EmotiVision: Emotion detection and gaze analysis of retrieved faces

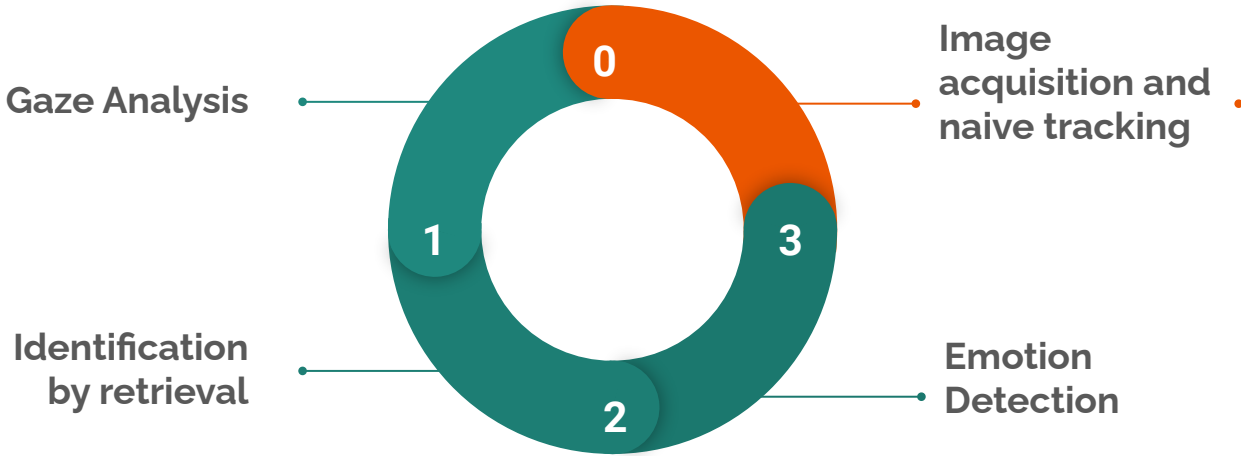
Michele Giarletta, Vincenzo Lapadula, and **Giacomo Salici**
Course of Computer Vision and Cognitive Systems - Final Project

