

PROYECTO DEEP LEARNING (MODULO I y II)

**Titulo. Expresiones Faciales con
CNN+TransferLearning+MTCNN**

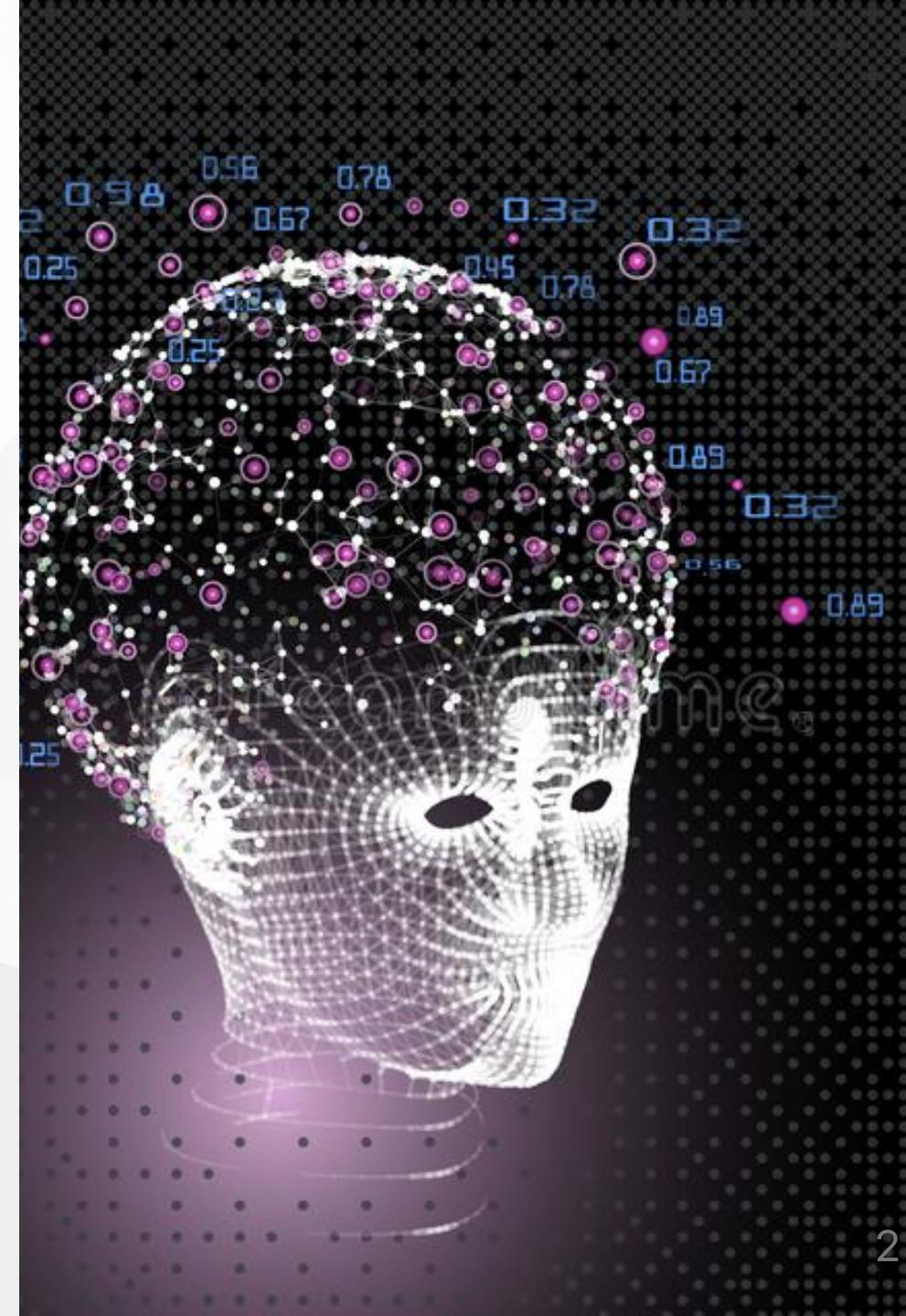
Elab. Mauricio Arancibia

m3IA



RESUMEN

1. Introducción
2. Objetivo del Trabajo
3. Dataset
4. Preparación de los Datos
5. FECNN
6. MTCNN
7. Modelo FECNN + MTCNN
8. Test's
9. Conclusiones y Recomendaciones



1.1 Introducción Expresiones Faciales (EF)

- Los seres humanos interactúan entre sí no solo a través de el habla, sino también a través de gestos corporales para enfatizar
- Son parte importante de la comunicación
- Transmiten señales no verbales
- Juegan un rol importante en las relaciones interpersonales



1.2 Introducción Expresiones Faciales (EF)

- Las EF pueden ser un componente importante de las interacciones hombre - máquina.
- Para estudiar ciencias del comportamiento y la práctica clínica.
- Neuromarketing



2. Objetivo del Trabajo

- Construir un modelo de Red Neuronal Convolucional (CNN) para reconocimiento de expresiones con Transfer Learning:
 - Enojo, felicidad, miedo, tristeza, asco, sorpresa y neutral
- Aplicar el modelo MTCNN para detección de rostros

