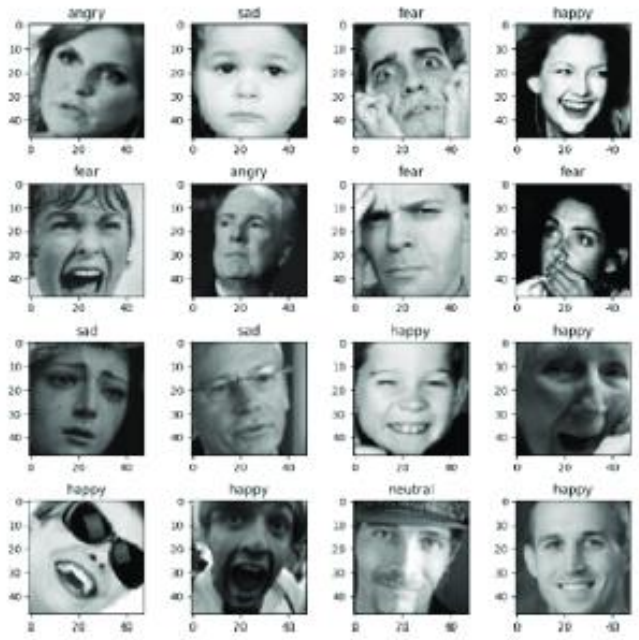
UNIVERSIDAD POLITÉCNICA DE MADRID

Visión por Computador: Entrega 2



Álvaro Pérez-Borbujo Mohedo
Silvia Ochando Valero
Raúl Chamorro Carrasco
José Arturo Morales Morales
Omar Ismel Velázquez Cazales
Delia Martínez Fernández

6 de Noviembre de 2025

Index

1. Project Overview. Objectives and scope of the deliverable.

1.1. Overview.

- **Project name**: Emotion detection.
- **Project duration**: 11.09.2025 - 01.01.2026
- **Project goal**: create a program that detects people's emotions through
- computer vision.
- **Budget**: 300 hours

1.2. Reports Evolution.

No.	Start Date	End Date	Description	Prepared by	Approved by
1	11-09-2025	15-10-2025	1st Deliverable Report	-Silvia Ochando, -Raúl Chamorro, -José Arturo Morales, -Omar Ismael Velázquez, -Delia Martínez, -Álvaro Pérez-Borbujo	
2	15-10-2025	8-11-2025	2st Deliverable Report	-Silvia Ochando, -Raúl Chamorro, -José Arturo Morales, -Omar Ismael Velázquez, -Delia Martínez, -Álvaro Pérez-Borbujo	

1.3. Objectives of the deliverable.

The main objective of this deliverable is to establish the conceptual, technical, and methodological foundation for the project: detection of basic emotions from facial expressions. This includes analyzing the state of the art, defining the tools and datasets to be used, and specifying the requirements and methodology that will guide the project's development. Specifically, this deliverable aims to:

- Review and summarize the background and current state of research in emotion detection using facial expressions.
- Identify and justify the selection of datasets, programming languages, and development environments to be used.
- Define the functional, operational, and performance requirements of the system.
- Select appropriate tools and technologies.
- Present the methodological approach that will be followed during implementation.
- Show the initial progress of the project's web platform.

1.4. Scope of the deliverable.

The scope of this deliverable is limited to the research and planning phase of the project. During this stage, the technical specifications are defined, including the programming environment and language, as well as the **s**election of tools that will be used in the overall development.

In addition, this deliverable covers the study of the background of the selected benchmark and the state of the art related to the problem, analysing other solutions that have already been implemented or developed for the same challenge: the detection of emotions. The chosen methodology will also be documented in this deliverable, outlining the steps and criteria that will guide the subsequent phases of the project.

This stage does not yet include the implementation of the system; its purpose is to build a solid and well-defined foundation that will serve as a reference for future development and experimentation stages.

2. Background of the problem.

Emotion detection is a very important part of Computer Vision and Machine Learning. Historically, the study of emotion detection has been achieved by classifying facial expressions into basic emotions. Over the years, research has been conducted in this field with certain objectives, which are:

- Human-Computer Interaction: To be able to develop interfaces that can respond according to the detected emotional state of the person being interacted with.
- Commerce: To succeed in evaluating the emotional reactions of people towards products or advertisements.
- Security: To be capable of detecting behavior or stress patterns in public environments.
- Mental Health: To monitor the state of patients.