Computer Engineering Master Degree
University of Modena and Reggio Emilia

{273839, 267759, 270385}@studenti.unimore.it

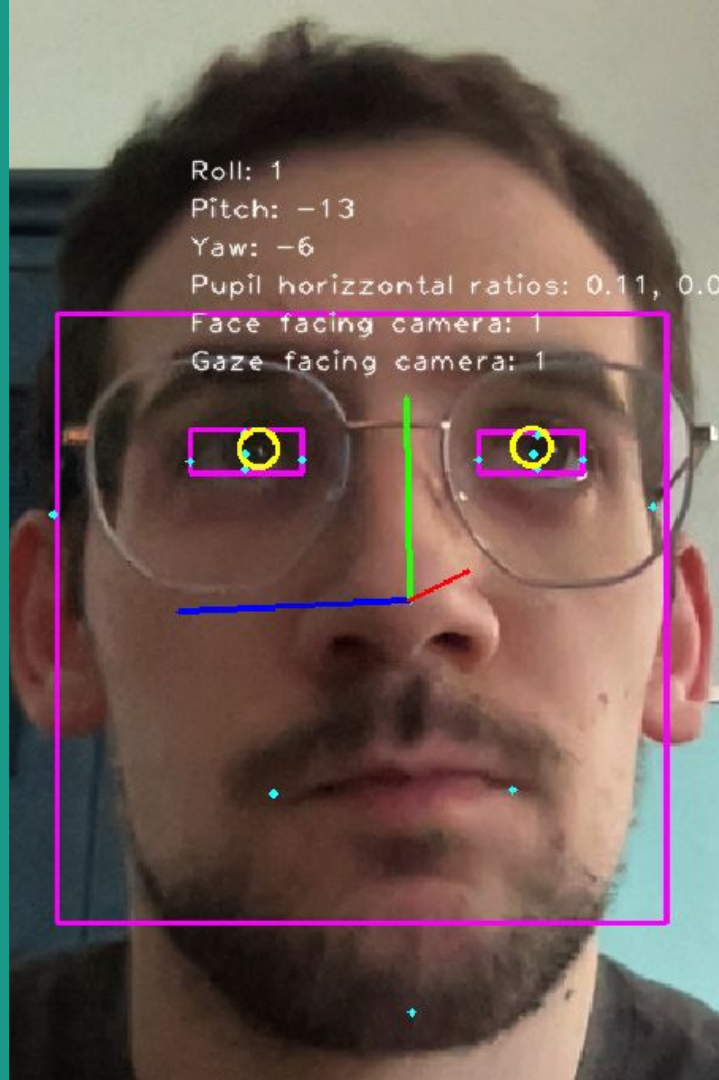

Gaze is Facing: False
Retrieval status: Detected and identified
Gaze is Facing: True
Retrieval status: Detected but not identified
Emotion Detected: surprise
FPS: 1.03
Gaze is Facing: True
IDENTIFICATION
BY RETRIEVAL

Pipeline


Gaze Analysis
0
Image acquisition and naive tracking
1
3
Identification by retrieval
2
Emotion Detection

01 Gaze analysis

- face and facial landmark detection
- pose estimation
- precize eye center localization



Face detection

Comparison between Viola-Jones Haar Cascade classifier and histogram of oriented gradients by Dalal and Triggs for the face detection.

Face detected	0	1	2	Time
Hog *dlib*	10932	24955	0	34.18 s
Haar Cascade *OpenCV*	15272	20608	7	38.38 s

Table 1. Comparison between two different classical face detection algorithms. They were tested on 35887 images containing a single image. They reach an accuracy of 0,70 and 0,57 respectively. Test made on a 2020 M1 MacBook Air.

Landmark detection

Dlib method that makes use of a cascade ensemble of regression trees

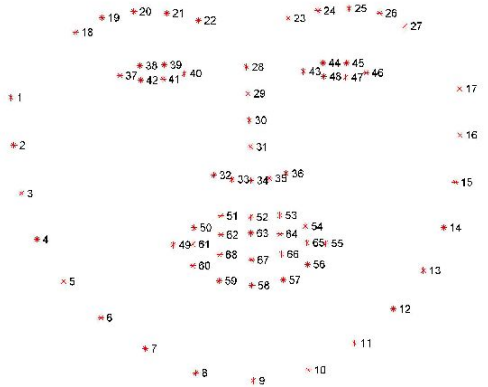


Figure 3. The 68 landmarks detected with *dlib*.

Pose estimation PnP problem

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

got using camera calibration, using the chessboard method

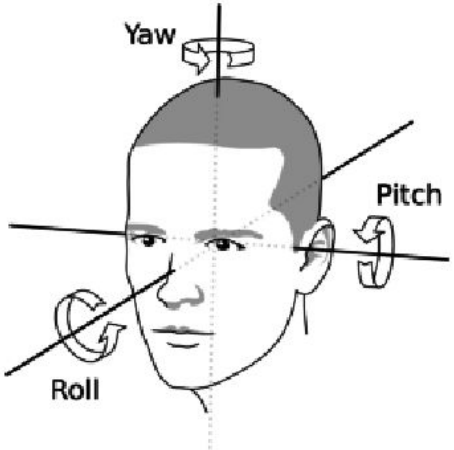
the unknowns to compute

given 3d model

$$\text{Pitch} = \text{atan2}(r_{32}, r_{33})$$

$$\text{Yaw} = \text{atan2}(-r_{31}, \sqrt{r_{32}^2 + r_{33}^2})$$

$$\text{Roll} = \text{atan2}(r_{21}, r_{11})$$



Pupil localization 1st method means of gradient

Using the the method proposed by Timm Barth, the point can be found by comparing the gradient vector g_i at position x_i with the displacement vector of a possible center d_i.

$$c^* = \arg\min_c \left\{ \frac{1}{N} \sum_{i=1}^{N} w_c (d_i^\top g_i)^2 \right\},$$

$$d_i = \frac{x_i - c}{||x_i - c||_2}, \forall i : ||g_i||_2 = 1$$

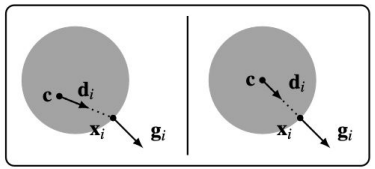
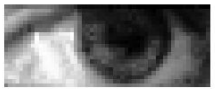


Figure 2: Artificial example with a dark circle on a light background, similar to the iris and the sclera. On the left the displacement vector $\mathbf{d}_i$ and the gradient vector $\mathbf{g}_i$ do not have the same orientation, whereas on the right both orientations are equal.