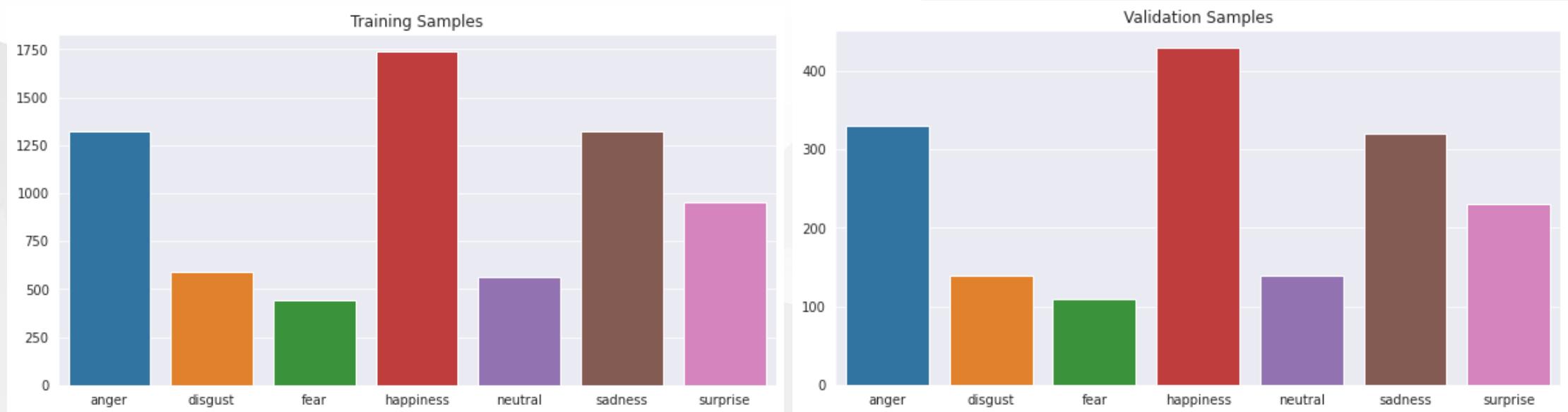


3. Dataset del trabajo

- Colección de 8638 imágenes RGB de 80x80 pixeles de caras de niños, adultos y mayores de edad obtenidas de Google Images.
 - 6938 imágenes para entrenamiento
 - 1700 imágenes para validación
- <https://www.kaggle.com/datasets/kmirfan/micro-expressions>

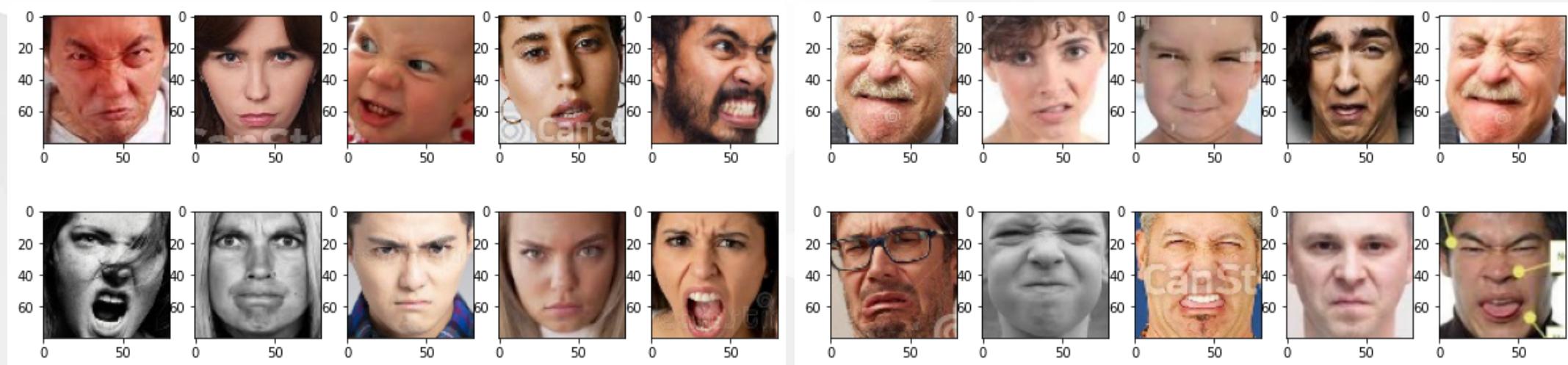
3.1 Dataset del trabajo

- Distribución de las muestras para entrenamiento y validación.



