Real-time Facial Emotion Recognition Using Deep Learning on Facial Landmarks with Pose Normalization

Agostara Nicolò Giovanni

Università degli Studi di Milano - Bicocca
n.agostara@campus.unimib.it

Keywords: Facial Emotion Recognition (FER), Facial Landmarks, MediaPipe, Deep Learning, Pose Normalization

1 Introduction

Facial expression recognition (FER) has become a critical element in human-computer interaction, particularly in safety-critical applications like automotive environments. Emotional states are known to significantly affect driving performance [5] and real-time detection of these emotions is essential for improving road safety. Furthermore, due to the dynamic nature of the automotive environment, robustness to different lighting conditions and head pose is crucial.
Recent studies show that driver emotions are a leading contributor to road accidents, making emotion detection an important component of intelligent in-vehicle systems [7].

Emotions can be derived from internal physiological responses, such as electroencephalography (EEG) and galvanic skin response (GSR), or from external cues, including facial expressions, gestures, and speech [7]. However, in non-laboratory settings, the analysis of internal responses presents significant challenges due to the invasive nature of the required sensors. Consequently, in uncontrolled, real-world environments, image-based methods are more commonly employed, despite the potential privacy concerns they raise.

This study proposes a real-time facial emotion recognition (FER) system that utilizes facial landmarks extracted through Google's MediaPipe framework. By leveraging landmark-based deep learning models, the approach achieves a balance between accuracy and computational efficiency. The system is pre-trained on the AffectNet dataset from Kaggle and fine-tuned on a custom dataset specifically designed to account for extreme facial poses and varying lighting conditions. The aim is to develop a computationally efficient and robust pipeline for real-time FER applications, providing a reliable solution to enhance safety.

2 Materials and methods

In this section, a description of the datasets used and the computational methods implemented in this study is provided. We begin with an overview of the Kaggle AffectNet dataset [6], highlighting its features and the preprocessing steps that were undertaken to extract facial landmarks using Google MediaPipe. This dataset serves as a foundation for pre-training both image-based and landmark-based deep learning models. Following this, we describe the fine-tuning and evaluation process using a custom dataset of a single subject specifically designed to include extreme facial poses. This custom dataset is employed to assess the robustness and adaptability of the pre-trained models under challenging conditions.

2.1 Google MediaPipe

MediaPipe is an open-source framework developed by Google for building applied machine learning pipelines. It is designed to process and analyze data in real-time. MediaPipe provides a variety of pre-trained models and tools that can be used for tasks such as object detection, facial landmark estimation, hand tracking, and pose estimation. The framework is highly optimized for performance, making it suitable for deployment on mobile devices, web applications, and embedded systems. It offers a ready-to-use python API and code examples.

FaceDetector model The initial step in any Facial Emotion Recognition (FER) pipeline is face localization within an image. Google's MediaPipe framework offers BlazeFace, a single-shot fully convolutional neural network designed for real-time face detection, optimized for mobile deployment environments [3]. This architecture achieves a balance between speed and accuracy, making it ideal for applications requiring low-latency.

The use of BlazeFace in this study is twofold. First, it is naturally integrated into the pipeline, as the face landmark model presented in section 2.1 relies on it for accurate face detection. Second, for consistency and comparability, BlazeFace is employed to crop faces from the custom dataset 2.5 and to measure inference times for the entire pipeline 4.3.

FaceLandmark Model The FaceLandmark model [2] employs a residual neural network architecture designed to balance high accuracy with efficient inference speed on mobile devices. The network features aggressive subsampling in its early layers, enabling the receptive fields to cover large portions of the input image at early stages. This design allows the network to effectively differentiate between facial features such as eyes and mouth, even in cases of partial occlusions or when the face partially exits the frame. The model directly predicts the 3D coordinates of 478 landmarks from single RGB camera inputs, making it suitable for real-time facial geometry estimation and facial expression recognition tasks.

By efficiently handling large-scale variations and partial occlusions, the model provides a detailed and accurate representation of facial expressions [4].

Training Data MediaPipe FaceLandmark model has been trained using a globally sourced dataset of approximately 30,000 in-the-wild mobile camera images, captured under various sensor types and dynamic lighting conditions. To enhance the robustness of the training process, the dataset is augmented using standard techniques such as cropping and simulating camera sensor noise, along with randomized non-linear parametric transformations to the image intensity histogram. Ground truth annotations for the 468 3D mesh points are created using a combination of synthetic renderings from a 3D morphable model (3DMM) and manually annotated 2D landmarks on real-world images [4].