Pupil localization 2nd method filtering

Our approach is based on a series of transformation and filter on the eye image. The pupil is the “blob” with the biggest contour.



(a) Higher contrast and equalization.

(b) Gaussian blur and erosion.

(c) Adaptive thresholding.

Pupil localization testing

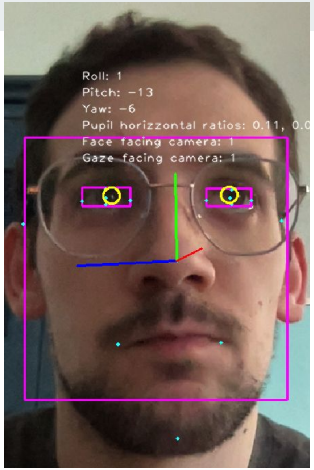
Computed with the Hausdorff distance which is the maximum all the distances between each point of a set and the closest point in the other set.

	Mean	Std. Dev.	Outliers	Avg. time
MoG	3.004	2.220	3	0.112
Filtering	3.198	1.156	3	0.028

Table 2. Test results of the pupil localization method. The time has been measured on a 2020 M1 MacBook Air.

Gaze estimation

The horizontal pupil ratio expresses how the pupil position within the eye, from -0.5 to 0.5, where 0.0 represent the position when the pupil is centered in the eye and 0.5 when the pupil is completely shifted towards the left corner.



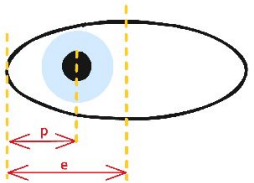


Figure 5. The horizontal pupil ratio hr can be found having the half length of the eye e and the distance of the pupil from the eye corner p. Then, $hr = (p/e) - 0.5$.

02 Identification by retrieval

- solution design
- prewhitening
- building phase
- inference phase
- testing phase

Solution design a very popular stack

MTCNN: Detection + Alignment using Cascaded specialized CNN

FaceNet: Based on InceptionResnet pre-trained on VGGFaces2/CASIA-WebFace for SoA face embeddings

Preprocessing: Prewhitening

Prewhitening

```
def prewhiten(x):
    mean = x.mean()
    std = x.std()
    std_adj = std.clamp(
        min=1.0/(float(x.numel()))**0.5))
    y = (x - mean) / std_adj
    return y
```

It subtracts the average and normalizes the range of the pixel values of input images. It makes training a lot easier.

Executed before every retrieval task

Building phase

Many transformation implemented [11]:

resize, *blur*, *motion blur*, *rotate*, *flip*, *brightness*, *contrast*, *saturation*, *zoom*, *tilt*, *translate*

Each ready to use and wrapped with safe boundaries but still with enough randomness

Fast flag that enables only suggested transformations.

Our TP enrolled faces dataset:

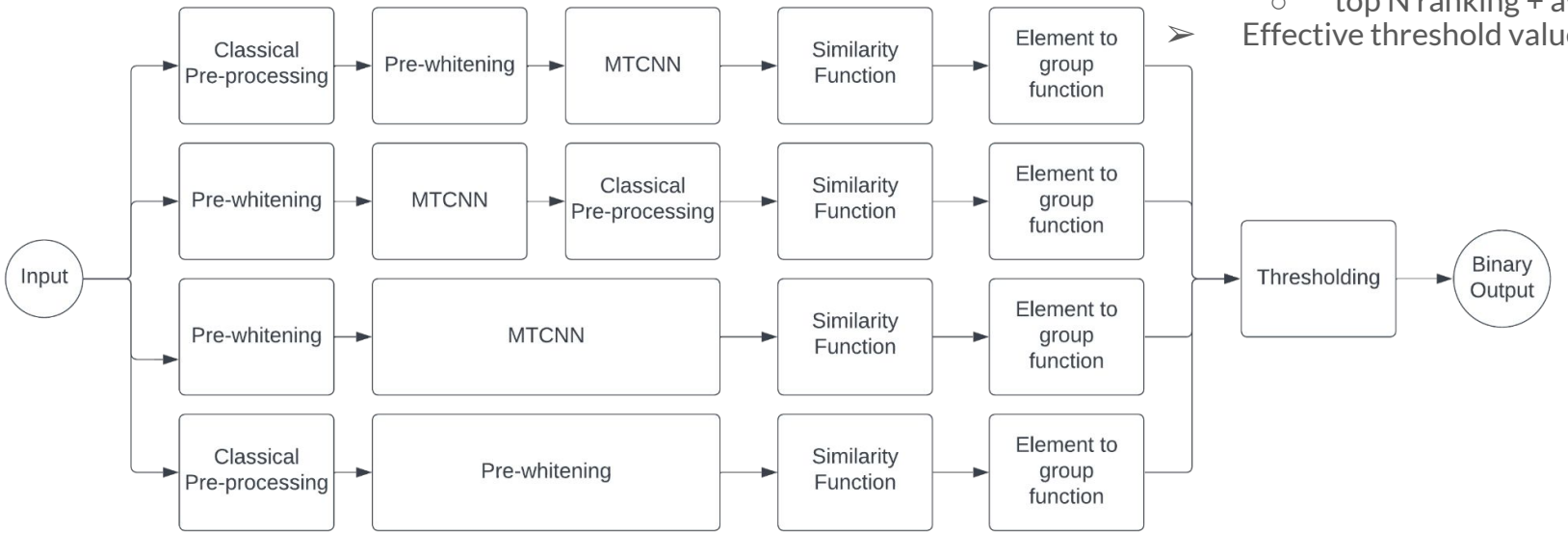
- 174 images
- no augmentation
- prewhitened
- aligned on the fly by MTCNN

Each image is augmented a configurable number of times.

Inference phase

A complex task:

- Prewhitening
- MTCNN or Classical face detection
- Real-time face embedding computation
- Effective similarity function
  - Cosine
  - L2 Euclidean
  - L1 Manhattan
- Effective element to group function
  - median
  - max
  - average
  - top N ranking + average (tentative)
- Effective threshold value



Test phase

We need a standard dataset as reference

=> Testing dataset: LFW (13’233) + **Unseen TP test set made by us** (391 no augm.) = 13’624 pictures

We need a standard output format

=> We must declare used metrics & configuration for each test in order to compare them

Why?

=> Many hyperparameters and design choices possible

Test phase a real output result example


```
"test_session_info": {
  "using_image_cap": false,
  "image_cap_value": 0,
  "threshold_used": 0.18,
  "distance_metric_used": "cosine",
  "pretrained_face_weights": "vggface2"
},  {
"metrics": {
  "precision": 100.0,
  "recall": 100.0,
  "f1_score": 100.0
}
```

```
"true_positives:": {
  "details": { ... },
  "outcome_summary": {
    "total_tp_dataset_size": 391,
    "detected_positives": 389,
    "false_negatives": 0,
    "errors": 2,
    "accuracy": 100.0
  }
```