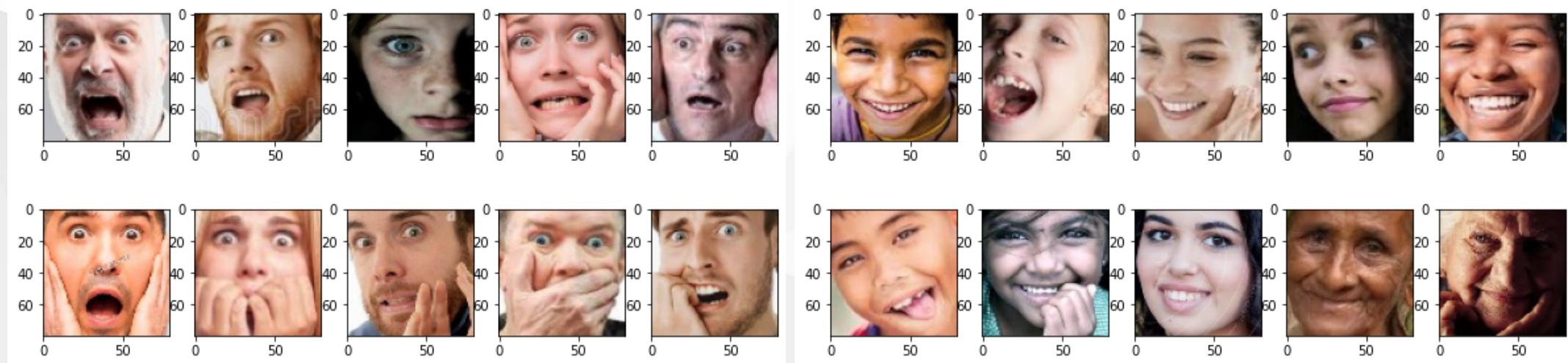
3.2 Muestras de imágenes

Enojo y Asco



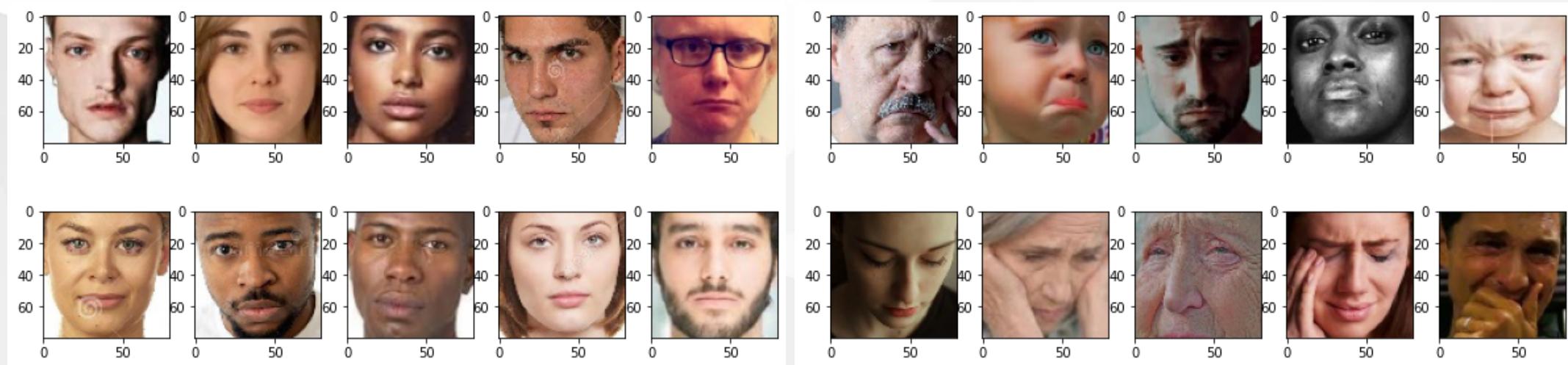
3.3 Muestras de imágenes

Miedo y Alegría



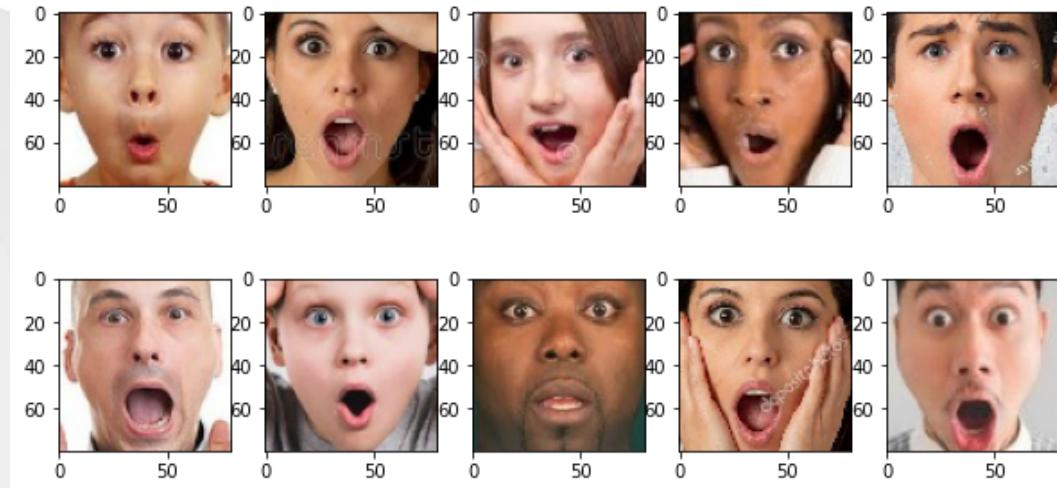
3.4 Muestras de imágenes

Neutral y Tristeza



3.5 Muestras de imágenes

Sorpresa



4.1 Preparación de los datos

- En esta etapa realizamos algunas transformaciones a los conjuntos de entrenamiento y validación
- Cargamos los datos con DataLoaders realizando algunas transformaciones y Data Augmentation como:
 - Rotacion aleatoria (random rotation)
 - Volcado horizontal aleatorio (random horizontal flip)
 - Redimensionamiento a imágenes 64x64 pixeles
 - CenterCrop
 - Normalización

