CSY3025 Artificial Intelligence Techniques

Scan your face to register your attendance

Face Recongnition Based Attendence System

In [19]:

Out[19]:

1.Introduction

Face recognition, a prominent application of deep learning, is a fascinating field that has garnered significant attention in recent years. The ability to accurately identify and verify individuals based on their facial features has numerous practical applications across various domains, including security systems, surveillance, access control, and personalized user experiences. This technology has the potential to revolutionize the way we interact with machines and each other.

Deep learning models have proven to be highly effective in face recognition tasks due to their ability to learn intricate patterns and extract meaningful representations from complex visual data. Unlike traditional methods that rely on handcrafted features, deep learning models can automatically learn discriminative features directly from raw images. This capability allows for more robust and accurate face recognition, even in the presence of variations in pose, lighting conditions, facial expressions, and occlusionsOne of the key advantages of deep learning-based face recognition is its ability to capture and utilize high-dimensional facial representations. By learning from large-scale datasets, deep neural networks can encode facial features into compact and semantically meaningful representations, often referred to as face embeddings. These embeddings possess the desirable property of being highly discriminative, allowing for efficient and accurate face matching and identification.

Furthermore, deep learning models can adapt and generalize well to diverse face datasets, accommodating a wide range of facial appearances, demographics, and ethnicities. This capability is crucial for ensuring fairness and inclusivity in face recognition systems, as they need to perform equally well across different populations.However, despite the remarkable progress in deep learning-based face recognition, challenges still exist. Variations in pose, illumination, resolution, and occlusions can affect the performance of recognition systems. Moreover, issues related to privacy, security, and ethical considerations surrounding the use of face recognition technology continue to be subjects of debate and regulation.

In this report, we delve into the exciting world of face recognition using deep learning, exploring the underlying principles, methodologies, and advancements in this field. We will discuss the advantages and limitations of deep learning models in face recognition tasks and examine real-world applications where this technology has made significant contributions.

2.Problem analysis and backgroundresearch

Face recognition is a challenging task that involves accurately identifying and verifying individuals based on their facial features. While humans can consider multiple factors, such as visual and verbal cues, to recognize faces, deep learning models, particularly in the context of Convolutional Neural Networks (CNNs), face limitations. Deep learning models struggle to capture the entirety of a person's presence and accurately predict their identity. Factors like variations in pose, lighting conditions, facial expressions, and occlusions pose challenges for achieving high accuracy. Additionally, each individual is unique, expressing their facial features and characteristics differently, making it challenging to generalize across diverse populations.

Additionally, face recognition systems encounter hurdles when faced with diverse populations. People possess unique facial characteristics, and models must account for variations in age, ethnicity, and gender to achieve accurate and unbiased recognition. Overcoming these challenges requires comprehensive and diverse training datasets that encompass a wide range of facial attributes and demographics. A further obstacle arises from practical considerations, such as the usage of face coverings. Masks and other facial obstructions impede the visibility of essential facial features, making accurate identification more difficult. The development of techniques that can adapt to partial face visibility and handle occlusions is crucial to maintaining the reliability and effectiveness of face recognition systems in real-world scenarios. Addressing these challenges is essential to unlock the full potential of face recognition technology.

Continued research and innovation are needed to refine algorithms, improve model training, and account for the diversity and uniqueness of human faces. By pushing the boundaries of what is possible, we can create face recognition systems that are accurate, fair, and adaptable, fostering trust and enabling their successful integration into various domains, including security, access control, and personalized experiences.