Even with the advancements that have been achieved over the years, the problem of emotion detection remains a very significant challenge due to the great variety of parameters, such as lighting, different face angles, occlusions... Our project is based on developing an accurate system capable of detecting the different basic emotions.

3. State of the art.

Facial Emotion Recognition (FER) aims to identify and classify affective states into emotional categories based on facial images. Convolutional Neural Networks (CNNs) are widely applied for this purpose.

The FER-2013 dataset was introduced for the Facial Expression Recognition Challenge in 2013. It contains 48×48 grayscale images, categorized into seven emotions: Angry, Disgust, Fear, Happy, Sad, Surprise, and Neutral. Dataset size: approximately 35,800 images.

3.1. Methodological Approaches.

Emotion recognition can be addressed through several complementary modalities:

1. Text-based approaches: Analyze messages, tweets, reviews, or conversations to extract emotional tone. Methods range from lexicon- and rule-based models to deep learning architectures such as RNNs**,** LSTMs**,** and Transformers.

2. Image / Facial expression-based approaches: Recognize emotion from static images or video frames by analyzing facial expressions.

3. Audio / Speech-based approaches: Detect patterns of tone, intonation, prosody, and pauses to infer emotion.

3.2. Most Commonly Used Datasets

- **Emotion Detection (FER-2013):** Dataset with 7 emotion categories https://www.kaggle.com/datasets/ananthu017/emotion-detection-fer
- **Emotion Detection from Text:** Predict emotions from textual data (multi-class classification) https://www.kaggle.com/datasets/pashupatigupta/emotion-detection-from-text
- **Facial Emotion Recognition Dataset:** Images of people showing eight emotions https://www.kaggle.com/datasets/tapakah68/facial-emotion-recognition
- **ShEMO (Persian Speech Emotion Detection Database):** Semi-natural emotional speech samples in Persian https://www.kaggle.com/datasets/mansourehk/shemo-persian-speech-emotion-detection-database
- **Text Dataset for Emotion Detection:** Text samples labelled by emotion https://www.kaggle.com/datasets/jarvis11/text-dataset-for-text-emotion-detection

3.3. Main Challenges of the FER-2013 Dataset.

When using FER-2013 for model evaluation, several important challenges arise:

- Class imbalance: Some emotion classes (e.g., *Disgust*) are underrepresented compared to others.
- Variability in conditions: Differences in facial pose, lighting, occlusion, subtle expressions, and diverse backgrounds make recognition significantly harder than in controlled environments.
- Emotional ambiguity: Emotions are not always clearly expressed, and labels may be ambiguous or subjective.
- Low image resolution (48×48): Limits the extraction of fine facial details.
- Generalization: A model performing well on FER-2013 may fail to generalize to other datasets or real-world conditions.

3.4. Best Reported Results for FER-2013 Dataset.

Source: Emotion Detection – October 28, 2021 (Sansjar Hasija) https://www.kaggle.com/code/odins0n/emotion-detection

The model was trained on the FER-2013 dataset. During evaluation, it achieved an overall accuracy of 63.04% and a loss of 1.0577 on the test set. A moderate performance consistent with typical baseline CNN results reported in the literature.

Emotion	**Precision**	**Recall**	**F1-score**	**Observations**
Angry (0)	0.52	0.61	0.56	Acceptable performance; often confused with sadness.
Disgust (1)	0.00	0.00	0.00	Failed to identify this class due to data scarcity.
Fear (2)	0.47	0.34	0.39	Weak recognition; often confused with surprise or sadness.
Happy (3)	0.87	0.86	0.87	Best performance; happiness is visually distinctive.
Sad (4)	0.55	0.69	0.61	Intermediate results; good recall but lower precision.
Surprise (5)	0.51	0.48	0.49	Acceptable; occasionally confused with fear.
Neutral (6)	0.75	0.75	0.75	Good recognition of neutral expressions.