4.2 Data Loaders

Trabajamos con un BATCH_SIZE de 32

```
train_data = datasets.ImageFolder(os.path.join(DIR_DATA, 'train'), transform=train_transforms)
valid_data = datasets.ImageFolder(os.path.join(DIR_DATA, 'test'), transform=valid_transforms)

#torch.manual_seed(42)
train_loader = DataLoader(train_data, batch_size=BATCH_SIZE, shuffle=True)
valid_loader = DataLoader(valid_data, batch_size=BATCH_SIZE, shuffle=False)

class_names = train_data.classes
class_names

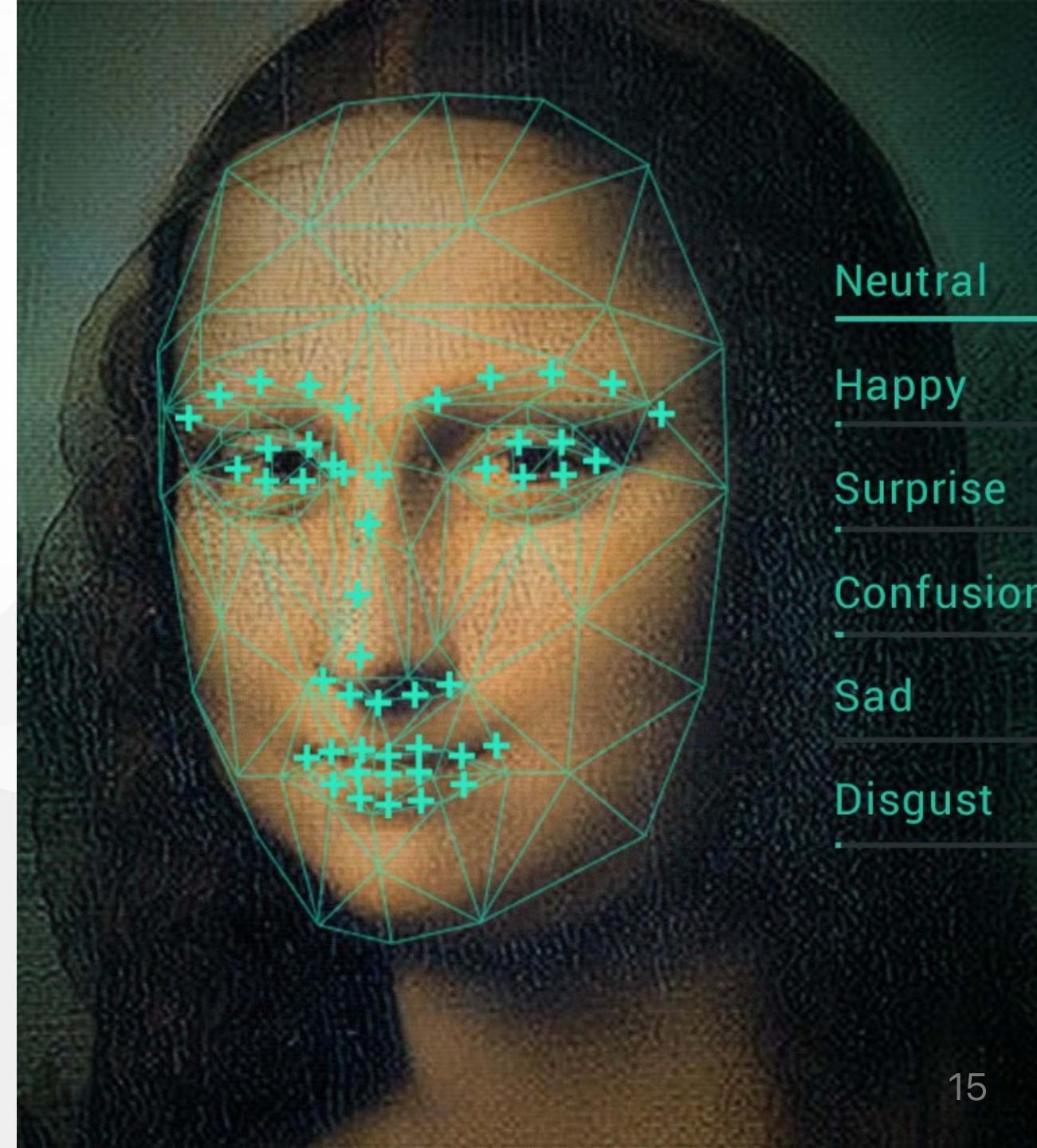
['anger', 'disgust', 'fear', 'happiness', 'neutral', 'sadness', 'surprise']
```

