4. **Distinguish between blinking and sleeping:**

   This feature aims to distinguish between blinking and sleeping. Blinking is defined as rapid closure of the eyes in a short period of time (100 - 400 millisecond) while sleeping can be detected by the following proposed ways

   i. Eye closure lasts longer than the normal time for blinking.

   ii. The driver starts to yawn with a higher frequency than normal.

   iii. The driver's head starts to tilt in all directions.

Chapter 3

Proposed Techniques

3.1 System Architecture

After reviewing our scope of work, this section will focus on outlining the System Architecture, providing an overview of the key components and their interactions. A detailed process architecture is presented through a diagram emphasizing the user experience with the application of drowsiness detection and a music recommender system based on facial emotion recognition. The proposed architecture shown in Figure 3.1 aims to provide an optimized user experience through a seamless integration of advanced technologies.

3.1.1 System Architecture of Drowsy Guard

1. **User Interaction:**
   The system starts when a user initiates the application. The application provides two options to the user:

   - **Drowsiness Detection Only:** This mode enables monitoring the user's drowsiness state and alerting them in case of drowsiness.
   - **Drowsiness Detection and Music Recommender:** This mode integrates drowsiness detection with music recommendation based on the user's facial emotion.

   After the user chooses an option, the system uses the device's camera to capture real-time frames of the user's face. These frames serve as input for upcoming processing.

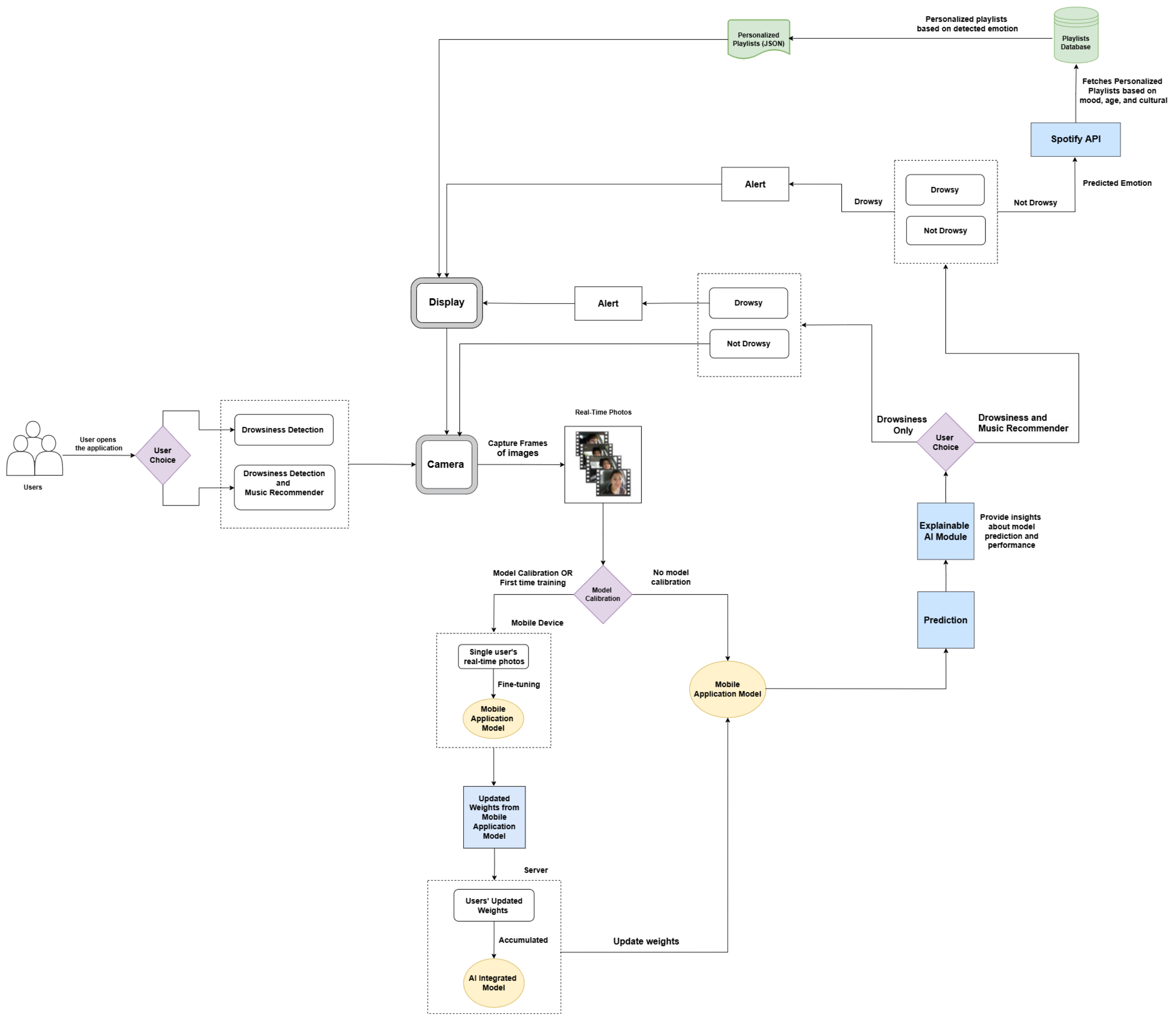

Figure 3.1: System Architecture of Drowsy Guard

2. **Model Calibration and Training:**
   In this phase the system checks whether if there is a model calibration required to be done:
   - *Model Calibration/First-Time Training:* The mobile application model undergoes fine-tuning using real-time photos captured by the user's mobile device. This ensures the AI model adapts to individual features. The user's calibrated weights are sent to the server.
     - **On the server side:**
       The server aggregates the weights from all users, updating the central AI Integrated Model so that the updates weights are sent back to the mobiles' applications model for better generalization across devices.

     This approach is taken to apply federated learning concept to ensure security and privacy by not sending user's sensitive data (frames) to the server. In addition to, leveraging the computational power of decentralized devices, reducing the need for centralized structure and lowering costs.
   - *No Model Calibration:* If the system was previously calibrated, the application directly uses the existing mobile application model without requiring additional training by using the stored updated weights.
3. **Prediction:** The captured input is processed for one of the following tasks depending on the operational mode chosen by the user:
   - ***Drowsiness Detection:***
     The model determines if the user is drowsy by analyzing facial features. Upon detecting drowsiness, an alert notification is generated by the system. The alert could be a sound, vibration, or visual notification to capture the user's attention.
   - ***Detecting Drowsiness and Recommending Music Based on Emotion:***
     If the user is not drowsy, the system classifies the user's emotional state using input frames. This classification sets the stage for personalized music recommendations.
   - ***Explainable AI (XAI) Module:***
     The Explainable AI Module offers insights into model performance, supplying complete explanations of the model's predictions for transparency. This guarantees trustworthiness by explaining how decisions (like drowsiness detection or emotion classification) are made.

4. **Music Recommender System:**
When a user opts for the music recommender mode and is detected as not drowsy, the system integrates with the Spotify API to retrieve a custom music playlist tailored to their emotional state. The process follows a predetermined sequence to ensure the user receives a playlist that matches their mood, preferences, and demographic characteristics.

   (a) **Emotion Detection:**
   The system starts detecting the user's emotional state (e.g., happy, sad, neutral) from the user's face expression. Hence, it represents the main input in choosing mood-suitable music.

   (b) **Retrieve Playlists from Spotify:**
   The API will make Spotify search for "happy playlist" or "sad playlist." These playlists will result from certain mood-related keywords that integrate this model with Spotify's comprehensive music library using an API.

   (c) **Personalization based on User Preferences:**
   The playlist can be further refined through data that the system accesses from the user's Spotify account. The system would then extract the user's favorite tracks, artists, genres, and recently played songs. With the inclusion of such information, the system would ensure that the recommended playlists would not only be mood-appropriate but also in line with the individual tastes of the user.

   (d) **Demographic Considerations:** Besides these, the system takes up the user's age, ethnicity, and cultural background that were inputted when setting up the application. This is for making more personalized playlists, making the results even more culturally acceptable and hence satisfactory for users.

   (e) **Data Storage and Result Presentation:** These custom playlists are stored in JSON format for easy retrieval to display quickly within the app. The JSON files store metadata, such as a playlist name, track URLs, artist names, and so on, which may be played instantly. It then presents the user with the personalized playlist, playing music that complements the users' emotions as well as their tastes.