Overall publicated metrics:

- Accuracy: 0.63
- Macro-average F1: 0.52
- Weighted-average F1: 0.62

The obtained results indicate moderate performance, with a general balance between precision (0.62) and recall (0.63).The model performs well for the most evident emotions, such as Happiness and Neutrality, but struggles with underrepresented or visually ambiguous categories like Disgust and Fear.

The main sources of error include class imbalance**,** low image resolution, and visual overlap between similar emotions.

3.5. General Analysis and Lines of Improvement.

An overall accuracy of around 63% suggests that the model performs above random baseline and falls within the expected range for initial CNN-based studies on FER-2013. The relatively high loss (>1) indicates that the model's fit can be further optimized.

However, the model still falls short of the state-of-the-art performances (>80%) reported in recent literature.

Suggested lines of improvement:

1. Data augmentation and balancing: Apply oversampling, synthetic generation (e.g., using GANs), or class weighting to mitigate imbalance.
2. Model architecture enhancement: Experiment with deeper or hybrid architectures such as ResNet, EfficientNet, or Vision Transformers (ViT) to capture more robust facial features.
3. Transfer learning: Use pretrained models on larger face datasets (e.g., VGGFace2) for better feature extraction.
4. Multimodal fusion: Combine image, text, and audio cues to enhance emotional context understanding.
5. Improved preprocessing: Techniques such as face alignment, illumination normalization, and attention mechanisms can boost recognition under variable conditions.
6. Cross-dataset evaluation: Test model generalization across different FER benchmarks to ensure robustness in real-world applications.

4. Technical Specifications.

The dataset, "Emotion Detection" contains 35,685 examples of 48x48 pixel gray scale images of faces divided into train and test dataset. Images are categorized based on the emotion shown in the facial expressions (happiness, neutral, sadness, anger, surprise, disgust, fear) [1].

To handle this dataset, Python will be used as the programming language. Besides being familiar to all team members, this language includes numerous libraries (OpenCV, TensorFlow, Pandas, NumPy, Matplotlib, etc.) that are extremely useful for computer vision. In addition, Python has a highly active community, which facilitates finding potential solutions to programming issues.

Regarding the programming environment, Google Colab will be primarily used, as it allows code execution without the need for local computational resources (GPU, RAM, etc.). In computer vision applications, where execution requirements can be very high, this tool enables switching between different hardware configurations (CPU, GPU T4, and TPU v5e-1) thus reducing execution times.

Visual Studio may also be used in cases where team members have personal computers with high-performance capabilities that can run the code locally. S oftware management within the team will be carried out using GitHub [2]. This platform enables collaboration and version control among different members, allowing the creation of separate code branches for testing without affecting the main one. For documentation, Google Docs and Overleaf will be used.

5. Requirements.

5.1. Functional Requirements.

- The system must process input images by applying classical computer vision techniques using OpenCV to enhance information quality, including noise reduction, edge detection, normalization, etc.
- The system must be capable of classifying basic emotions (happiness, sadness, anger, fear, surprise, disgust, and neutral) both in real-time and in static images.
- The system must include two parallel processing modes:
  - A CNN built from scratch, designed for 48x48 single-channel image inputs.
  - Transfer learning using a pre-trained network such as VGG, ResNet, or MobileNet, adapted for a 7-class output.
  - Both the pre-trained network and the CNN developed from scratch must be fully trainable.
- The system must perform model training as follows:
  - Transfer Learning Training: Initially, only the last layer will be trained (approximately 167,000 parameters).
- The system must load the selected dataset using TensorFlow.

5.2. Operational Requirements.

- The software must run in a Python environment (3.10) or higher.
- The system must use the main libraries: TensorFlow and OpenCV.
- The system must be compatible with GPU execution environments such as Google Colab or systems equipped with Nvidia GPUs.
- The source code must be managed using GitHub for version control and accessibility.
- The system must be executable in Visual Studio Code or Google Colab.

5.3. Performance Requirements.

- The final system accuracy must exceed 85% on test evaluations.

5.4. Non-Functional Requirements.

- The code must be documented and structured in a modular way to ensure better understanding and facilitate future modifications.
- The project must be written and documented using Overleaf and Google Docs, allowing collaborative work.
- The user interface must be intuitive and allow the selection of the model type: CNN or Transfer Learning.
- The results must be clearly displayed, indicating the detected emotion.