4.3 Data Loader samples



5. CNN - Preparación del Modelo (Facial Expressions)

- Custom Model
- Transfer Learning

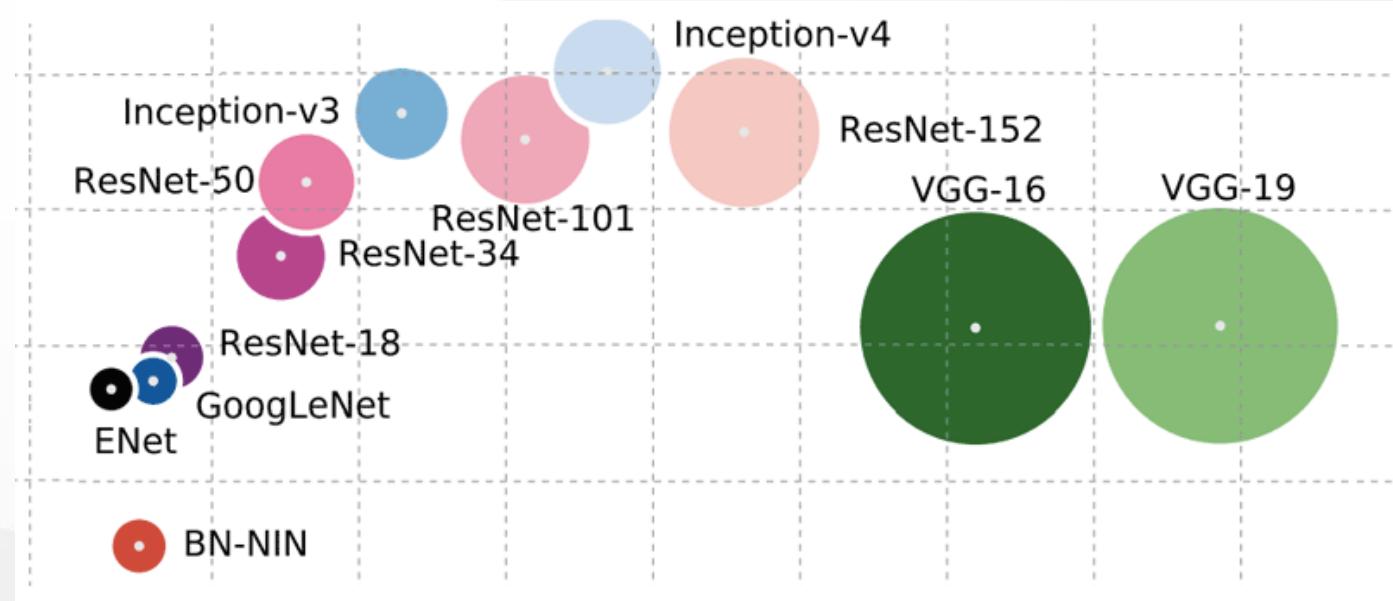


5.1 Custom Model

```
CustomModel(  
    (conv1): Conv2d(3, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))  
    (conv2): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))  
    (conv3): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))  
    (fc1): Linear(in_features=65536, out_features=1024, bias=True)  
    (fc2): Linear(in_features=1024, out_features=7, bias=True)  
    (pool): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)  
    (dropout): Dropout(p=0.2, inplace=False)  
)
```

5.2 Transfer Learning

- Resnet18
- VGG19
- Resnet50



5.3 Resnet18, VGG19 y Resnet50

- Resnet18 (Trainable = True)

```
model_resnet18.fc = nn.Sequential(nn.Linear(num_ftrs, 256),  
                                  nn.ReLU(inplace=True),  
                                  nn.Dropout(0.2),  
                                  nn.Linear(256, 7))
```

- VGG19 (Trainable = False)

```
vgg_based.classifier = nn.Sequential(nn.Linear(25088, 256),  
                                      nn.ReLU(inplace=True),  
                                      nn.Dropout(0.2),  
                                      nn.Linear(256, 7))
```

5.4 Resnet18, VGG19 y Resnet50

- Resnet50 (Trainable = False)

```
rn50_based.fc = nn.Sequential(  
    nn.Linear(2048, 256),  
    nn.ReLU(inplace=True),  
    nn.Dropout(0.2),  
    nn.Linear(256, 7))
```

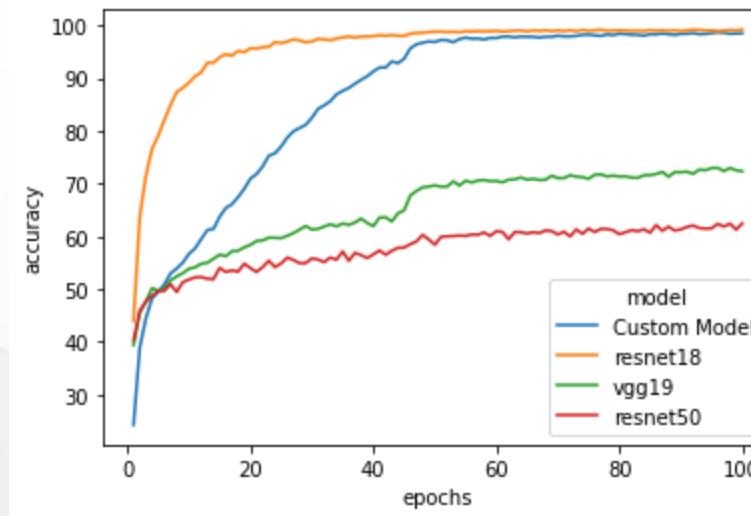
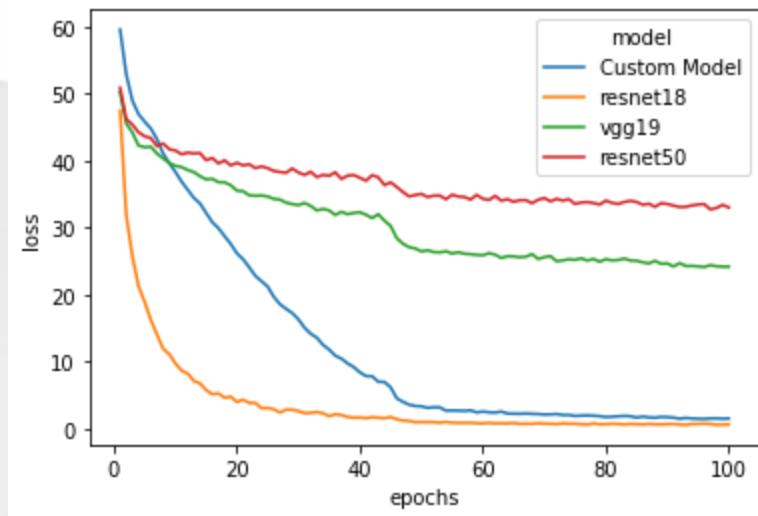
5.5 Training

- Learning Decay = 0.1
- Epochs = 100
- Optimizador = SGD
- Loss = CrossEntropyLoss



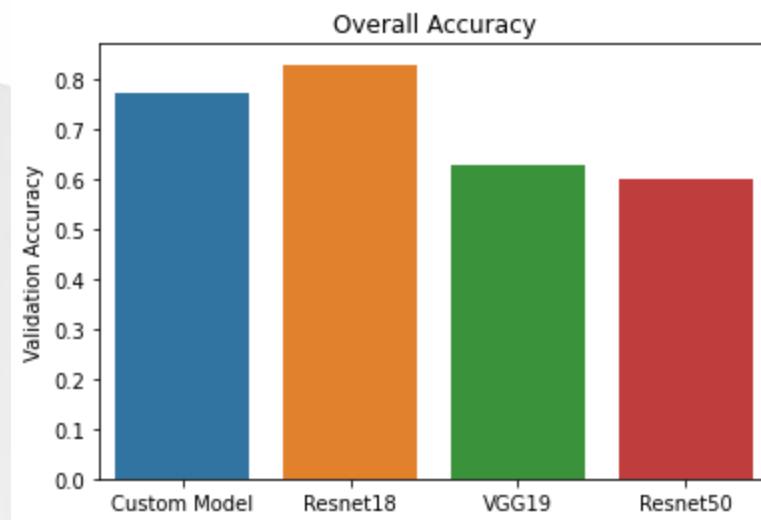
5.6 Métricas del modelo CNN y desempeño

- Loss y Accuracy (training)