2.2 Kaggle AffectNet Dataset

The Kaggle AffectNet dataset [6] used in this study is specifically curated for facial expressions in the wild. Unlike traditional laboratory datasets, which are often collected under controlled conditions, AffectNet is composed of images sourced from the internet, representing a wide variety of real-world scenarios. This diversity includes variations in lighting, background, head pose, occlusions, and demographic differences, making it highly suitable for training models that need to generalize well to real-world applications.

The dataset contains a total of 28,175 images, each labeled with one of eight basic emotion categories: disgust, surprise, happy, neutral, fear, sad, contempt, and anger distributed as in figure 1.
Each image has been already cropped around the subject face and has a size of 96×96 pixels.

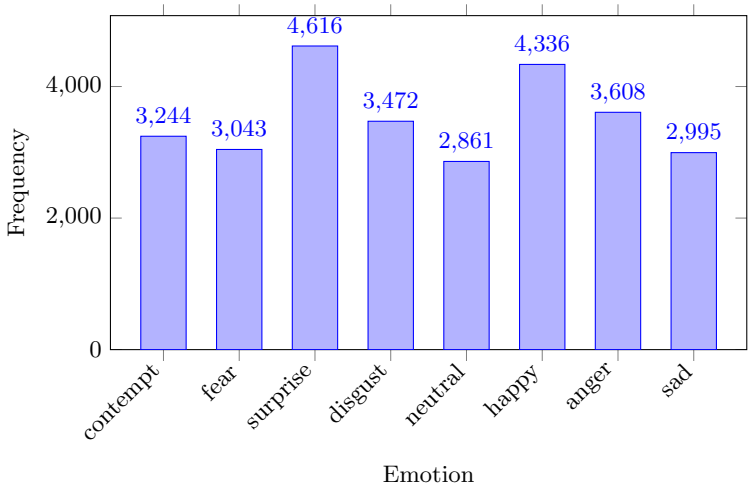

Fig. 1: AffectNet Label Distribution

Even if the dataset if fairly balanced, during training we will mitigate the unbalanced label effect in our loss function by defining a weight tensor based on the inverse frequency of each class. Let c_i be the count of samples in class i. The weight for class i is defined as:

$$w_i = \frac{1}{c_i}$$

To ensure the weights are normalized (i.e., sum to 1), we compute:

$$\tilde{w}_i = \frac{w_i}{\sum_{j=1}^{n} w_j} = \frac{\frac{1}{c_i}}{\sum_{j=1}^{n} \frac{1}{c_j}}$$

where n is the total number of classes.

Creating AffectNet Landmark Dataset: In order to perform the pre-train step of the landmark model described in section 2.4 it is needed to extract landmarks from each image of the AffectNet dataset and save them for later processing. This study heavily rely on the correct functioning of the MediaPipe model, for this reason in this step only outputs with a confidence score greater than 0.9 are considered. This decision will affect the number of training images available, going from 28,175 images to a landmark dataset of 18,379 entries but will also maximize landmarks quality for training data. Another factor affecting the quality of the extracted landmark data is the resolution of the training images from which the landmarks are derived. MediaPipe is optimized for images with a resolution of 256×256 pixels, whereas the AffectNet dataset contains images with a resolution of 96×96 pixels. To address this discrepancy, the images are upsampled to match the required dimensions. However, this process can impact the accuracy and precision of the predicted landmarks.

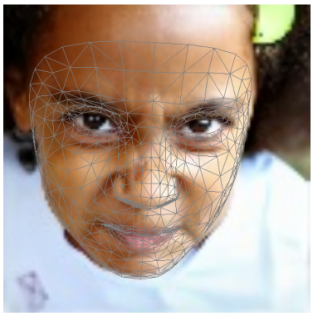

Fig. 2: Example of landmarks extracted from an AffectNet image

2.3 Landmark Normalization Function

The goal of this normalization function is to estimate the head pose and rotate the landmarks of each detected face so that it points forward, ensuring consistent inputs regardless of orientation and scaling. This method increases model robustness to variations in face orientation as shown in section 4.