   (f) **Dynamic Adjustments:** It is worth noticing that the user's preferences and emotional state might change, so the system updates recommendations dynamically during subsequent interactions. The recommendation engine of Spotify also fetches tracks according to some parameters: either on the user's favorite artists or genres.

5. **Continuous Monitoring:**
   The system works in an infinite loop: capturing the real-time frames and updating the predictions. Integrating the prediction, alert, and music recommendation mechanisms would make the system responsive to the user's selected mode. Model calibration and periodic updates will further enhance the accuracy of the system for personalized and reliable solutions.

3.1.2 AI Integrated Model Architectures

This section will elaborate in detail on the various architecture that can be used during designing the proposed drowsiness detection and emotion-based music recommendation. After extensive research, two different architectures workflow have been explored: The first one implemented the CNN for both drowsiness and emotion detection is shown in Figure 3.2. In the second architecture shown in Figure 3.3, a CNN to get the emotion analysis has been used along with face landmarks extraction integrated with threshold logics to detect the drowsiness. Each approach presents an optimal balancing of precision, computational performance, and ease of operation.

- **First Approach:**
  This approach presents a method that incorporates the object detection model YOLO and a CNN to detect drowsiness and recommend music based on emotion. Based on the detection of user drowsiness, a system automatically detects the user's state and acts accordingly.
  - ***Input Data Processing:***
    It initiates by acquiring the real-time photos or frames of video from the surroundings of the user. For that, the acquired photos go under processing to identify ROIs in those photos, specifically detecting the user's face. YOLO is used since it is pre-trained with very high accuracy for detecting images in real time. This identified facial region goes for further processing as an input for the next process.
  - ***Drowsiness Detection:***
    The facial ROI is analyzed using a CNN model, which has been fine-tuned on a custom dataset for drowsiness detection. This dataset includes features such as eye closure (for detecting eye drowsiness) and yawning (for detecting mouth drowsiness). Based on CNN's predictions, the system classifies the user as either drowsy or not drowsy.
    If the user is detected to be drowsy, the system generates an alert to prompt the user to regain alertness. If the user is not drowsy, the system

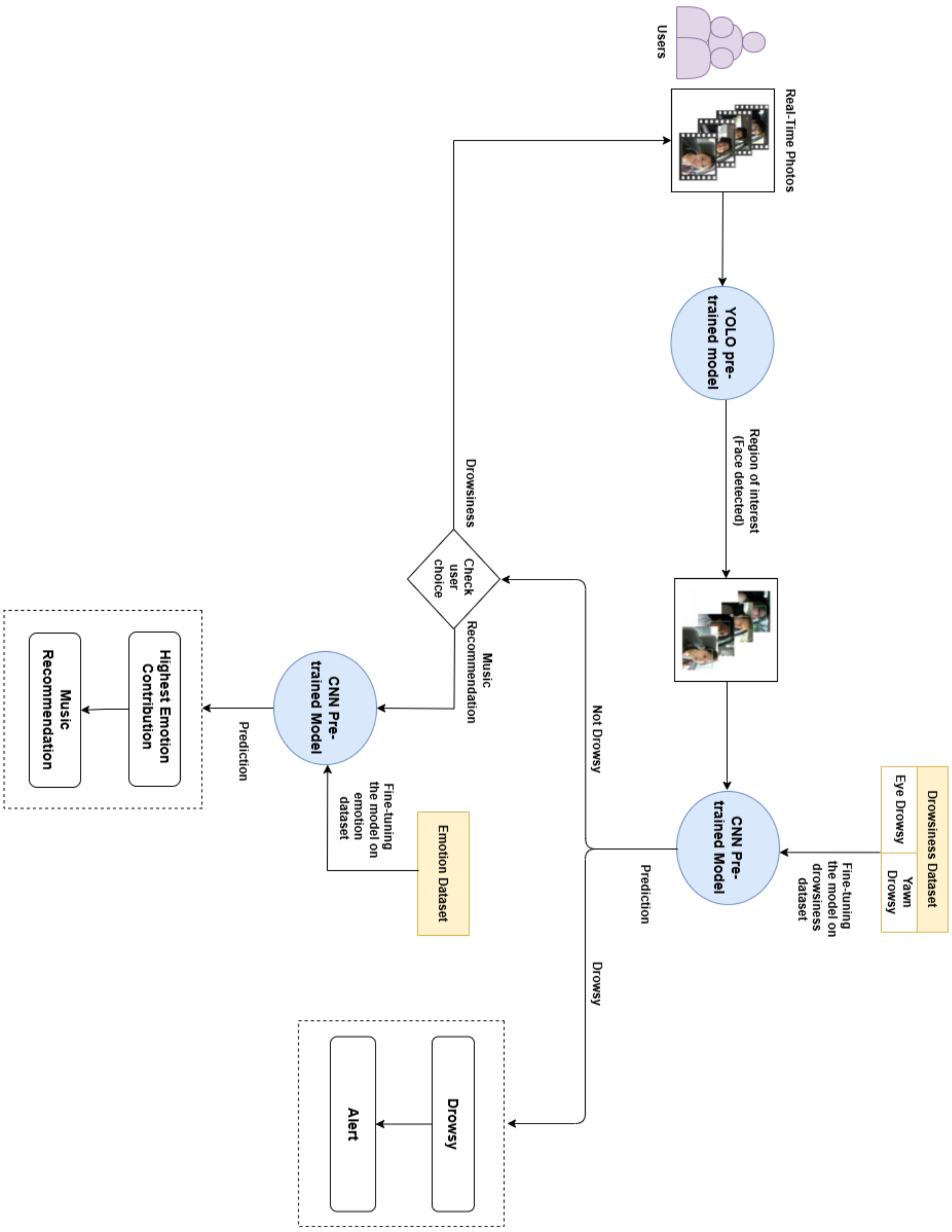

Figure 3.2: First AI Approach Architecture of Drowsy Guard

proceeds to check the user's choice for further interaction. If the user have chosen the drowsiness mode, the system will operate in a continuous loop, capturing real-time frames again.

- ***Emotion-Based Music Recommendation:***
  The input frames classify the user's emotional state in the case of a music recommendation mode. This is done by fine-tuning the CNN on the emotion dataset, allowing the model to set correct classification of the user's emotions. The highest contribution of the emotion is decided and based on that the generation of recommendations of music takes place to match or influence the user's emotional state.

- **Second Approach:**
The second approach further refines the drowsiness detection process by integrating Dlib face landmarks into the pipeline, offering more precise facial feature analysis, while maintaining the CNN-based emotion detection and music recommendation.

  - ***Input Data Processing:***
    First, like in the previous approach, real-time photos or video frames are captured from the user's environment. In such a method, the pre-trained YOLO model is employed for detecting the user's face as the ROI. It will ensure the reliable localization of the facial region to be further analyzed.

  - ***Drowsiness Detection with Dlib68 :*** The detected facial region is processed with Dlib's 68-point face landmark model, which extracts detailed facial features such as eye contours, mouth shape, and jawline. By calculating the distances between specific landmarks, the system measures yawning intensity and eye closure levels. These metrics are compared to predefined thresholds to determine whether the user is drowsy or not.
    If the user is classified as drowsy, the system triggers an alert to notify the user. For non-drowsy users, the system checks for the user's chosen mode for further processing. If the user has chosen the drowsiness mode,the system will operate in a continuous loop, capturing real-time frames again.