6. Selection of tools.

As mentioned earlier, Python will be used for programming. This language includes several useful libraries that will be essential for image processing, namely OpenCV and TensorFlow.

- OpenCV (Open Source Computer Vision Library) is an open-source library designed for computer vision and image processing tasks. It provides a wide range of functions for real-time image and video analysis, such as object detection, face recognition, edge detection, image filtering, and geometric transformations. OpenCV is optimized for performance and widely used in machine learning, robotics, and AI applications.

- TensorFlow is an open-source library developed by Google for machine learning and deep learning applications. It provides tools for building, training, and deploying neural networks across different platforms. TensorFlow supports both CPU and GPU execution, allowing efficient computation for complex models. It is widely used in fields such as computer vision, natural language processing, and predictive analytics, thanks to its flexibility, scalability, and integration with frameworks like Keras.

It is possible that additional libraries specific to computer vision will be added as the project progresses.

In cases where data from the dataset needs to be processed or visualized, the NumPy, Pandas, and Matplotlib libraries are the best options.

To program the code, Google Colab will be primarily used for the reasons mentioned above.The code will be shared in the GitHub repository.

Finally, the documentation will be prepared using Google Docs and Overleaf.

7. Methodology.

After considering various solutions, it was decided to carry out two parallel implementation lines. Both have their advantages and disadvantages.

7.1. CNN.

- Advantages: Total flexibility, allows choosing the inputs to optimize the information the network needs. The number of layers and their properties is chosen, which allows for a balance between the effectiveness and efficiency of the network.

- Disadvantages: Training a network from scratch can be a very complex task requiring a large volume of images and computation time.

7.2. Transfer Learning.

- Advantages: These pre-trained networks are already capable of "identifying features," which means the problem focuses more on adapting the network to the specific case. This saves a lot of time, as the network already has considerable knowledge, and they are models trained very meticulously, so they start with very high accuracy.

- Disadvantages: Its main disadvantage is the lack of flexibility. First, we would only train the last layers (few parameters) to adapt it to our problem, and then we would have to train the entire network to finish fine-tuning (many parameters). Second, this low flexibility is a problem since the network is designed for 224×224 RGB images (three channels).

In most cases, a pre-trained network would be used, as it would normally achieve better results with much less effort. However, the images in the dataset being worked with are smaller (48 x 48) and in grayscale (single channel). Although the images can be adapted to be used with the network (enlarging the image and using it in all three channels), it would be more inefficient and could also clash with a network trained to identify using colour.

Designing our own network would make the most of its structure, requiring fewer parameters, getting the maximum performance out of training, and avoiding the problem of the network identifying by colours. The number of inputs decreases considerably: from 150,528 (224 x 224 x 3) to 2,304. Therefore, both strategies will be followed, and an initial training will be done to decide if one should be discarded over the other or if it is simply worth comparing the two.

7.3. Steps.

Initially, both will follow the same path with slight modifications.

1. Dataset Load: We download the chosen dataset from Kaggle (TensorFlow has a function that allows this without needing other standard libraries like OpenCV).

2. Image Processing: Here, the OpenCV library will be used (this is the most standard and allows all kinds of techniques) to process the images with classic techniques, to remove useless information from the image and highlight the useful, thus giving the network only what is necessary.

3. Network Design:
   - CNN from scratch: using the TensorFlow library, a network is created, and the layers to be used are chosen. The inputs will be 48 × 48 × 1 channel (1 input for each pixel), and the output will be the 7 possible categories. Both input and output values will range from 0 to 1, although in the output, all values must sum to 1.

   - Transfer Learning: here we will simply load the chosen network with TensorFlow. Then we will adjust the last layer (the output) to have 7 values (instead of the default ones). We will not change the input as it is not possible (the network is designed to receive specific parameters).

4. First training: this is unique to the transfer learning network. Initially, be- fore training the entire network, the last layer (the one we modified) must be trained to learn to predict with 7 categories. This reduces the problem to about 167,000 parameters. This first training does not need a long duration as the rest of the network is already prepared to receive images. One point to explain is that to adapt our images to the network's input, we will resize them to 224 × 224 and then use that image in all three channels.