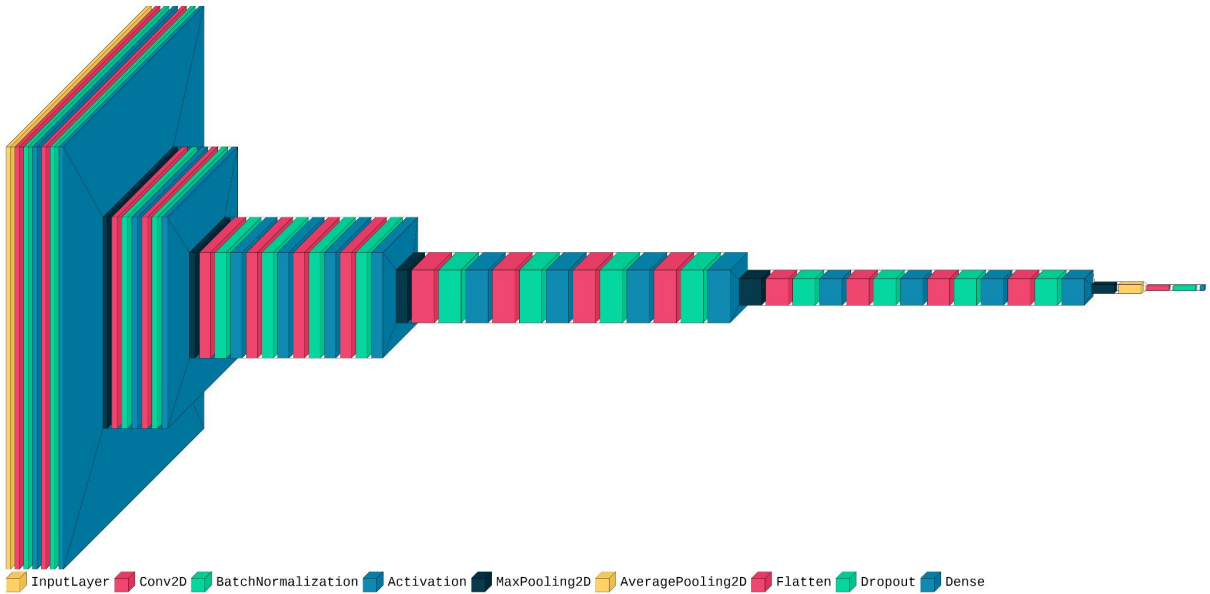
```
"true_negatives:": {
  "details": { ... },
  "outcome_summary": {
    "total_tn_dataset_size": 13233,
    "detected_negatives": 13232,
    "false_positives": 0,
    "errors": 1,
    "accuracy": 100.0
  }
```


* accuracy does not take into account errors

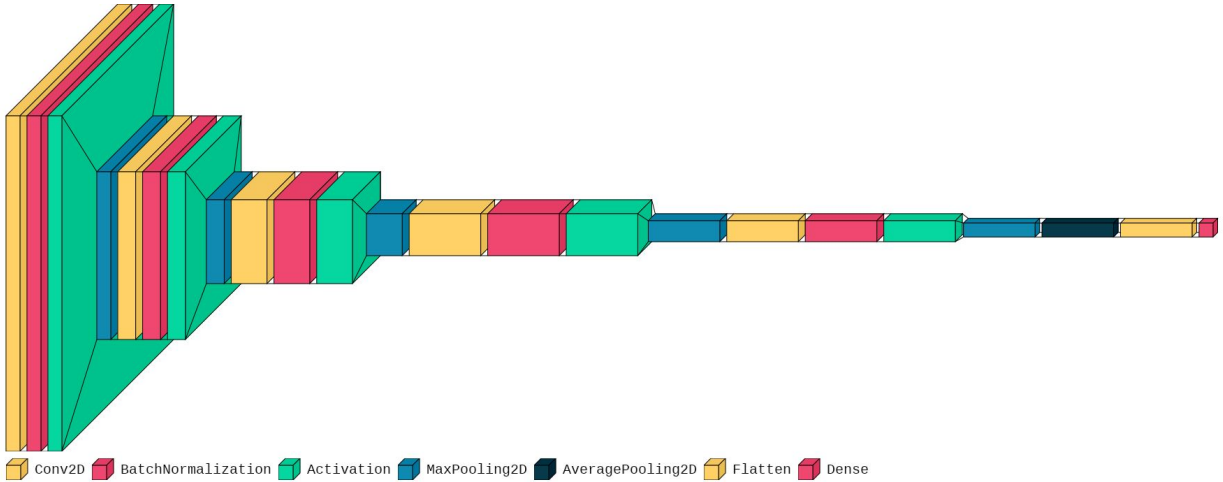
03 Emotion Detection

- network architectures (VGG19 and FERNet)
- data augmentation
- distillation
- loss function
- results

VGG19 and FERNet's architectures



- Activation function: ReLU
- Dropout set to 0.5
- Input Shape: 1x48x48
- Output shape: 1x7
- Total params: 20,037,831
- Trainable params: 20,037,831
- Non-trainable params: 0



- Activation Function: ReLU
- Input shape: 1x48x48
- Output shape: 1x7
- Total params: 3,916,167
- Trainable params: 3,916,167
- Non-trainable params: 0