  - ***Emotion-Based Music Recommendation:***
    If the user has chosen the music recommender mode, the system applies a CNN pre-trained model finetuned on the emotion dataset. The model predicts the user's dominant emotion, and the system determines the highest emotion contribution. Based on this, it recommends music

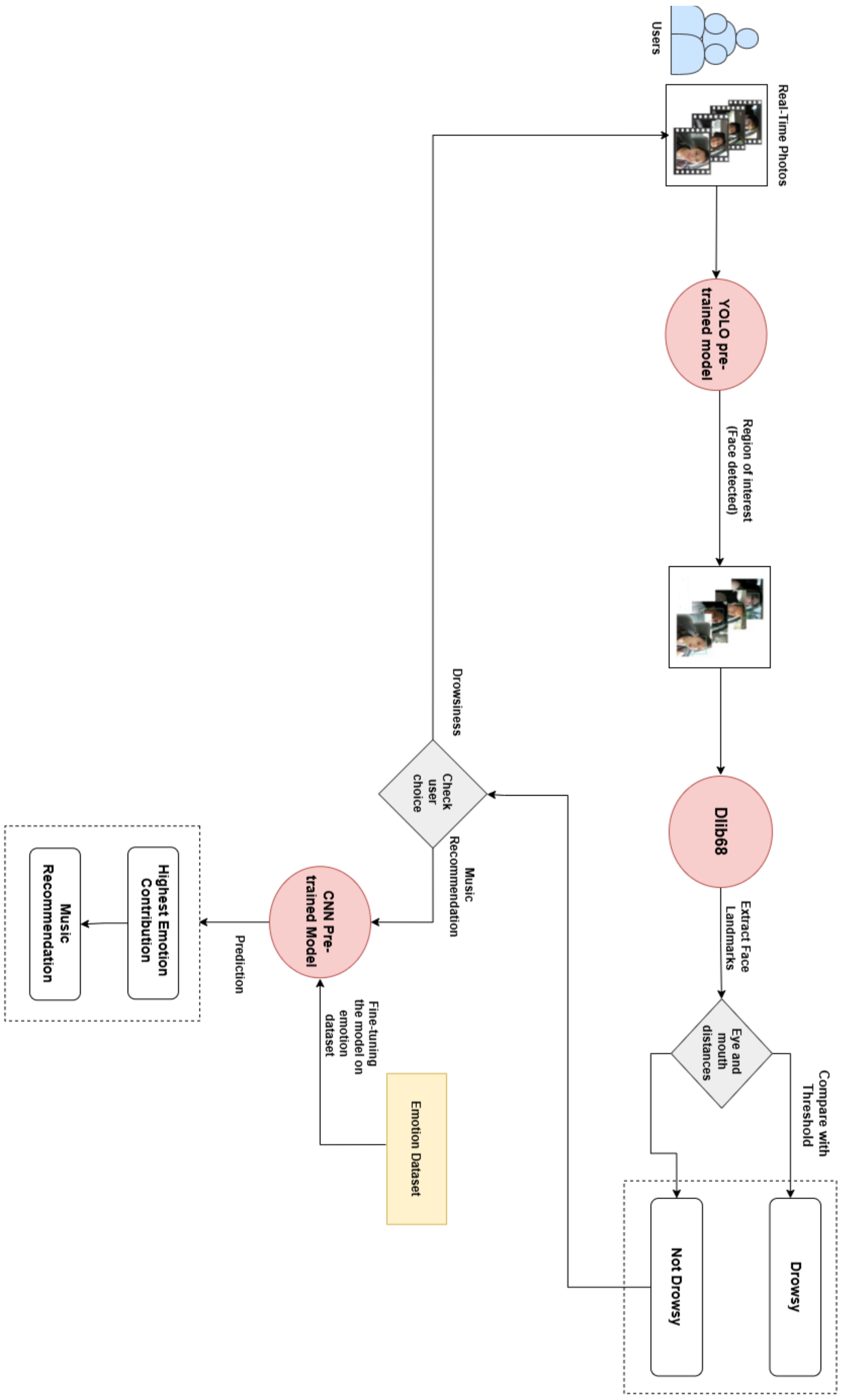

Figure 3.3: Second AI Approach Architecture of Drowsy Guard

tailored to either enhance or complement the detected emotional state.

Comparison of the Two Approaches

Criteria	First Approach	Second Approach
Structure	Focuses on combining face detection with CNN model fine-tuned on a specific drowsiness dataset for accurate classification and personalized interaction.	Focuses on combining face detection with Dlib for facial landmark analysis, for a more precise geometric analysis of facial landmarks.
Complexity	Higher complexity due to reliance on CNN fine-tuning for drowsiness detection.	Simpler implementation for drowsiness detection using geometric face landmarks and thresholds.
Accuracy for Drowsiness	Potentially higher as CNN learns subtle patterns from the drowsiness dataset.	May depend on the precision of landmark detection and thresholds; sensitive to noise in detection.
Flexibility	More flexible for different datasets since it uses a single fine-tuned model for drowsiness.	Limited flexibility as it depends on manually defined thresholds for yawning and mouth distances.
Computational Cost	Higher computational cost due to fine-tuned CNN for both tasks.	Lower computational cost for drowsiness detection due to lightweight Dlib-based approach.
Scalability	Scalable for more tasks by adding additional fine-tuned models.	Less scalable due to reliance on predefined thresholds for specific tasks.
Ease of Integration	Requires training a CNN model, which may increase development time.	Easier to integrate since Dlib landmarks are pre-built and only require threshold logic.
Real-time Suitability	May experience latency due to CNN processing for drowsiness detection.	Likely better suited for real-time processing due to Dlib's efficiency in landmark-based analysis.

Table 3.1: Comparison Between the Two AI Approaches

3.1.3 Pre-trained Models

- **YOLOv8 (You Only Look Once Version 8)**
  YOLOv8 is a pre-trained model known for its efficiency and high performance in real-time object detection tasks. In our project, YOLOv8 has a very crucial role in detecting faces from the real-time frames captured by the mobile device camera for both drowsiness detection and emotion recognition. Fine-tuning the pre-trained model of YOLOv8 for face detection allows it to yield the most accurate and fast identifications of facial regions even in challenging real-world scenarios. This lightweight architecture and detection mechanism will easily run on mobile devices without sacrificing speed or accuracy [33].

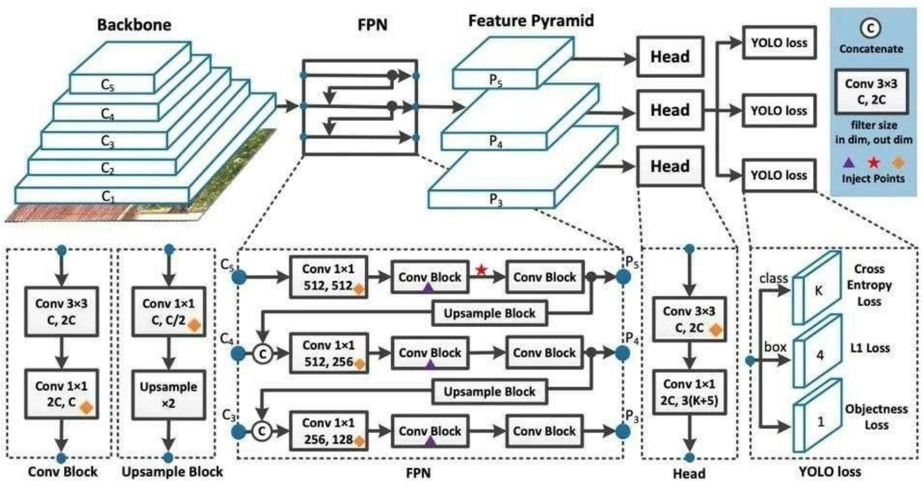

Figure 3.4: YOLO Architecture [11]

The YOLOv8 model combines a backbone network, FPN, and detection heads together for achieving high accuracy with efficiency. The backbone extracts multiscale features in the input image through C1 to C5 convolutional layers. These features are fed to the FPN that fuses the information from different scales, enhancing the model capability of detecting objects of variable sizes. It predicts the bounding boxes, object classes, and confidence scores based on the fused features from FPN with three loss functions: cross-entropy loss for classification, L1 loss for box regression, and objectness loss for detecting objects against the background. The hierarchical architecture assures that YOLOv8 strikes a good balance between speed and accuracy, as required by real-time applications [33].

YOLOv8 is very fast, accurate, and efficient to meet real-time requirements of our application. Since the model is lightweight, it will be effective on mobile devices for supporting on-device computation, ensuring privacy and offline functionality. Its pre-trained capabilities, fine-tuned for our specific use case, minimize substantial effort in dataset preparation while maintaining high detection performance [33].

- **ResNet50 (Residual Network 50)**

  ResNet-50 (Residual Network 50) is a widely-used deep learning model designed for for image analysis, which have been pre-trained on huge datasets like ImageNet, recognizing and classifying as many as thousands of objects. The ResNet-50 model has the property of achieving very good accuracy but staying quiet efficient compared to larger models; thus, it is used in most of the projects that involve image processing and classification. That model is finetunable for particular applications and is supported by most deep learning frameworks, including TensorFlow and PyTorch. We have fine-tuned the model on FER2013 (emotion dataset) and on yawn_eye_dataset_new (drowsiness dataset); the results of the test are given in Table 3.2. [35] [38].