5.7 Métricas del modelo CNN y desempeño

- Overall Accuracy para conjunto Validación



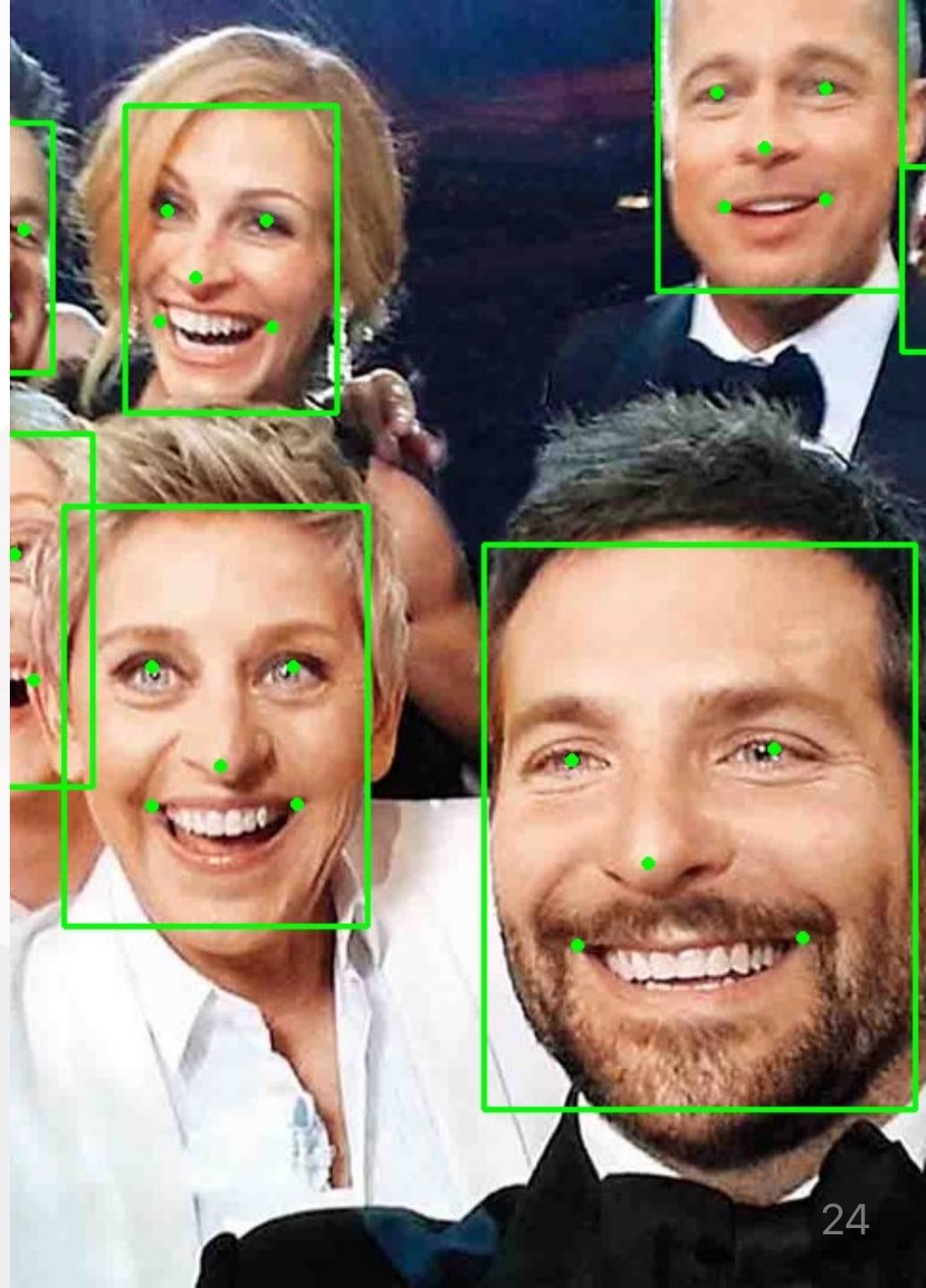
5.5 Selección del modelo

- Training Accuracy: 99.251%
- Training loss: 0.657
- Validation Accuracy = 82%
- Por clase:

Accuracy of anger: 82 %
Accuracy of disgust: 73 %
Accuracy of fear: 64 %
Accuracy of happiness: 94 %
Accuracy of neutral: 74 %
Accuracy of sadness: 81 %
Accuracy of surprise: 83 %

6. Modelo MTCNN para detección de rostros

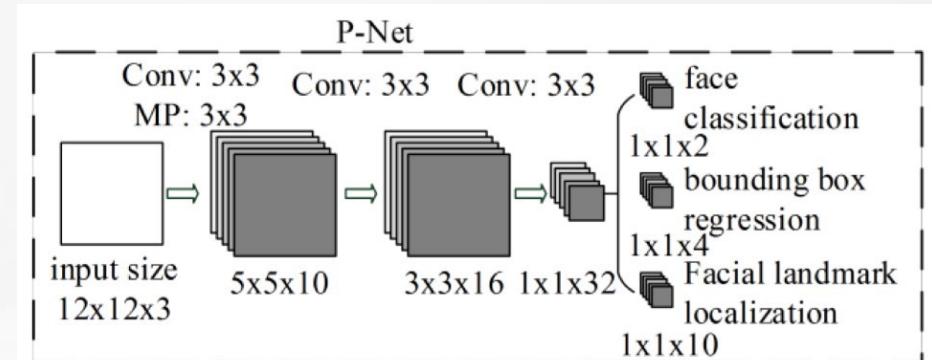
- Multi-Task Cascaded Convolutional Neural Networks (MTCNN), es una rede neuronal que detecta rostros humanos en imágenes.
- Es una de las herramientas más populares y más precisas para detección de rostros
- Consiste de tres NN conectadas en cascada (P-Net, R-Net, O-net)