3. Building deep learning network

Developing a facial recognition deep learning model entails various techniques, data acquisition, and preparations. To create a solution capable of identifying and classifying facial characteristics, particularly for individuals wearing face coverings, several critical steps must be taken. These include sourcing a reliable dataset specifically suited for this context, designing the model architecture, and training it using the gathered data. Achieving a high level of accuracy in predicting emotions from portrait images becomes a formidable challenge, necessitating thorough investigation and analysis of all the essential components required for the development of an effective model

3.1.Dataset

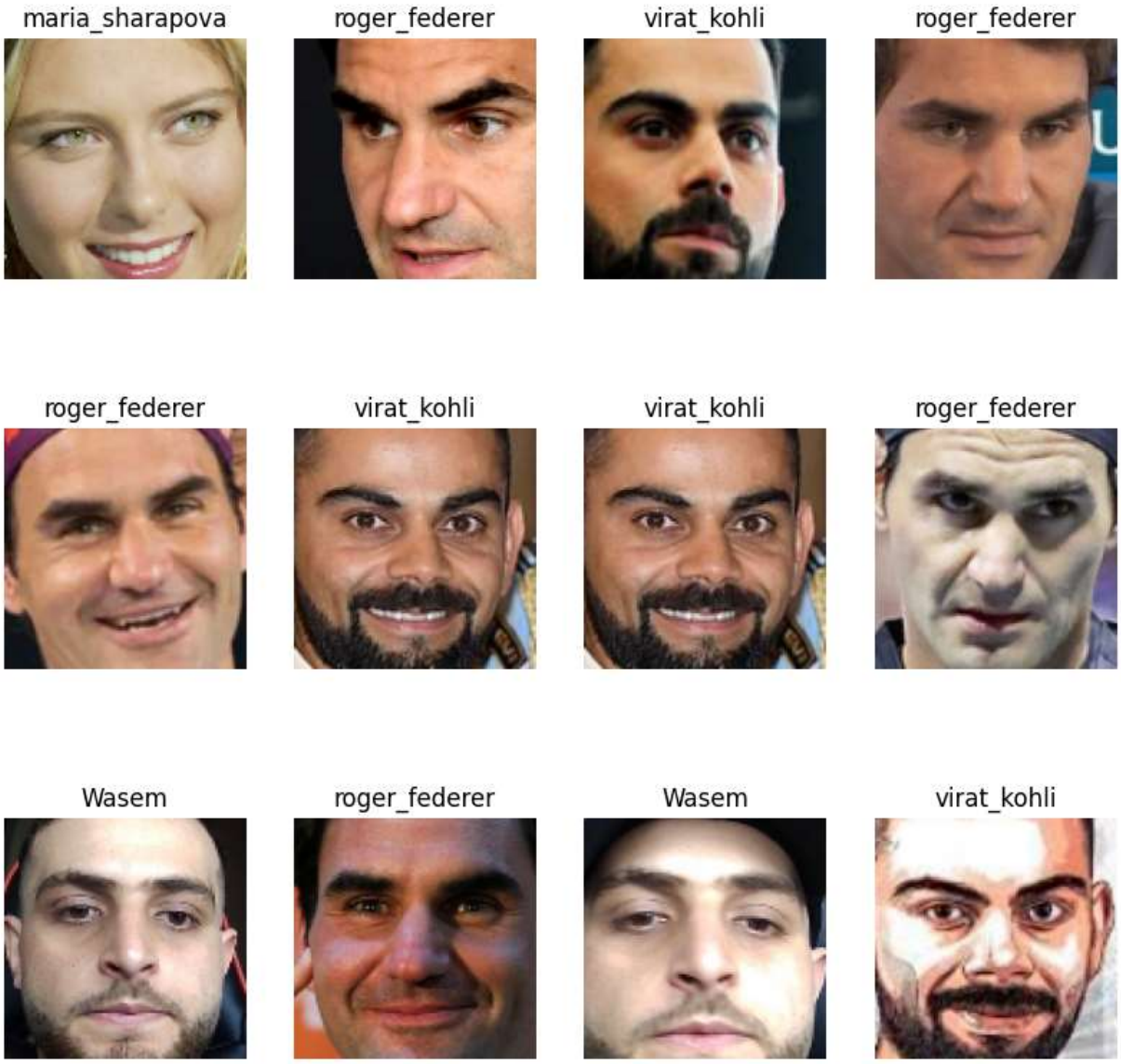
Dataset Link

https://drive.google.com/drive/folders/1t6n99FPXqr1_6DuN1rWTXNQABES2Msc6?usp=share_link