The Google MediaPipe model, with landmark refinements, outputs 478 points in 3D space, defined by their x, y, and z coordinates.
To normalize the orientation of the detected faces, we utilize an affine transformation that aligns the landmarks of each face to a pre-defined canonical face. The canonical face represents a forward-facing, neutral pose, ensuring that all faces are normalized to a consistent orientation.

Affine Transformation Estimation To estimate the affine transformation, we select a subset of landmarks from both the detected face and the canonical face. In this study, 13 key landmarks are chosen, including points around the outer corners of the eyes, the outer edges of the lips, the center of the nose and face contours as shown by the red dots in figure 3.

Let:

- $L \in \mathbb{R}^{n\times 3}$ represent the detected face landmarks.
- $C \in \mathbb{R}^{n\times 3}$ represent the canonical face landmarks.
- $S \subseteq \{1, \ldots, 478\}$ be the subset of indices corresponding to the selected key landmarks, with $|S| = 13$.

The task is to find an affine transformation $A \in \mathbb{R}^{3\times 4}$ such that the transformation of L_S (the subset of detected face landmarks) aligns as closely as possible with C_S (the corresponding canonical face landmarks). This is formulated as a least-squares problem:

$$A \cdot L_S^T \approx C_S^T$$

where $L_S \in \mathbb{R}^{13\times 4}$ is the subset of detected face landmarks augmented with a column of ones to account for translation, and $C_S \in \mathbb{R}^{13\times 3}$ is the corresponding canonical face subset.

The affine transformation matrix A is estimated by solving the following least-squares system:

$$A = \arg\min_A \|A \cdot L_S^T - C_S^T\|_2$$

Once the affine transformation matrix is estimated, it is applied to the entire set of face landmarks L by computing:

$$L' = A \cdot \begin{bmatrix} L & \mathbf{1} \end{bmatrix}^T$$

where $L' \in \mathbb{R}^{478\times3}$ is the transformed set of landmarks in 3D space. The additional column of ones in L allows the affine transformation to include translation along with rotation. This aligns the detected face with the forward-facing canonical face.

After applying the affine transformation, the resulting landmarks L' are normalized using min-max normalization across each of the x, y, and z axes. This scales the coordinates of each landmark within the range [0, 1] in order to preserve spatial relationships and normalizing each face to a common scale.
In this study we will refer to the above described normalization function as **Proposed normalization function**.

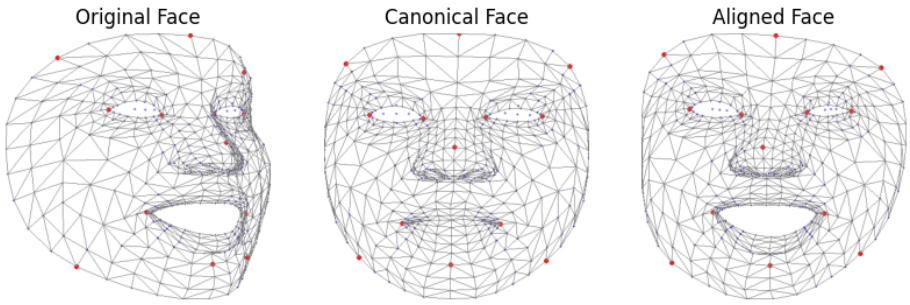

Fig. 3: Affine transformation-based face normalization example, red dots are the ones used to compute the roto-translation matrix

2.4 Selected Model Architectures

For this task, we selected two distinct model architectures with a focus on ensuring real-time processing capabilities. Model size and efficiency were key considerations in our choice. While some implementations, such as the approach outlined in this Kaggle notebook [1], achieve an accuracy of 84% using a Vision Transformer (ViT) on the AffectNet dataset[6], our emphasis is on optimizing for real-time performance while mantaining model's robustness. Both the proposed models have under 1 million parameters.

Two architectures were selected for this study:

- **Baseline Model:** A 2D CNN trained on image datasets. This model serves as a foundational benchmark for comparison.
- **Landmark Model:** A 1D CNN trained on landmark datasets. This model leverages facial landmark data to provide an alternative approach focused on efficient feature extraction.

2.5 Custom Dataset

In this study, a custom dataset was created by simultaneously capturing images and facial landmarks using a keypress-based system. Six emotions were recorded: neutral, happy, sad, angry, surprised, and disgusted. For each emotion, two sessions were conducted—one indoors and one outdoors—to ensure variability in environmental conditions. Additionally, to assess the model's robustness, extreme head poses with significant rotational angles were deliberately included. Each keypress triggers the saving of both the image and the corresponding facial landmarks. This dataset provides both visual and landmark data, which will be used to fine-tune separate models for facial expression recognition (FER): one based on images and another based on landmarks.