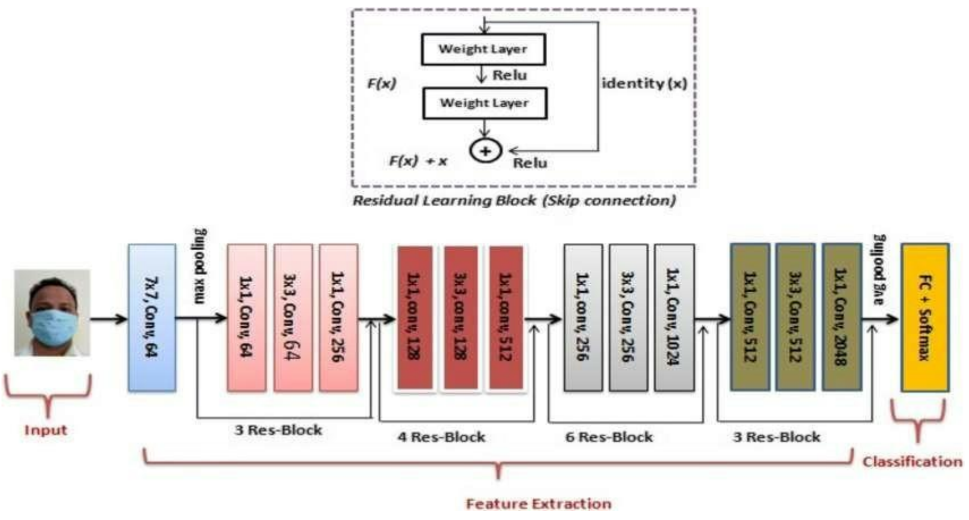

Figure 3.5: ResNET50 Architecture [12]

ResNet-50 utilizes residual learning to address the problem of vanishing gradients in deep networks, enabling the training of very deep models. It is composed of 50 layers and approximately 25.6 million trainable parameters. This makes it powerful enough to handle complex image processing tasks but lightweight enough to be used on a wide range of hardware. The layers are organized into residual blocks, each featuring shortcut connections (skip connections) that add the input of a block directly to its output. This allows the network to learn identity mappings, making it easier to optimize. The architecture begins with an initial convolutional layer and max pooling, followed by several residual blocks with increasing depth (3, 4, 6, and 3 blocks), consisting of 1x1 and 3x3 convolution layers. These layers progressively extract hierarchical features from the input. Finally, the network concludes with a global average pooling layer and fully connected layers, followed by a softmax layer for classification [35] [38].

- **Inception V3**
  Inception v3 is a strong deep learning model often used for image classification and recognition. Because of its high accuracy and efficiency, it is one of the most popular pre-trained models. Inception v3 has been trained on large datasets such as ImageNet where the model can recognize thousands of objects with a fair accuracy. The model is adaptable and can be trained to solve numerous problems making it very suitable for transfer

learning. Balanced design helps image processing perform well without massive computations. So, it can be used in a variety of tasks without requiring excessive computational resources. The model is supported by popular deep learning frameworks such as TensorFlow, PyTorch, and Keras, making it straightforward to use in different workflows. We finetuned the model on FER2013 (emotion dataset) and on yawn_eye_dataset_new (drowsiness dataset) and the test results were recorded in Table 3.2.

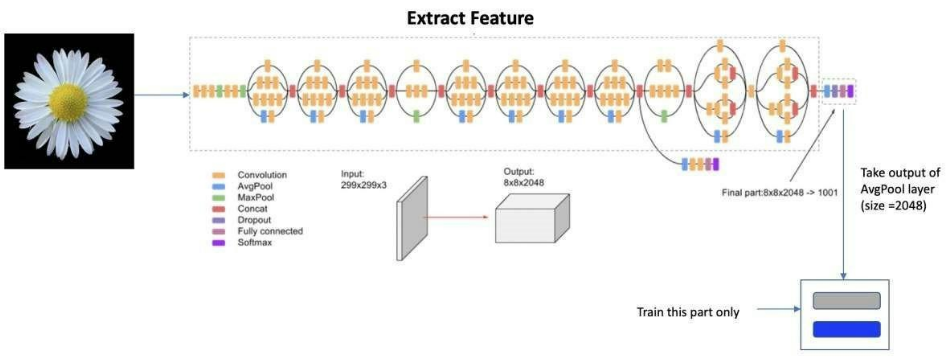

Figure 3.6: Inception V3 Architecture [13]

Inception v3 processes images through multiple parallel operations in order to handle small, medium, and large features of the image. This is a large neural network: more than 23 million trainable parameters. The input starts as an image with dimensions 299 x 299 x 3 color channels that progressively shrink in size, with the model capturing detailed patterns from them. The model summarizes everything that it learns in a compact size of 2048, at the end into a feature vector, which is used for classification tasks. For transfer learning, the model's earlier layers, trained to find general features such as edges and shapes, can be reused by only retraining the final few layers on specific tasks.

- **EfficientNet-B7**
  EfficientNet B7 is a deep learning model for image classification and feature extraction. It aims at high accuracy with optimized computational efficiency. EfficientNet B7 can be used when the recognition of complex images is required in an application. This model has already been pre-trained on massive datasets like ImageNet, and is normally used for transfer learning to various other applications to save time and resources. It is a well-balanced model with depth, width, and resolution thanks to its advanced scaling technique, hence it performs well without wasting computational power. It is supported by the most popular deep learning frameworks such as

TensorFlow and PyTorch, and it can be easily integrated into machine learning pipelines. We finetuned the model on FER2013 (emotion dataset) and on yawn_eye_dataset_new (drowsiness dataset) and test results were recorded in Table 3.2.

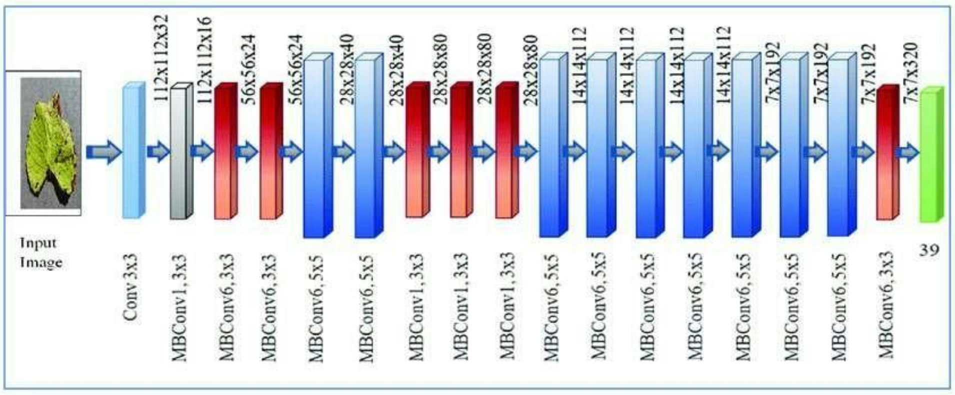

Figure 3.7: Efficient-B7 Architecture [14]

EfficientNet-B7 is a very powerful deep learning model for image classification, having about 66 million parameters; it supports an input image size of 600x600 pixels. It employs a step called compound scaling that helps in balancing depth-how many layers it has-width-how wide the layers are, and resolution-how detailed the input image is. It starts with a simple 3x3 convolution to detect simple patterns, followed by MBConv blocks that efficiently process features, paying attention only to important parts of the image by depthwise and pointwise convolutions. To further improve the accuracy, squeeze-and-excitation blocks have been used to act like attention mechanisms on the most relevant features. Although it is small in size as the image navigates down the model, the level of representation becomes richer as the deeper layers catch more detailed features of patterns. This information is then mapped to a feature map and used for classification.