Dataset sample

In [20]:
```
from IPython.display import Image
Image(filename='datasample.png')
```

Out[20]:



The foundation of this endeavor lies in acquiring a dataset that accurately represents individuals wearing face coverings, providing diverse examples of facial expressions. This dataset becomes the basis for training the deep learning model, enabling it to recognize and classify emotions based on limited facial cues. The model's architecture must be carefully designed, leveraging techniques such as Convolutional Neural Networks (CNNs) to extract features and capture the nuances of facial expressions. Adequate training is crucial, as it allows the model to learn the complex patterns and correlations between facial characteristics and emotions, enhancing its ability to make accurate predictions. Developing an efficient and reliable facial recognition model for individuals wearing face coverings poses a unique set of challenges. The model needs to navigate the limitations imposed by obscured facial features and the reduced visual information available. Extensive research and analysis are required to identify and incorporate the most suitable techniques and strategies to address these challenges effectively. By conducting a comprehensive investigation and leveraging the power of deep learning, we can strive towards developing an accurate and robust face recognition solution that can analyze facial characteristics even when individuals are wearing face coverings.

3.2 DL Network (structure, loss function, optimiser, etc.)

Purpose and Structure:

CNNs are deep learning models specifically designed for analyzing visual data, such as images. They are composed of multiple layers, including convolutional layers, pooling layers, and fully connected layers. The architecture of CNNs is inspired by the organization of the visual cortex in the human brain.

Structure of CNNs:

CNNs consist of multiple layers that work together to process and analyze images. Here's a breakdown of the key components:

1. ### Convolutional Layers:

   Convolutional layers are the building blocks of CNNs. Each convolutional layer consists of learnable filters, also known as kernels or feature maps. These filters convolve across the input image, performing element-wise multiplication and summing up the results. This process extracts important local features, such as edges, corners, and textures, from the input image.

2. ### Pooling Layers:

   After each set of convolutional layers, pooling layers are commonly used to downsample the feature maps. Pooling reduces the spatial dimensions of the feature maps while retaining the most important information. The most common pooling operation is max pooling, which selects the maximum value within each pooling window. This downsampling process helps to make the network translation-invariant and reduces the computational burden.

3. ### Fully Connected Layers:

   The fully connected layers come after the convolutional and pooling layers. They take the high-level representations learned by the previous layers and map them to specific classes or categories. In these layers, every neuron is connected to every neuron in the previous layer, similar to traditional neural networks. The fully connected layers enable the network to make predictions based on the extracted features.

4. ### Activation Functions:

   Activation functions introduce non-linearity to the network, allowing it to learn complex relationships between features. The most commonly used activation function in CNNs is the

Rectified Linear Unit (ReLU). ReLU sets negative values to zero and keeps positive values unchanged. This non-linear activation helps the network learn more expressive representations.

5. # Parameter Sharing:

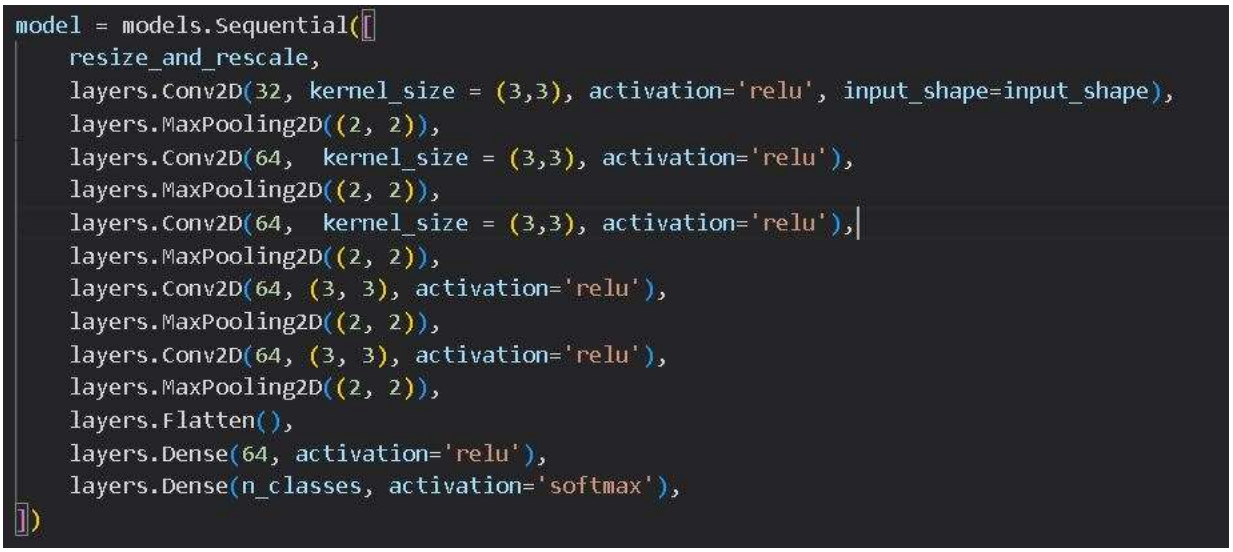
CNNs utilize parameter sharing to reduce the number of learnable parameters in the network. The idea is to use the same set of weights (parameters) for different regions of the input. This sharing of weights enables the network to detect the same features regardless of their location in the image. By sharing parameters, CNNs become more efficient, require less memory, and can better generalize to new data.

The inspiration for CNN architecture comes from the organization of the visual cortex in the human brain. The visual cortex contains layers of cells that progressively process visual information, with each layer analyzing features at different levels of abstraction. CNNs mimic this hierarchical structure by using layers to capture increasingly complex visual representations. This design allows CNNs to learn and extract meaningful features from images, making them highly effective for visual data analysis tasks.

Structure used In the code

In [21]:
```
from IPython.display import Image
Image(filename='network.JPG')
```

Out[21]:



```
model = models.Sequential([
    resize_and_rescale,
    layers.Conv2D(32, kernel_size = (3,3), activation='relu', input_shape=input_shape),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64,  kernel_size = (3,3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64,  kernel_size = (3,3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dense(n_classes, activation='softmax'),
])
```

Compiling the Model

We use `adam` **Optimizer**, `SparseCategoricalCrossentropy` **LOSS**

SparseCategoricalCrossentropy

The SparseCategoricalCrossentropy loss function is a commonly used loss function in deep learning models, particularly in tasks involving multi-class classification. It is

designed to handle scenarios where the target variable is represented as integers (class labels) rather than one-hot encoded vectors. The function compares the predicted probability distribution with the true class labels and calculates the cross-entropy loss, encouraging the model to minimize the difference between predicted and true labels. It is a suitable choice when dealing with large number of classes, as it eliminates the need for one-hot encoding, saving computational resources and memory.

Adam

The Adam optimizer is an adaptive optimization algorithm commonly used in deep learning. It combines the concepts of adaptive learning rates and momentum to efficiently update the model parameters during training. Adam adapts the learning rate for each parameter based on estimates of both the first-order (gradient) and second-order (momentum) moments. This adaptive nature allows Adam to automatically adjust the learning rate for each parameter, leading to faster convergence and better performance compared to traditional optimization methods. It is widely used due to its effectiveness, robustness, and ease of implementation in various deep learning tasks.