In order to train an image model that performs FER, it is needed to crop each image around the subject face. Since MediaPipe uses BlazeFace [3], also the face trim for the custom dataset is made with the same model.

To generate train, validation, and test splits from the dataset, since consecutive frames in a video are highly similar, we extract each split in non-overlapping chunks. For each emotion, the video is divided into three parts: the first portion is used for training, the second for validation, and the third for testing. This ensures that frames within each split are less correlated, reducing the risk of overfitting and improving the generalization of the model.

3 Experiments

The experimental process consists of two distinct phases: a pre-training phase on the AffectNet dataset [6] and a fine-tuning phase using a custom dataset. Below, we detail the experimental setup and process for both phases.

3.1 Training Setup

For both the pre-training and fine-tuning phases, the following configuration was used:

- **Optimizer**: Adam optimizer with an initial learning rate of 0.001.
- **Learning Rate Scheduler**: A linear learning rate scheduler decaying the learning rate by a factor of up to 10 during training.
- **Epochs**: Training was performed for a maximum of 1000 epochs, with early stopping.
- **Loss Function**: Crossentropy loss function was employed throughout the training process.

Performance metrics for evaluating model performance:

- **Accuracy**: $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$
- **F1-Score**: $F1 = 2 \times \frac{Precision \times Recall}{Precision+Recall}$
- **Precision**: $Precision = \frac{TP}{TP+FP}$
- **Recall**: $Recall = \frac{TP}{TP+FN}$

3.2 Landmark Subset Selection and Regularization

During the training of the landmark model with all 478 landmark points, empirical results indicated significant overfitting. To mitigate this, a subset of key landmarks was selected, including the contours of the lips, eyes, irises, and eyebrows. These facial regions exhibit the most prominent changes during emotional expressions.

This subset, combined with regularization techniques such as L2 regularization and dropout, helped stabilize training and validation losses. Figure 4 shows a comparison between the raw detected landmarks and the normalized subset of selected landmarks.

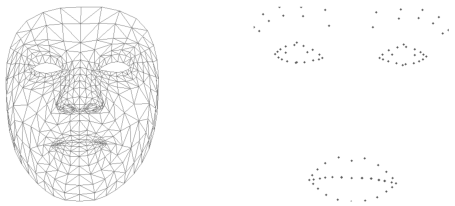

Fig. 4: Comparison between raw landmarks and normalized selected subset

3.3 Pre-training Phase

During the pre-training phase, both models were trained on the AffectNet dataset, which contains a diverse collection of faces in the wild, exhibiting various angles, lighting conditions, races, and facial expressions. This diversity allowed the models to learn generalized features relevant to facial emotion recognition.

The **baseline model** was trained using full image data, while the **landmark model** was trained on a selected subset of facial landmarks. For the landmark model, two normalization techniques were applied: the **Min-Max** and the **Proposed** one (see Figure 3 for an illustration). The performance metrics for both models are discussed in Section 4.

3.4 Fine-tuning Phase on Custom Dataset

In the fine-tuning phase, both models were trained on a custom dataset designed to assess their robustness to extreme facial poses and varying environmental conditions. The same training setup described earlier was used.

For both models, experiments were conducted with two configurations: one with frozen backbone weights and another without freezing the backbone. Additionally, the fine-tuning phase involved experimenting with different batch sizes. For

the landmark model, further tests were carried out using different normalization functions to assess their impact on model performance under challenging conditions.

4 Results

This section presents the results obtained from the pre-training and fine-tuning phases for both the baseline and landmark models in terms of accuracy, recall, f1-score and precision as well as results based on the inference time of both models.

4.1 Pre-training Results

Table 1 shows pre-training result metrics of baseline model while table 2 shows landmark model result metrics with the two different normalization functions.

Metric	Value
Accuracy	0.61
F1-Score	0.57
Precision	0.58
Recall	0.57

Table 1: Baseline Pre-Train Metrics

Landmark Normalization	Accuracy	F1-Score	Precision	Recall
Proposed	0.58	0.55	0.56	0.56
MinMax	0.56	0.53	0.53	0.54

Table 2: Landmark Model Pre-Train Metrics

4.2 Fine-tuning Results

Baseline Model Fine-tuning Results The results of these experiments are shown in Table 3.