After reviewing the system architecture and the previous pre-trained model, we moved to phase of testing and training for each model on the drowsiness dataset (Eye Yawn dataset) and Emotion dataset (FER 2013) to obtain the following results recorded in Table 3.2

Model	**Test Accuracy (Yawn Eye Dataset)**	**Test Accuracy (FER2013)**	**Pros**	**Cons**	**Parameters**	**Input Size**
ResNet-50	72.29%	85.71%	- Excellent balance between performance and efficiency. - High accuracy on various tasks. - Ideal for transfer learning. - Widely supported by libraries.	Requires significant computational power, which makes it not recommended for devices with limited processing capacity, such as mobile devices.	25.6 Million	224x224
Inception v3	99.31%	43.62%	- High accuracy in classification tasks. - Efficient compared to many other deep learning models of similar accuracy for its size, allowing it to be used on systems with moderate computational power. - Suitable for transfer learning.	- Computationally intensive for training. - Limited use on edge devices with limited hardware.	23.9 Million	299x299
EfficientNet-B7	98.85%	56.17%	- Exceptional accuracy. - Highly efficient. - Efficient for transfer learning.	- High computational demands. - Requires significant memory and processing power. - Time-consuming fine-tuning, especially on systems without access to GPUs or TPUs.	66 Million	600x600

Table 3.2: Comparison Between Pre-trained Models

According to Table 3.2 and the proposed Test accuracies, we concluded that ResNet-50 will be the most efficient pre-trained CNN model for the music recommendation in emotion recognition with recorded accuracy 85.71%. While Inception V3 is the most efficient pre-trained CNN model for the drowsiness detection with recorded accuracy 99.31%.

3.2 System Algorithms

3.2.1 Dlib-68 Algorithm:

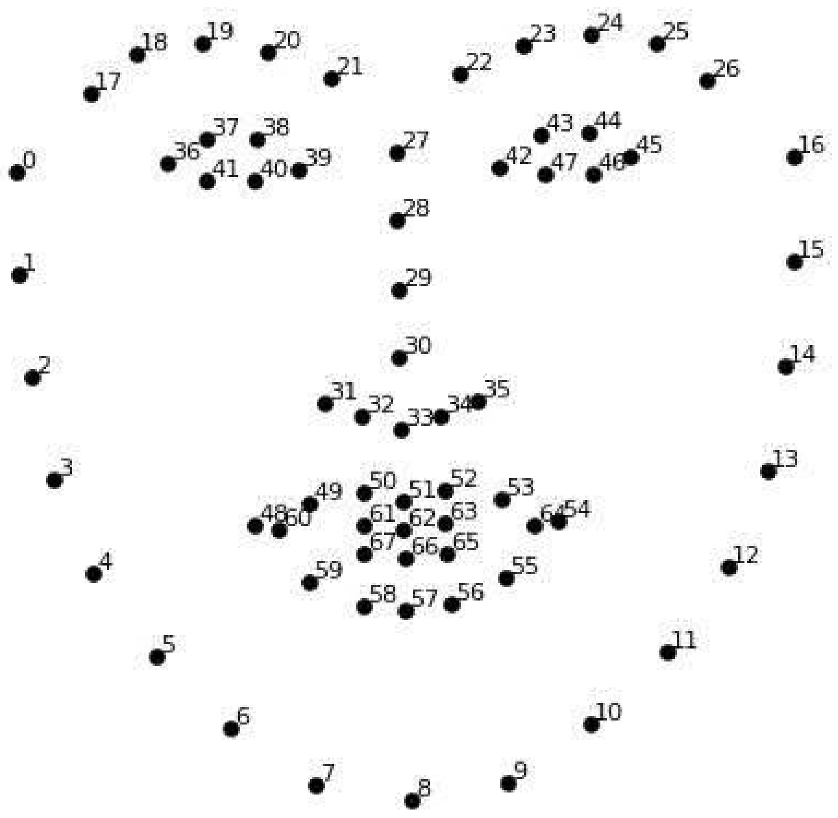

Figure 3.8: Representation of Dlib68 points on the Face [15]

The Dlib-68 algorithm, Implemented within the Dlib library, it detects 68 facial landmarks representing key features like eyes, eyebrows, nose, mouth, and jawline. It is widely used for geometric facial analysis and drowsiness detection.

How the Dlib-68 Algorithm Works

1. ***Input Image Preprocessing:***

   - The algorithm starts by detecting a face within the input image or video frame. Normally, it uses the YOLOv8 CNN-based detector to locate

the face bounding box.
- After detecting a face, the image is cropped to retain only the facial region for reducing computational overhead.

2. ***Facial Landmark Detection:***
   - It uses an Ensemble of Regression Trees model to predict the locations of 68 predefined landmarks on the detected face.
   - The ERT works by iteratively refining the predicted positions of the landmarks based on the pixel intensities around those regions. Each step of regression incorporates knowledge regarding the structure of the face for more accurate predictions.

3. ***Output Landmarks:***
   - The algorithm returns the coordinates (x, y) of the 68 points:
     - Eyes (36–47)
     - Eyebrows (17–26)
     - Nose (27–35)
     - Mouth (48–68)
     - Jawline (1–17)
   - These points can be visualized on the face for further processing or used as input for tasks like drowsiness detection or emotion analysis.

4. ***Post-Processing:***
   - Applications perform geometric calculations on these landmarks. Examples include:
     - **Eye Aspect Ratio (EAR):**
       Measures eye openness to detect drowsiness. Using the Euclidean distance, the distance of the coordinates of the points on the face is calculated to find the eye aspect ratio. Of the 68 facial points, each eye is denoted by 6 (x-y) points with separate coordinates, starting at the left corner of the eye-as if you were looking at the person-and moving clockwise around the rest of the region, as shown in Figure 3.9 [16].

       The formula used for calculating EAR from point p1 to p6:

$$EAR = \frac{||p_2 - p_6|| + ||p_3 - p_5||}{2||p_1 - p_4||} \tag{3.1}$$

       where $p_1, p_2, \ldots, p_6$ are 2D facial landmark locations.

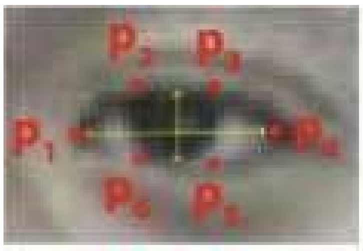

Figure 3.9: Distribution of Points Around Eye in Dlib68 [16].

The numerator provides the distance between the vertical eye landmarks while the denominator calculates the distance between horizontal eye landmarks, with appropriate weighting since there is only one set of horizontal points but two sets of vertical points. The eye aspect ratio remains roughly constant when the eye is open but rapidly falls to zero when a blink occurs. This simple equation then allows us to detect blinks relying on the ratio of eye landmark distances and does not require complex image processing techniques [16].

Figure 3.10, the eye is open with the eye aspect ratio large and constant. When the person blinks, the eye aspect ratio falls considerably, closing in on zero. In order to make a decision on whether or not the eye is closed based on the EAR, a threshold value becomes rather important. Normally, this threshold is taken as that value below which the EAR drops-say, 0.25, that is commonly suggested. This value is sufficient for the correct detection of blinks and extended eye closures, important for driver alertness assessment in the system. Lowering the threshold to about 0.25 further improves the performance across variations in lighting conditions and face orientations. As can be seen from the plot, the variation in the eye aspect ratio as a function of time for a video clip. As can be seen, the eye aspect ratio remains constant, then drops near zero, and rises again, indicating a blink [16].

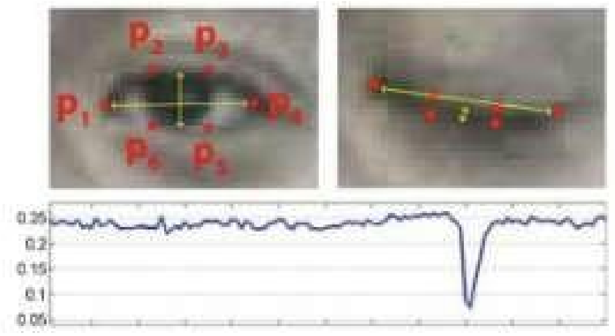

Figure 3.10: Eye-frame rate values graph when eyes are closed and open [16].