6.1 MTCNN - Primera Etapa

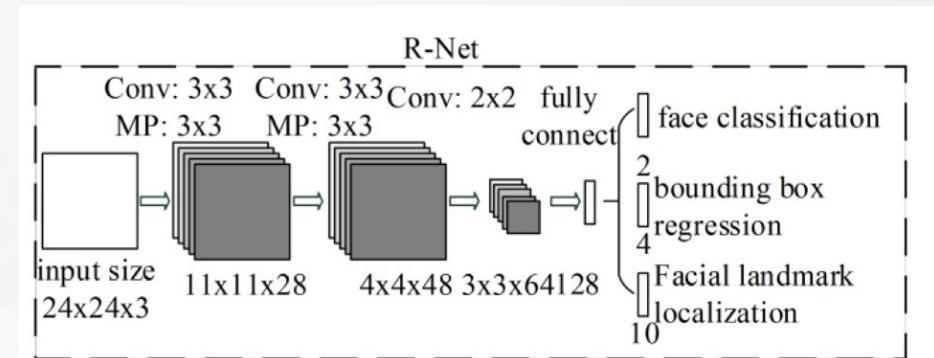
P-Net

- En esta etapa se tiene una Red de Propuestas Fully Connected que se utiliza para obtener ventanas candidatas y reducir la superposición y el número de cajas.
- Toma como entrada una imagen piramidal formada por copias a diferente escala de la imagen de entrada. Esto proporciona al modelo una amplia gama de tamaños de ventana para elegir y ayuda a que el modelo sea invariable a escala.



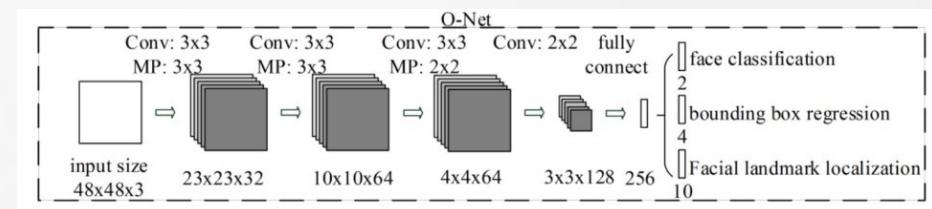
6.2 MTCNN - Segunda Etapa R-Net

- La segunda etapa es una CNN Refine Network (R-Net). Reduce aún más el número de casillas y fusiona candidatos superpuestos utilizando supresión no máxima (NMS).