Batch Size	Frozen	Accuracy	F1	Precision	Recall
128	True	0.793	0.783	0.806	0.793
128	False	0.857	0.858	0.875	0.857
64	True	0.766	0.761	0.785	0.766
64	False	0.909	0.909	0.918	0.909
32	True	0.77	0.765	0.787	0.77
32	False	0.886	0.887	0.903	0.886
16	True	0.768	0.763	0.79	0.768
16	False	0.846	0.846	0.866	0.846

Table 3: Results of Baseline Model on custom dataset

Landmark Model Fine-tuning Results For the landmark model, Table 4 shows that the proposed landmark normalization function improved performance by an average of more than 10% across all evaluation metrics, while the best model performances are strictly comparable with the baseline model if not better.

Batch Size	Normalization	Frozen	Accuracy	F1	Precision	Recall
1024	proposed	True	0.97	0.97	0.972	0.97
1024	proposed	False	0.931	0.931	0.935	0.931
1024	minmax	True	0.783	0.786	0.821	0.783
1024	minmax	False	0.812	0.816	0.856	0.812
512	proposed	True	0.956	0.956	0.959	0.956
512	proposed	False	0.923	0.922	0.929	0.923
512	minmax	True	0.848	0.849	0.877	0.848
512	minmax	False	0.834	0.833	0.856	0.834
256	proposed	True	0.964	0.964	0.966	0.964
256	proposed	False	0.966	0.965	0.966	0.966
256	minmax	True	0.869	0.869	0.891	0.869
256	minmax	False	0.851	0.851	0.882	0.851
128	proposed	True	0.957	0.957	0.959	0.957
128	proposed	False	0.917	0.917	0.926	0.917
128	minmax	True	0.861	0.86	0.883	0.861
128	minmax	False	0.838	0.839	0.874	0.838
64	proposed	True	0.966	0.966	0.966	0.966
64	proposed	False	0.924	0.924	0.93	0.924
64	minmax	True	0.848	0.851	0.882	0.848
64	minmax	False	0.758	0.759	0.803	0.758

Table 4: Fine-tuning results of the landmark model on the custom dataset. Green highlights the best model using the proposed normalization, and yellow indicates the model trained with min-max normalization.

4.3 Inference Time

In the introduction of this study, the importance of minimizing latency in certain Face Emotion Recognition (FER) pipelines was discussed. Therefore, the performance of both baseline and landmark models is analyzed in terms of inference time.
Initially, the complete inference pipeline of both models is tested. Subsequently, since some steps could potentially be optimized for efficiency, the models are also evaluated with toy input tensors to isolate the inference times of key components.

The platform used for testing inference times is equipped with an M1 Max processor, featuring 10 CPU cores and 32 GPU cores. Both GPU and CPU inference times are shown, with tests conducted using different batch sizes.
Different batch sizes are analyzed because, in order to have a more stable prediction outcome during inference, one might consider not only the best prediction

for each frame but the best prediction in a batch of frames.
For each experiment, the inference time is estimated by first warming up the models, as PyTorch performs certain initialization steps during the first inference, such as memory allocation or kernel compilation. After the warm-up phase, inference is run for 1000 iterations to ensure accurate timing measurements.

The inference time is calculated using the formula:

$$\text{inference time} = \frac{(\text{end time} - \text{start time})}{n}$$

where n is the number of inference runs.
To calculate the frames per second (FPS), which indicates how many inferences can be performed in one second, the formula is:

$$\text{FPS} = \frac{1}{\text{inference time}}$$

Thus, FPS is the reciprocal of the average inference time per frame, providing a direct measure of the model's real-time performance.

To compute the percentage improvement of the landmark model over the baseline model, the following formula can be used:

Let:

- $\text{FPS}_{\text{landmark}}$ be the frames per second (FPS) of the landmark model.
- $\text{FPS}_{\text{baseline}}$ be the frames per second (FPS) of the baseline model.

The percentage improvement in FPS of the landmark model over the baseline model is given by:

$$\text{Improvement } (\%) = \frac{\text{FPSlandmark} - \text{FPSbaseline}}{\text{FPS}_{\text{baseline}}} \times 100$$

This formula calculates the relative increase in performance of the landmark model compared to the baseline model, expressed as a percentage. If the result is positive, the landmark model is faster, while a negative result would indicate the baseline model is faster.