- **Mouth Aspect Ratio (MAR):**

  Measures yawning to identify potential fatigue and drowsiness.
  The algorithm calculates the mean positions of the upper and lower lips using the following steps:

  (a) Calculate the Mean Positions of the Outer Upper Lip Landmarks.

$$\text{upper\_mean}_x = \frac{x_{u50} + x_{u51} + x_{u52} + x_{u61} + x_{u62} + x_{u63}}{6} \tag{3.2}$$

$$\text{upper\_mean}_y = \frac{y_{u50} + y_{u51} + y_{u52} + y_{u61} + y_{u62} + y_{u63}}{6} \tag{3.3}$$

  (b) Calculate the Mean Positions of the Outer Lower Lip Landmarks.

$$\text{lower\_mean}_x = \frac{x_{l56} + x_{l57} + x_{l58} + x_{l65} + x_{l66} + x_{l67}}{6} \tag{3.4}$$

$$\text{lower\_mean}_y = \frac{y_{l56} + y_{l57} + y_{l58} + y_{l65} + y_{l66} + y_{l67}}{6} \tag{3.5}$$

  (c) Compute the Vertical Distance Between the Mean Positions

$$\text{distance} = |\text{upper\_mean\_y} - \text{lower\_mean\_y}| \tag{3.6}$$

  The algorithm identifies whether the mouth is open, as seen from the distance between the upper and lower lip landmarks measured from Figure 3.11, as an important indicator for yawning. A sudden increase of this distance depicts a yawn; thus, the driver is suffering from fatigue. Yawning detection is also dependent on a threshold that is normally determined by the vertical distance between upper and lower lip landmarks. While the exact thresholds may vary, experimental values used fall within the range of 0.5 to 1.0. Such an adjustment in threshold allows for correct identification of yawning events, which is considered crucial for driver fatigue assessment within the system due to variations in facial expressions and conditions [17].

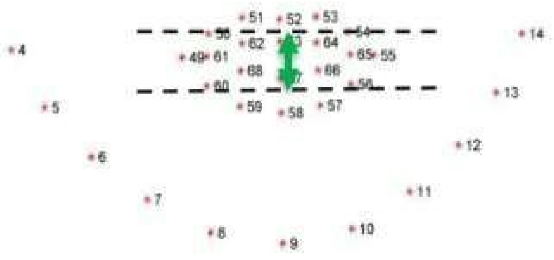

Figure 3.11: Facial landmarks for yawning detection [17].

Dlib-68 algorithm is lightweight and efficient, enabling real-time performance even without a GPU, while it is pre-trained on large datasets such as iBUG 300-W, making it ready for general use without additional training. It's robust under different lighting conditions and facial orientations, and with the straightforward APIs provided by the Dlib library, this algorithm is very versatile and user-friendly for developers.

3.3 Datasets

1. ***Pre-trained ImageNet:*** A large-scale dataset containing over 14 million images across 1,000 object categories, widely used for transfer learning in deep learning models to enhance feature extraction and classification tasks [33].
2. ***iBUG 300-W:*** A facial landmark dataset providing 300 annotated images with precise keypoints for face alignment and pose estimation [36] [37].
3. ***FER2013:*** Contains 48x48 grayscale facial images labeled with 7 emotions (angry, disgust, fear, happy, sad, surprise, neutral). The dataset includes 28,709 training images and 3,589 testing images [39] [38] [40].
4. ***Yawn Eye Dataset:*** Contains labeled images of eyes and mouths for detecting yawning and drowsiness. It includes RGB images in four classes: "Yawn","No Yawn", "Open" and "Closed" with cropped facial features, suitable for safety applications like driver monitoring.
5. ***Driver Drowsiness Dataset (DDD):*** Includes over 41,790 RGB images (227x227 resolution) labeled as "Drowsy" or "NonDrowsy," extracted and cropped.
6. ***Yawn Dataset:*** Contains 5,119 images in color and grayscale across two classes: "Yawn" and "No Yawn." Images are cropped from faces detected via facial recognition.

7. ***JAFFE:***Comprises 213 grayscale facial images from 10 Japanese female subjects showing 7 facial expressions (6 basic emotions + neutral). Each facial expression image in the dataset was reviewed by 60 human evaluators, who rated or labeled the image based on their perception of the displayed facial emotion [30].
8. ***CKPlus:*** Includes 920 grayscale, face-cropped images resized to 48x48 pixels, labeled with 8 emotions (anger, disgust, fear, happiness, sadness, surprise, neutral, contempt). Split as 80% training, 10% public test, and 10% private test [30].
9. ***KDEF:*** Contains 4,900 images of 70 individuals displaying 7 facial expressions (happy, angry, afraid, disgusted, sad, surprised, neutral). Each image is available in color and grayscale [30].
10. ***FDDB (Face Detection Dataset and Benchmark):*** Includes 5,171 labeled faces of varying resolutions (e.g., 363x450, 229x410), with challenging conditions like pose variations, low resolution, and occlusions [41].
11. ***LFW (Labeled Faces in the Wild):*** Features 13,233 images of 5,749 individuals (250x250 resolution) designed for unconstrained face recognition, with pre-defined train-test splits.
12. ***WIDER FACE:*** Contains 32,203 images with 393,703 labeled faces across diverse environments and challenges like occlusion, pose, and scale variations.
13. ***CASIA-WebFace:*** Includes 494,414 face images of 10,575 real identities, used for face verification and identification tasks.

3.4 Visualization

To effectively visualize data related to drivers and cars, utilizing mobile applications is the most straightforward and efficient approach. Mobile applications are widely used for several key reasons.

3.4.1 Mobile Application

Mobile applications have become an integral part of our daily lives, providing a seamless and efficient way for users to interact with various services and perform tasks on the go.

3.4.2 Mobile Application Benefits:

. **Convenience and Accessibility:** Convenience and Accessibility: Mobile apps have made certain services or information accessible from any part of the world. Besides their more portable nature, it's generally easier to use than desktop computers at any given time and location.

. **User Engagement:** User Engagement: It allows them to develop personal engagement opening direct channels, which help in maintaining users.

. **Integration with Device Features:** Integration with Device Features: Applications make use of certain device features like GPS, cameras, and sensors to make it further functional and user-friendly. Push Notifications: Apps may send notifications regarding any upcoming promotions, changes, or anything important to keep the user in the know. Generally, interactive features are more available when compared to mobile websites in applications.

. **Faster Loading:** Faster Loading: Since all data is stored in them locally, it takes considerably less time to load a mobile app compared to web applications that store all their data on remote servers.

. **Increased Functionality:** More Functional: By using the hardware and software of the device itself, applications can offer certain functionalities that are not available to web applications.

. **Offline Accessibility:** Applications could save data on the local device, making complete/ full access to certain features and users available even without being connected to the internet.

. **Data Protection:** It is observed that mobile apps provide better protective measures towards the security of user data than mobile websites.

3.4.3 Utilizing Flutter for Mobile Application Development

To visualize data in a way that is easily accessible to all customers, we considered multiple approaches, including mobile applications and web pages (desktop). We decided to use a mobile application for its convenience and widespread usage [42].

- **Why Use Flutter?**
  Mobile applications have become an integral part of our daily lives, providing a seamless and efficient way for users to interact with various services. However, the challenge arises when we need to develop applications for both Android and iOS platforms. This is where cross-platform mobile

application development with a single codebase becomes advantageous . It is an open-source mobile application development framework created by Google. It uses the Dart programming language to build high-performance, cross-platform mobile applications with a single codebase. Flutter allows developers to build beautiful, fast, and responsive apps for both Android and iOS platforms. Flutter uses a unique approach to building user interfaces called widgets. Widgets are the building blocks for creating user interfaces, and they can be combined and customized to create complex layouts and designs. Flutter provides a rich collection of customizable widgets that can be easily integrated into an application's user interface. Flutter also includes a hot reload feature, which allows developers to quickly see changes they make to their code in real-time. This feature significantly speeds up the development process, as developers can iterate quickly and see the results of their changes instantly [42].

- **Advantages of Using Flutter: [42]**
  - **Single Codebase for Multiple Platforms:** One codebase supports both iOS and Android.
  - **Fast Development:** The hot reload feature of Flutter which is able to show the modifications made in code almost instantly, without the need to reload application state.
  - **High Performance:** Flutter framework turns your mobile app code into native code directly, which helps in making your app very performant.
  - **Expressive and Flexible User Interface (UI) :** Flutter offers a wide range of customizable widgets that follow the rules of Material Design and Cupertino (iOS).