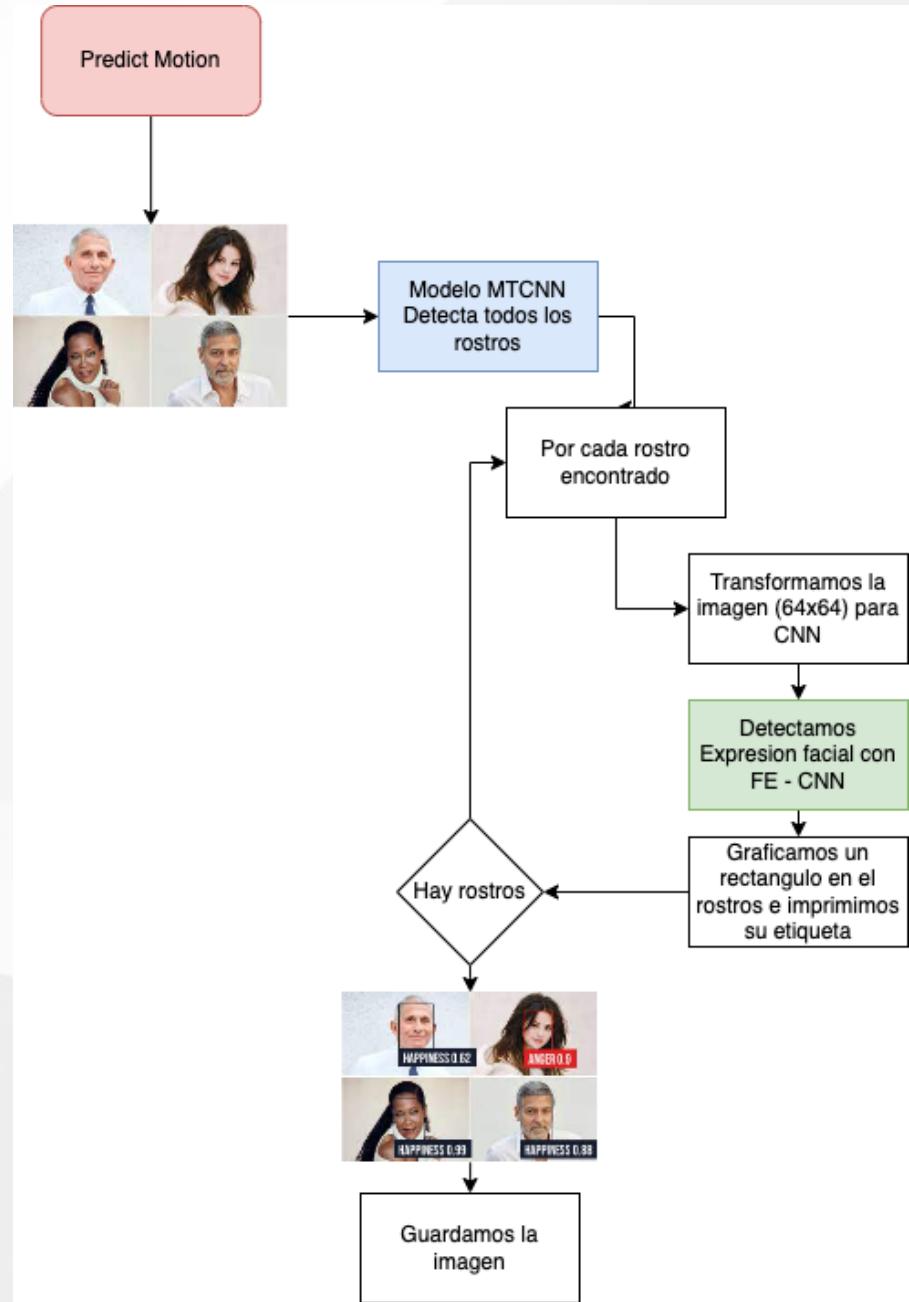


6.3 MTCNN - Tercera Etapa O-Net

- La red de salida en la tercera etapa hace más de lo mismo que hace R-Net, y agrega el punto de referencia de 5 puntos de ojos, nariz y boca en el cuadro delimitador final que contiene la cara detectada.



7. Algoritmo de Detección de Rostros (MTCNN) y Expresiones Faciales (CNN)



8.1 Test 1



8.2 Test 2



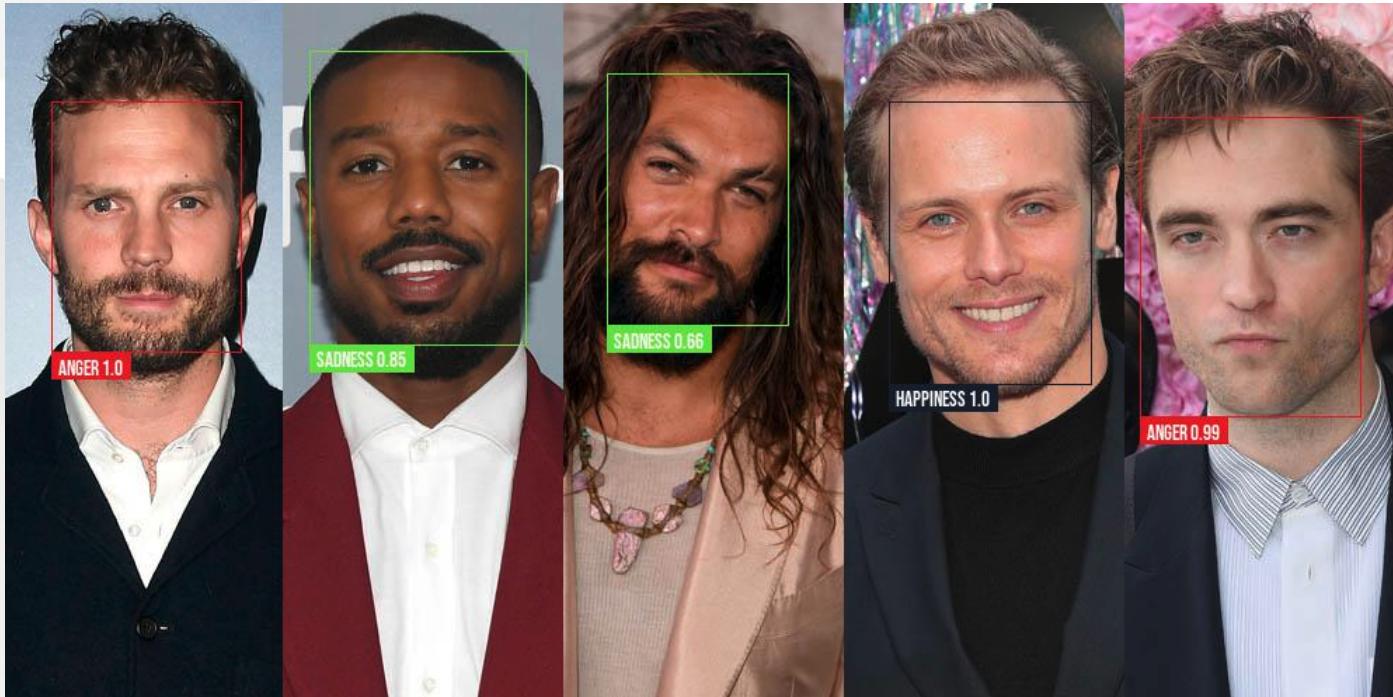
8.3 Test 3



8.4 Test 4



8.5 Test 5



9. Conclusiones y Recomendaciones



Referencias

- <https://analyticsindiamag.com/top-8-datasets-available-for-emotion-detection/>
- http://cs231n.stanford.edu/reports/2016/pdfs/005_Report.pdf
- <https://arsfutura.com/magazine/face-recognition-with-facenet-and-mtcnn/>
- <https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/iet-ipr.2019.0141#:~:text=Multi-task cascaded convolutional neural,whole process of the MTCNN.>
- <https://medium.com/swlh/identifying-faces-with-mtcnn-and-vggface-9d0d4927cccf>

Recursos

<https://github.com/urvog/FECNN-MTCNN>

Gracias!