5. Second training: Here, classic training will be done. In this case, the complete transfer learning network will be trained (4.5 million parameters), which will Contents 4 start with a much higher percentage. While for [the TL network] this will be a training to finish adjusting, this will be when the network from scratch learns to identify.

6. Functional Test: Although they are already done during training, once it is finished, a test will be run with all the test images, and with it, an estimate of how good the network is will be obtained.

7.4. Training.

The training sessions will follow the classic pattern of neural networks (the chosen training parameters will be left for later). The dataset already comes separated into test images and training images, so the training images will be used in batches. They are fed into the network, which predicts a category. Depending on how close it is to being correct, the weights are readjusted to improve the prediction.

After each epoch, the training data will be used to test the model, but in this case, the network will not learn from this data, as that would make this data useless for checking how good our model is. This is useful because if the network is seen to not be improving, the training can be stopped.

The choice of parameters such as batch size, epochs, learning rate, etc., will vary according to the available computing capabilities and time limitations.

8. Website progress.

A website [3] has already been created, containing basic information, team members, and the project's objective. More information will be added over time.
The tool used for this is Google Sites.

9. References.

[1]: Emotion detection datasheet- FER- 2013: https://www.kaggle.com/datasets/ananthu017/emotion-detection-fer/data

[2]: Github repository: https://github.com/rchamo01/TrabajoVisionComputador/tree/main

[3]: Website: https://sites.google.com/view/deteccion-de-emociones/p%C3%A1gina-principal?authuser=0

Annex I: Meetings.

Project Information			
Customer	UPM	Name	Work Computer Vision

ASSISTANTS	
Name	Project Position
Silvia Ochando Valero	Management
Raúl Chamorro Carrasco	Technological
Omar Ismael Velázquez Cazales	Documentation
José Arturo Morales Morales	Documentation
Álvaro Pérez-Borbujo Mohedo	Programming
Delia Martínez Fernández	Testing and validation

INFORMACION MEETING

Date	27/10/2025	Start-End	12:00-13:00	Type	Online
Objectives	Define the project specifications				

ITEMS TO BE DISCUSSED
Dataset
Programming language and environment
Website creation

TOPICS DISCUSSED
Arturo y Omar will be in charge of the website creation
Which tool will we use for the website
The programming language will be Python
We will look for a better dataset (since the one that we chose is not good enough)
Decide whether to create a neural network from scratch or modify an existing one.

COMMITMENTS AND AGREEMENTS ADOPTED

No.	Description	Date of Application	Responsible	Expiration Date
0	Look for a better dataset	27/10/2025	Raul and Delia	06/11/2025
1	Create the first version of the website in Google Sites	27/10/2025	Arturo and Omar	06/11/2025
2	Create a new neural network, try an existing one and compare them	27/10/2025	Silvia and Álvaro	06/11/2025

NEXT MEETING AGENDA:	YES	Date	06/11/2025

REMARKS	None relevant
PREPARED BY:	Silvia

Project Information			
Customer	UPM	Name	Work Computer Vision

ASSISTANTS	
Name	Project Position
Silvia Ochando Valero	Management
Raúl Chamorro Carrasco	Technological
Omar Ismael Velázquez Cazales	Documentation
José Arturo Morales Morales	Documentation
Álvaro Pérez-Borbujo Mohedo	Programming
Delia Martínez Fernández	Testing and validation

INFORMACION MEETING					
Date	06/11/2025	Start-End	13:00-13:30	Type	Online
Objectives	Preparation of the Deliverable 2				

ITEMS TO BE DISCUSSED
Task Assignment
Next Meeting Scheduling

TOPICS DISCUSSED
Task Assignment
The next meeting will be the 12/11/2025

COMMITMENTS AND AGREEMENTS ADOPTED

No.	Description	Date of Application	Responsible	Expiration Date
0	Drafting and Completion of Assignment	06/11/2025	All members	08/11/2025
1	Finalization of the project in LaTeX	06/11/2025	Silvia and Álvaro	08/11/2025

NEXT MEETING AGENDA:	YES	Date	12/11/2025
REMARKS	None relevant		
PREPARED BY:	Silvia		