3.4.4 Flutter Development

- **Setting Up Flutter:**
  1. Download and install the Flutter SDK from the official website.
  2. Set up the code editor (like Visual Studio Code or Android Studio) to install the Flutter and Dart plugins.
  3. Start flutter project: In order to create a brand new application, you can use the command flutter create my_app via Flutter CLI.

- **Project Structure:**
  - `lib/main.dart`: Where he main entry point for a Flutter application is located.
  - `pubspec.yaml`: The file has the application metadata and dependencies.

- `android/`: Houses Android-specific code and configuration files.
- `ios/`: The folder that contain tools specified just for iOS.

- Building the User Interface:

- **Widgets:** They are the building blocks of a Flutter app contributing in formation of the screen.
- **Stateful Widgets:**: They are stateful in nature and can change state while they are in memory. Stateless widgets cannot change state.
- **Layouts:** They help organize other widgets on the screen and are of various types like Row, Columns, Stacks, and Containers.

References

[1] Deepinstinct, "Deep learning glossary." [Online]. Available: https://www.deepinstinct.com/glossary/deep-learning

[2] "Applied deep learning: Artificial neural networks." [Online]. Available: https://towardsdatascience.com/applied-deep-learning-part-1-artificial-neural-networks-d7834f67a4f6

[3] F. Technologies, "Neural network quantization: Techniques and applications," accessed: January 5, 2025. [Online]. Available: https://www.flavius.io/media/neural-network-quantization

[4] Manav, "Understanding convolutional neural networks," 21 Nov, 2024. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/

[5] S. Pokhrel, "Beginner's guide to understanding cnns," sep 19, 2019. [Online]. Available: https://towardsdatascience.com/beginners-guide-to-understanding-convolutional-neural-networks-ae9ed58bb17d

[6] A. Vidhya, "Convolutional neural networks (cnn): Understanding key concepts and applications," 2021, accessed: January 5, 2025. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/

[7] Coursesteach, "Computer vision (part 30)-correlation vs convolution." [Online]. Available: https://shorturl.at/opVUu

[8] Bharat, "What is computer vision in 2025? a beginners guide," 2023. [Online]. Available: https://opencv.org/blog/what-is-computer-vision/

[9] GeeksforGeeks, "What is transfer learning?" 2024. [Online]. Available: https://www.geeksforgeeks.org/ml-introduction-to-transfer-learning/

[10] G. AI, "Federated learning for machine learning models." [Online]. Available: https://gemmo.ai/federated-learning/

[11] M. Li and L. Zhang, "Deep learning-based license plate recognition in iot smart parking systems using yolov6 algorithm," *International Journal of Advanced Computer Science and Applications*, vol. 14, 01 2023. [Online]. Available: https://shorturl.at/gSyho

[12] R. Kumar Bania, "Ensemble of deep transfer learning models for real-time automatic detection of face mask," *Multimedia Tools and Applications*, vol. 82, pp. 1–23, 02 2023. [Online]. Available: https://shorturl.at/lDsEc

[13] S. Ramaneswaran, K. Srinivasan, D. Vincent P M, and C.-Y. Chang, "Hybrid inception v3 xgboost model for acute lymphoblastic leukemia classification," *Computational and Mathematical Methods in Medicine*, vol. 2021, pp. 1–10, 07 2021. [Online]. Available: https://shorturl.at/b5xhY

[14] P. Kaur, S. Harnal, D. R. Tiwari, S. Upadhyay, S. Bhatia, A. Mashat, and A. Alabdali, "Recognition of leaf disease using hybrid convolutional neural network by applying feature reduction," *Sensors*, vol. 22, p. 575, 01 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/2/575

[15] P. Korshunov and S. Marcel, "Speaker inconsistency detection in tampered video," 09 2018, pp. 2375–2379. [Online]. Available: https://ieeexplore.ieee.org/document/8553270

[16] T. Hien, Q. Liang, and N. Linh, *Design Driver Sleep Warning System Through Image Recognition and Processing in Python, Dlib, and OpenCV*, 05 2021, pp. 386–393. [Online]. Available: https://www.researchgate.net/publication/351519224_Design_Driver_Sleep_Warning_System_Through_Image_Recognition_and_Processing_in_Python_Dlib_and_OpenCV

[17] A. Bhavana and D. N. Sivakumar, "Real-time driver drowsiness detection using eye closure and yawn detection using facial landmarks," 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:245149380

[18] Geotab Blog, "Drowsy driving statistics," 2024. [Online]. Available: https://www.geotab.com/blog/drowsy-driving-statistics/

[19] K. B. M. T. I. J. D. C. M. T. I. Megan Hoffer MM MT-BC, Julie Avirett MM MT-BC, "How music affects your mind, mood, and body," 2022. [Online]. Available: https://www.tmh.org/healthy-living/blogs/healthy-living/how-music-affects-your-mind-mood-and-body

[20] International Association of Oil and Gas Producers (IOGP), “Driver fatigue safety.” [Online]. Available: https://www.iogp.org/workstreams/safety/safety/driver-fatigue/

[21] Augmented Startups, “Drowsiness detection systems: A comprehensive guide,” 2023. [Online]. Available: https://www.augmentedstartups.com/blog/drowsiness-detection-systems-a-comprehensive-guide

[22] I. H. Sarker, “Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions,” *SN Computer Science*, vol. 2, no. 5, p. 420, 2021. [Online]. Available: https://doi.org/10.1007/s42979-021-00815-1

[23] K. O’Shea and R. Nash, “An introduction to convolutional neural networks,” 2015. [Online]. Available: https://arxiv.org/abs/1511.08458

[24] O. A. Montesinos López, A. Montesinos López, and J. Crossa, *Fundamentals of Artificial Neural Networks and Deep Learning.* Cham: Springer International Publishing, 2022, pp. 379–425. [Online]. Available: https://doi.org/10.1007/978-3-030-89010-0_10

[25] S. R. Dubey, S. K. Singh, and B. B. Chaudhuri, “Activation functions in deep learning: A comprehensive survey and benchmark,” 2022. [Online]. Available: https://arxiv.org/abs/2109.14545

[26] A. L. Davila, “Neural networks: The backpropagation algorithm,” n.d., math 400, College of William and Mary, Professor Chi-Kwong Li. [Online]. Available: https://cklixx.people.wm.edu/teaching/math400/Annette-paper.pdf

[27] F. Mehmood, S. Ahmad, and T. K. Whangbo, “An efficient optimization technique for training deep neural networks,” *Mathematics*, vol. 11, no. 6, 2023. [Online]. Available: https://www.mdpi.com/2227-7390/11/6/1360

[28] R. Yamashita, M. Nishio, and R. Do, “Convolutional neural networks: Overview and application in radiology,” *Insights Imaging*, vol. 9, pp. 611–629, 2018. [Online]. Available: https://link.springer.com/article/10.1007/s42979-021-00815-1

[29] IBM, “What is computer vision?” 2021. [Online]. Available: https://www.ibm.com/think/topics/computer-vision

[30] S. B. Punuri, S. K. Kuanar, T. K. Mishra, V. V. R. M. Rao, and S. S. Reddy, “Decoding human facial emotions: A ranking approach

using explainable ai," *IEEE Access*, vol. 12, pp. 186 229–186 245, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10705307

[31] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," *International Journal of Computer Vision*, vol. 128, no. 2, p. 336–359, Oct. 2019. [Online]. Available: http://dx.doi.org/10.1007/s11263-019-01228-7

[32] K. Martineau, "What is federated learning?" 2022. [Online]. Available: https://research.ibm.com/blog/what-is-federated-learning

[33] M. Andrean, G. Shidik, M. Naufal, F. Zami, S. Winarno, H. Al Azies, and P. Putra, "Comparing haar cascade and yoloface for region of interest classification in drowsiness detection," *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 8, p. 272, 01 2024. [Online]. Available: https://ejurnal.stmik-budidarma.ac.id/index.php/mib/article/view/7167

[34] B. Rajesh, V. Keerthana, N. Darapaneni, and A. R. Paduri, "Music recommendation based on facial emotion recognition," *ArXiv*, vol. abs/2404.04654, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:269004653

[35] "Music recommendation based on face emotion recognition," vol. 2, p. 1–11, Jun. 2021. [Online]. Available: https://jieee.a2zjournals.com/index.php/ieee/article/view/45