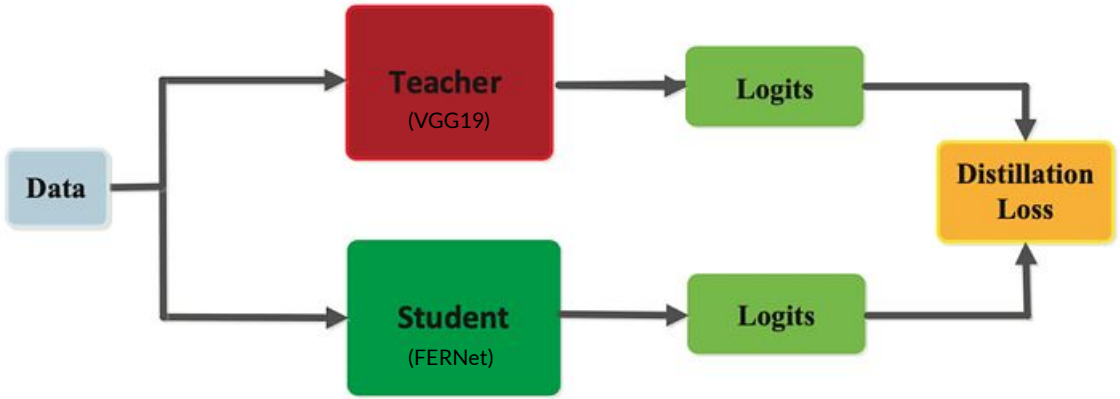
Data Augmentation & Dataset

Applied transformations:

- Grayscale
- RandomHorizontalFlip
- RandomAdjustSharpness

- 48x48 grayscale images
- CK+ (https://www.kaggle.com/datasets/shawon10/ckplus)
- BigFER (https://www.kaggle.com/datasets/jonathanoheix/face-expression-recognition-dataset)
- FER2013 (https://www.kaggle.com/datasets/msambare/fer2013)

Distillation



Loss Function

$$L = \alpha L_{KL} + (1 - \alpha) L_{CE}^{Student}$$

$$L_{KL} = L(y_{pred}, y_{true}) = y_{true} * log \frac{y_{true}}{y_{pred}}$$

Kullback-Leibler divergence Loss

$$L_{CE} = -\sum_{i=1}^{n} t_i \log p_i$$

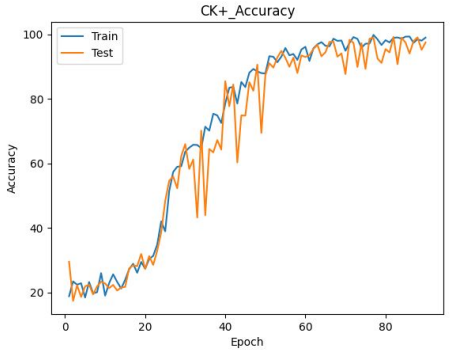
Cross-Entropy Loss

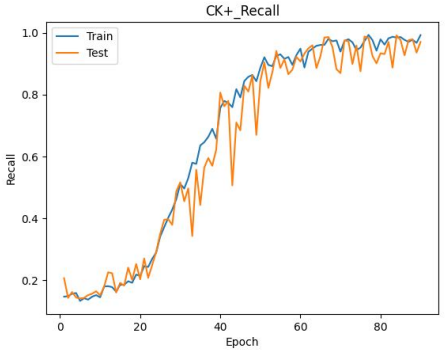
α : weighting factor
T: temperature parameter

