REPORT

Submitted by,

NIDIN V NANDAN

MSc Computer Science With Data Analytics
Reg No: 223039

TASK: CELEBRITY IMAGE CLASSIFICATION

Dataset:

Different folders named on celebrity name containing their photos

CHOSEN MODEL- Convolutional Neural Network(CNN)

"

```
model = tf.keras.models.Sequential([
        tf.keras.layers.Conv2D(32, (3, 3), activation='relu', input_shape=(128, 128, 3)),
        tf.keras.layers.MaxPooling2D((2, 2)),
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(256, activation='relu'),
        tf.keras.layers.Dropout(0.5),
        tf.keras.layers.Dense(512, activation='relu'),
        tf.keras.layers.Dense(5, activation='softmax')

])

model.compile(optimizer='adam',loss='sparse_categorical_crossentropy', metrics=['accuracy'])"
```

Here

Input Layer: Accepts images of size 128x128 pixels with three color channels (RGB).

Convolutional Layers:

32 filters of size 3x3, using ReLU activation function.

Followed by max-pooling with a 2x2 window to reduce spatial dimensions.

Flattening Layer:

Flattens the output from the convolutional layers into a 1D array to feed into the densely connected layers.

Densely Connected Layers:

First dense layer with 256 neurons and ReLU activation.

Dropout layer with a rate of 0.5 to reduce overfitting.

Second dense layer with 512 neurons and ReLU activation.

Final dense layer with 5 neurons, using the softmax activation function for multi-class classification (outputting probabilities for 5 classes).

Optimizer: Adam optimizer is used

Loss function: Sparse categorical cross-entropy, which is suitable for multi-class classification.

Metrics: Accuracy, to evaluate the model's performance during training.

Model summary:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 126, 126, 32)      896

 max_pooling2d (MaxPooling2  (None, 63, 63, 32)        0
 D)

 flatten (Flatten)           (None, 127008)            0

 dense (Dense)               (None, 256)               32514304

 dropout (Dropout)           (None, 256)               0

 dense_1 (Dense)             (None, 512)               131584

 dense_2 (Dense)             (None, 5)                 2565

=================================================================
Total params: 32649349 (124.55 MB)
Trainable params: 32649349 (124.55 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Training:

history = model.fit(x_train, y_train, epochs=40, batch_size=128, validation_split=0.1)

Here the model is trained for 40 epochs with batch size of 128

Evaluation:

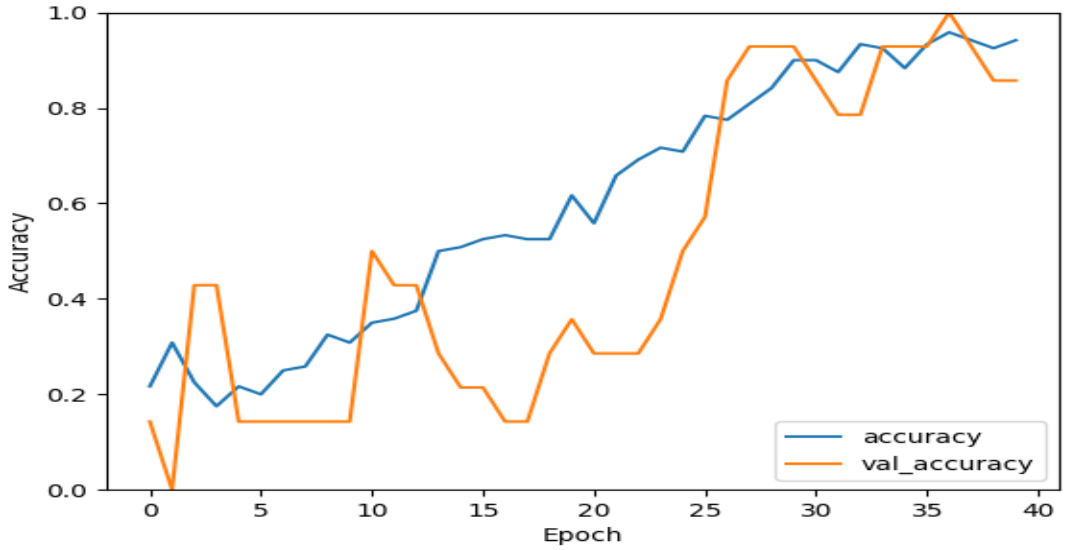
Here we evaluated the model on the test data and a classification report was generated

The accuracy was recorded as 76%

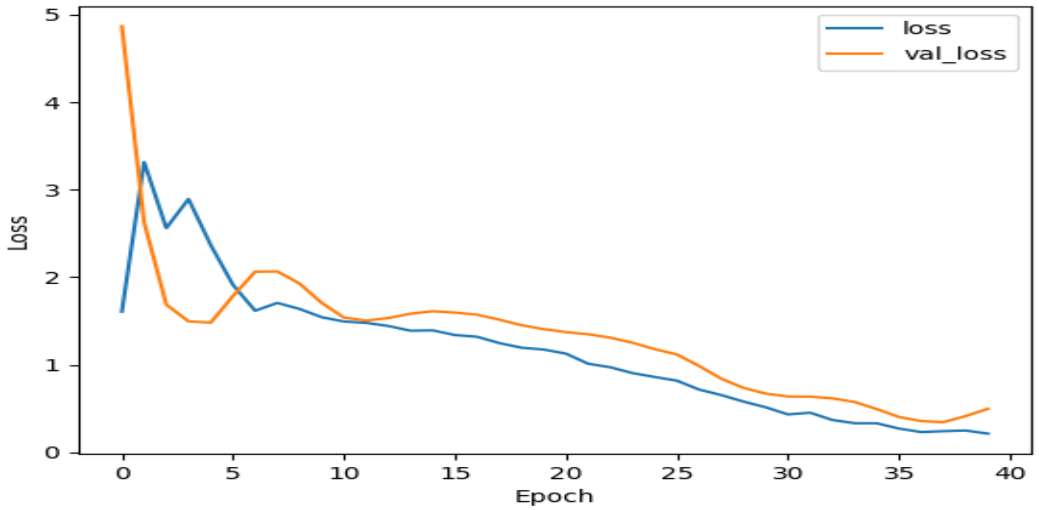
Classification Report:

```
Classification Report
              precision    recall  f1-score   support

           0       0.88      0.70      0.78        10
           1       0.75      0.86      0.80         7
           2       0.75      0.75      0.75         4
           3       0.75      0.50      0.60         6
           4       0.70      1.00      0.82         7

    accuracy                           0.76        34
   macro avg       0.77      0.76      0.75        34
weighted avg       0.78      0.76      0.76        34
```

Accuracy plot:



Loss plot:



Prediction:

A function "predict_celebrity" was created in order to take the user input image a preprocessed for predicting the celebrity

And we can observe that the model is predicting correctly the celebrities among the inputed image