[36] Y. Lu, "Real-time eye blink detection using general cameras: a facial landmarks approach," *International Science Journal of Engineering Agriculture*, vol. 2, pp. 1–8, 10 2023. [Online]. Available: https://www.researchgate.net/publication/374379383_Real-time_eye_blink_detection_using_general_cameras_a_facial_landmarks_approach

[37] M. Jain, "Real-time driver drowsiness detection using computer vision," *International Journal of Engineering and Advanced Technology*, vol. 11, pp. 109–113, 10 2021. [Online]. Available: https://www.ijeat.org/wp-content/uploads/papers/v11i1/A31591011121.pdf

[38] R. Mammadli, H. Bilgin, and A. C. Karaca, "Music recommendation system based on emotion, age and ethnicity," 2022. [Online]. Available: https://arxiv.org/abs/2212.04782

[39] M. Athavle, “Music recommendation based on face emotion recognition,” *Journal of Informatics Electrical and Electronics Engineering (JIEEE)*, vol. 2, pp. 1–11, 06 2021. [Online]. Available: https://shorturl.at/LTN6i

[40] J. Maddala, “Music recommendation system based on facial expressions by using cnn,” *International Journal for Research in Applied Science and Engineering Technology*, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:268808759

[41] M. Shakeel, N. Bajwa, A. Anwaar, A. Sohail, A. Khan, and H. Ur Rashid, *Detecting Driver Drowsiness in Real Time Through Deep Learning Based Object Detection*, 05 2019, pp. 283–296. [Online]. Available: https://shorturl.at/x5wFW

[42] S. Kapase, R. Hande, S. Teke, Y. Dhumane, and P. Rajhans, “Driver drowsiness detection system using google ml kit and flutter,” *International Research Journal of Engineering and Technology (IRJET)*, vol. 10, no. 5, pp. 141–145, 2025. [Online]. Available: https://www.irjet.net/archives/V10/i5/IRJET-V10I5141.pdf

[43] N. P. Mankale, “Music recommendation system based on facial emotions,” *International Journal of Research Publication and Reviews*, vol. 4, no. 10, pp. 73–79, 2023. [Online]. Available: https://ijrpr.com/uploads/V4ISSUE10/JRPR18073.pdf

[44] U. Author, “Title of the paper,” *International Journal for Research in Applied Science and Engineering Technology (IJRASET)*, vol. 10, June 2022. [Online]. Available: https://www.ijraset.com/best-journal/driver-drowsiness-detection-using-smartphone

[45] J. Gill and R. Scholar, “A review: Driver drowsiness detection system,” 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:212589831

[46] M. A. Assari and M. Rahmati, “Driver drowsiness detection using face expression recognition,” in *2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, 2011, pp. 337–341. [Online]. Available: https://ieeexplore.ieee.org/document/6144162

Appendix A

Appendix

A.1 Detailed Literature Review

Paper	Research Objective	Problem Addressed	Findings	Limitations	Future Research	AI Features	Datasets
Music Recommendation System Based on Facial Emotions [43]	Music recommendation system based on facial emotions.	Focusing on the human physiological signals using sensors.	To recognize a variety of emotions along with HAAR Cascades for the detection of the lips and eyes.	Reliance on users to manually capture their own images.	The system collects GSR and PPG data from the user. Sensors could be used.	Facial recognition (haar-cascade classifier used and CNN)	Kaggle dataset FER2013
2212.04782 [38]	Music recommendation system that personalize playlists based on the user's emotion, age, and ethnicity, using deep learning techniques.	Existing music recommendation systems do not consider multiple user attributes like emotion, age, and ethnicity simultaneously.	User-friendly system effectively combines emotion, age, and ethnicity.	Users of the same ethnicities prefer similar music. Limited emotion and ethnicity classifications.	Mobile-based application including more diverse emotions, ethnicities, and music genres. Enable real-time dynamic updates.	Use of CNN models for emotion, age, and ethnicity detection.	- Kaggle dataset FER2013 - AGE Dataset for age ethnicity detection.
(PDF) Detecting Driver Drowsiness in Real Time Through Deep Learning Based Object Detection [41]	Real time, efficient, and accurate drowsiness detection system based on deep learning that can be deployed on mobile devices and embedded systems.	Most existing computer vision approaches either require high computational power or fail to perform well under various lighting conditions and driver postures.	Drowsiness is detected by localizing open and closed eyes with a high mAP of 0.84. It is computationally efficient, making it suitable for real time deployment on mobile devices.	The evaluation was limited to a small dataset and specific hardware.	Testing on more diverse datasets, optimizing the model for newer smartphones, and integrating additional sensors to improve accuracy.	CNN-based approach using the MobileNet architecture combined with a Single Shot Multibox Detector (SSD) for real-time object detection tasks on mobile devices.	- "FDDB" (Face Detection Data Set and Benchmark) - "YawDD" (Yawning while Driving) - Closed Eyes in the Wild (CEW) - Custom Images Used

Real-time Driver Drowsiness Detection using Computer Vision [37]	Drowsiness detection system for eye closure and yawn.	Drowsiness detection technologies that exist are very costly and not robust.	The model is capable of detecting drowsiness by monitoring the eyes and mouth. There are a lot of other outer factors that could affect the driver's states.The outer factors may be weather conditions, state of the vehicle, time of sleeping and, mechanical data.	The accuracy of the model degrades if the eye frames are not captured clearly due to any kind of obstacles or when the driver is not facing the camera.	The future work of this paper can be focused on the use of outer factors for measuring fatigue and drowsiness.	OpenCV built in HAAR cascades (adaboost).	iBUG 300-W dataset.
Microsoft Word-1418023204093503 Music Recommendation Based on Facial Emotion Recognition.docx [34]	System integrating emotion recognition, music recommendation, and explainable AI.	Existing music recommendation systems often lack personalization based on real-time emotional states.	The system achieved an accuracy of 82% focusing on the eye region (ROI). GRAD-CAM visualization provided insights into the model's decision-making process.	The study focused on seven basic emotions.	Expand the range of recognized emotions. Incorporate (voice, gesture) emotion recognition. Real-time system. Personalized feedback.	ResNet50 model for facial emotion detection. GRAD-CAM for visualizing model decisions. Extraction of the region of interest (ROI) using a Haar cascade classifier.	FER dataset. Collected emotional dataset. Music dataset with labeled emotions.
Driver Drowsiness Detection Using Smartphone [44]	Smartphone user-friendly application used in drowsiness detection. Facial analysis is used to detect drowsiness before and during using the application.	Using electrophysiological devices in drowsiness detection ranges from affordable to very expensive.	Drowsiness detection system that is easy to use, cost-effective, and ensures privacy by storing data on the user's phone. It allows for drowsiness testing both before and during driving. The system achieved an accuracy of about 88.5% using physiological analysis.	Using a car charger to power the smartphone.	Improve the quality of face and eye detection under low-lighting conditions.	- Android Studio 4.4.2 (software tool) - Mobile Vision API	Dataset from Google used by Mobile Vision API.

A Review: Driver Drowsiness Detection System [45]	- Collecting video frames to detect drowsiness. - Lab color technique is applied to each frame. - The FCM technique is to be applied on the face region. - The iris is detected using CHT and classified with an ANN.	Existing methods are limited by their failure to consider individual differences and are not robust to varying lighting.	Non-intrusive methods using camera-based detection are effective for real-world driving situations.	Existing methods require specialized equipment, are not robust to pose variation, and face challenges in low-light conditions.	Advanced sensor technology, integrate with smart cars, and focus on user feedback.	Face tracking, eye detection, lab color space, thresholding, connected components, fuzzy c-means clustering, circular hough transform,ANN	Camera-based Drowsiness Reference for Driver State Classification under Real Driving Conditions (Bin Yang et al., 2010) The study used a large dataset from 90 hours of real road drives.
Driver drowsiness detection using face expression recognition [46]	Hardware system is implemented to detect drowsiness of drivers, especially those diagnosed at the right time to alert.	The lack of sensors in vehicles to assist in low-light conditions.	Infrared light provides simplicity and works independently of environmental lighting. Appropriate response will result in case of glasses or a beard and mustache.	Assumption of stable head position. It may not handle such dynamic variations effectively in facial expressions.	Template-based method can be used for stable and controlled conditions. AI-based techniques are preferred for accurate facial tracking in dynamic environments.	AI methods are not used	-

Table A.1: Literature Review Table