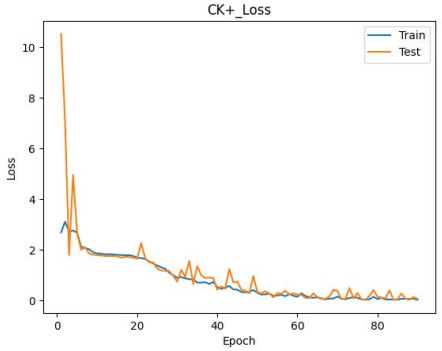
VGG19 Results

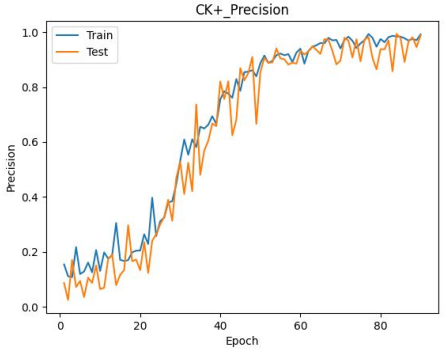
VGG19 trained on CK+

	CK+	BigFER	FER2013
Accuracy	98.47	66.40	65.71
Loss	0.0153	1.87	1.85





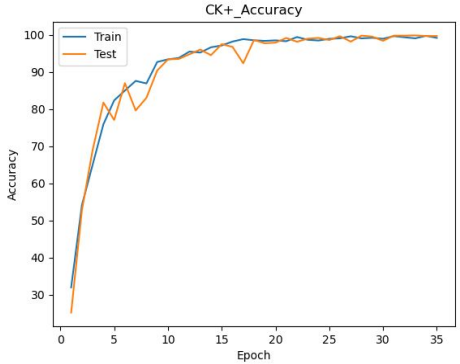


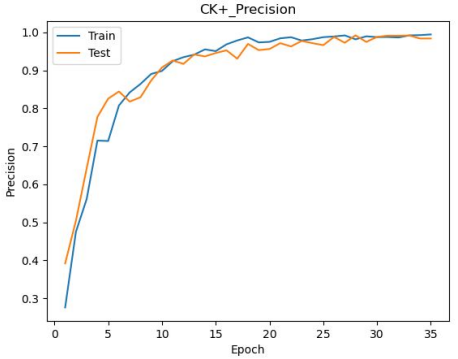


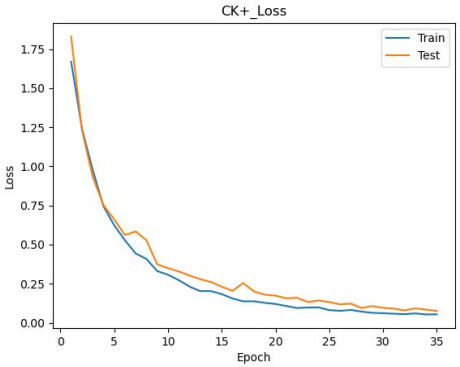
FERNet Results

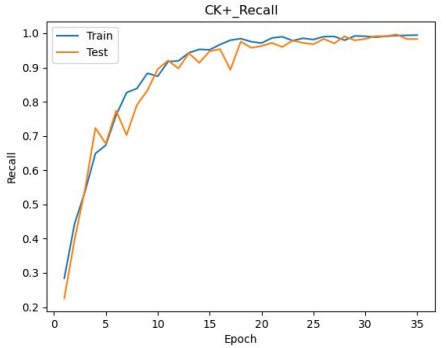
FERNet trained on CK+ using distillation with VGG19 as teacher

Accuracy	Loss	Precision	Recall	F1-Score
99.89	0.014	99.24	99.66	99.42

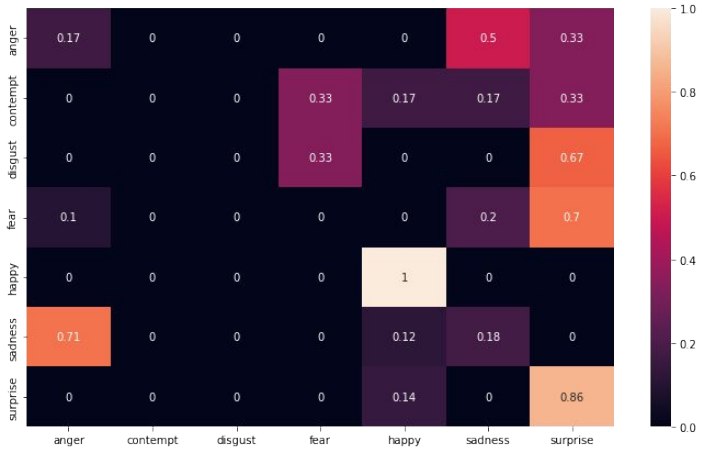








Confusion Matrix



FERNet confusion matrix

VGG19 confusion matrix

Thank you for your attention