Pipeline Results The inference pipeline is similar for both the baseline and landmark-based models. However, the landmark-based pipeline introduces an additional step of extracting and normalizing facial landmarks.
The following tables present the results of the inference time analysis across different batch sizes and devices, table 5 for CPU and table 6 for GPU.

CPU			
Batch Size	**FPS Baseline Model**	**FPS Landmark Model**	**Improvement**
1	68.39	202.31	195%
10	7.37	169.48	2200%
32	1.25	23.4	1777%
64	0.82	16.58	1916%

Table 5: Pipelines performances on cpu

GPU			
Batch Size	**FPS Baseline Model**	**FPS Landmark Model**	**Improvement**
1	138.77	142.22	2%
10	99.99	151.31	51%
32	34.85	151.27	334%
64	18.62	153.65	725%

Table 6: Pipelines performances on GPU

Raw Models results - Table 7 provides the raw inference times of the baseline and landmark models on the GPU (MPS) while 8 presents performances when inference is made on CPU.

GPU			
Batch Size	**FPS Baseline Model**	**FPS Landmark Model**	**Improvement**
1	955.64	858.83	-10%
10	121.79	803.81	560%
32	37.02	786.25	2023%
64	19.46	751.51	3760%

Table 7: Raw Models performances on GPU

CPU			
Batch Size	**FPS Baseline Model**	**FPS Landmark Model**	**Improvement**
1	108.58	3427.42	3056%
10	8.08	737.47	9022%
32	1.27	25.52	1903%
64	0.84	17.71	2019%

Table 8: Raw Models performances on CPU

These tables enable a detailed comparison of the inference time performance of both models on different processing units and under different batch sizes. Those performances are very high, also in the baseline model, because are measured on a very powerfull machine but, on more resource constrained devices, the improvement percentage holds. By analyzing both the complete pipeline and isolated key components, it becomes clear which model and configuration are most

suitable for real-time FER applications especially where GPU acceleration is not available.

5 Conclusion

In this study, we proposed a lightweight, real-time face emotion recognition (FER) system that utilizes facial landmarks extracted through Google's MediaPipe framework. The system was pre-trained on the AffectNet dataset and fine-tuned on a custom dataset designed for real-world environments with challenging conditions, such as extreme head poses and varying lighting.

The results demonstrate that the landmark-based approach, coupled with a robust normalization technique, achieves performance comparable with traditional image-based methods in terms of accuracy. Specifically, the proposed landmark normalization function improved model performance by approximately 7% across multiple metrics compared to the traditional Min-Max normalization approach.

In addition, the landmark-based model significantly outperforms traditional image-based models in inference speed, making it highly suitable for real-time applications, especially in environments where low-latency is critical, such as automotive systems.

Future work may focus on extending the dataset to include a broader range of subjects and emotions, as well as exploring advanced architectures to further improve model accuracy and robustness in highly dynamic real-world environments. Moreover, there is the potential to extend this approach by incorporating the time dimension. Capturing emotions from a single frame can be challenging; leveraging a sequence of landmark data over time could provide a more accurate and comprehensive representation of emotional states.

References

1. Mohammed Abdeldayem. Facial expression recognition using vit, 2024.
2. Google AI. Mediapipe face landmarker.
3. Valentin Bazarevsky, Yury Kartynnik, Andrey Vakunov, Karthik Raveendran, and Matthias Grundmann. Blazeface: Sub-millisecond neural face detection on mobile gpus, 2019.
4. Yury Kartynnik, Artsiom Ablavatski, Ivan Grishchenko, and Matthias Grundmann. Real-time facial surface geometry from monocular video on mobile gpus. *CoRR*, abs/1907.06724, 2019.
5. Guofa Li, Weijian Lai, Xiaoxuan Sui, Xiaohang Li, Xingda Qu, Tingru Zhang, and Yuezhi Li. Influence of traffic congestion on driver behavior in post-congestion driving. *Accident Analysis Prevention*, 141:105508, 2020.
6. Noam Segal. Affectnet training data. https://www.kaggle.com/datasets/noamsegal/affectnet-training-data, 2023.
7. Huafei Xiao, Wenbo Li, Guanzhong Zeng, Yingzhang Wu, Jiyong Xue, Juncheng Zhang, Chengmou Li, and Gang Guo. On-road driver emotion recognition using facial expression. *Applied Sciences*, 12(2), 2022.