In [22]:
```
from IPython.display import Image
Image(filename='loss.JPG')
```

Out[22]:

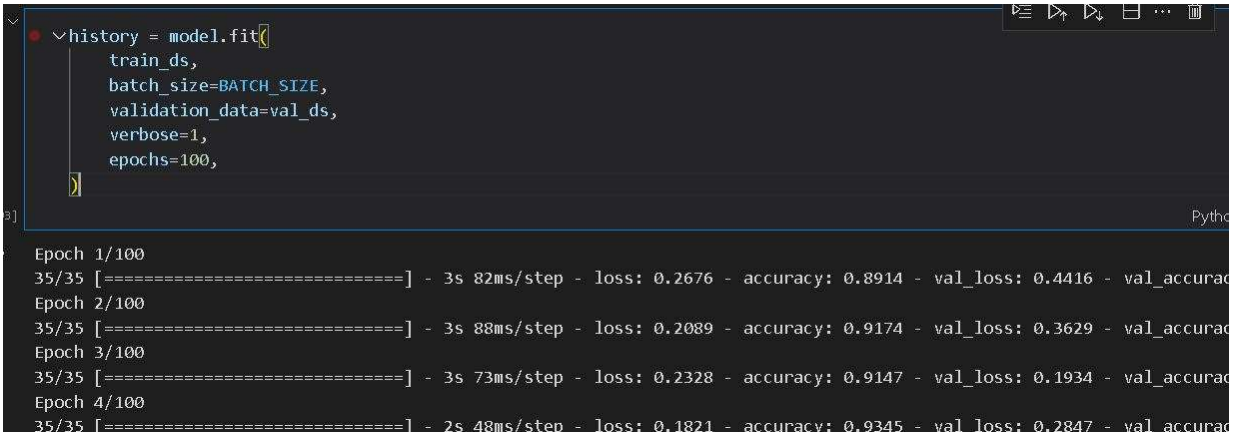
Training

Training in the context of a Convolutional Neural Network (CNN) involves the process of optimizing the model's parameters by iteratively feeding it with labeled data, calculating the loss or error between the predicted and true labels, and using an optimization algorithm to update the model's weights. The goal of training is to enable the CNN model to learn patterns and features from the input data, improving its ability to make accurate predictions over time.

Current Model is trained on 100 epochs

In [23]:
```
from IPython.display import Image
Image(filename='train.JPG')
```

Out[23]:



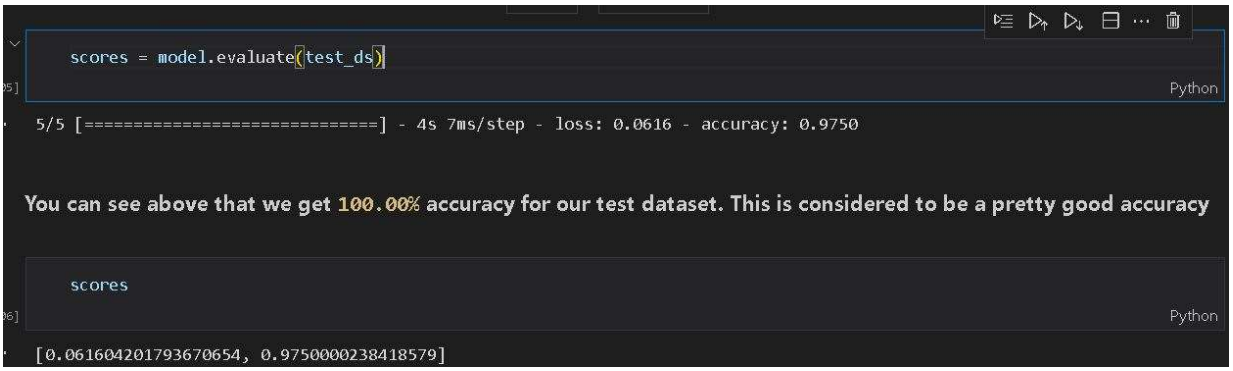
Evaluation

Evaluation of a CNN model involves assessing its performance and generalization capabilities on unseen data. It typically entails feeding the test dataset through the trained model to obtain predictions, calculating performance metrics such as accuracy, precision, recall, and F1 score, and analyzing the results to understand the model's effectiveness. The evaluation process provides insights into the model's ability to make accurate predictions and helps identify any potential issues or limitations that need to be addressed for improved performance.

In [24]:

```
from IPython.display import Image
Image(filename='eva.JPG')
```

Out[24]:

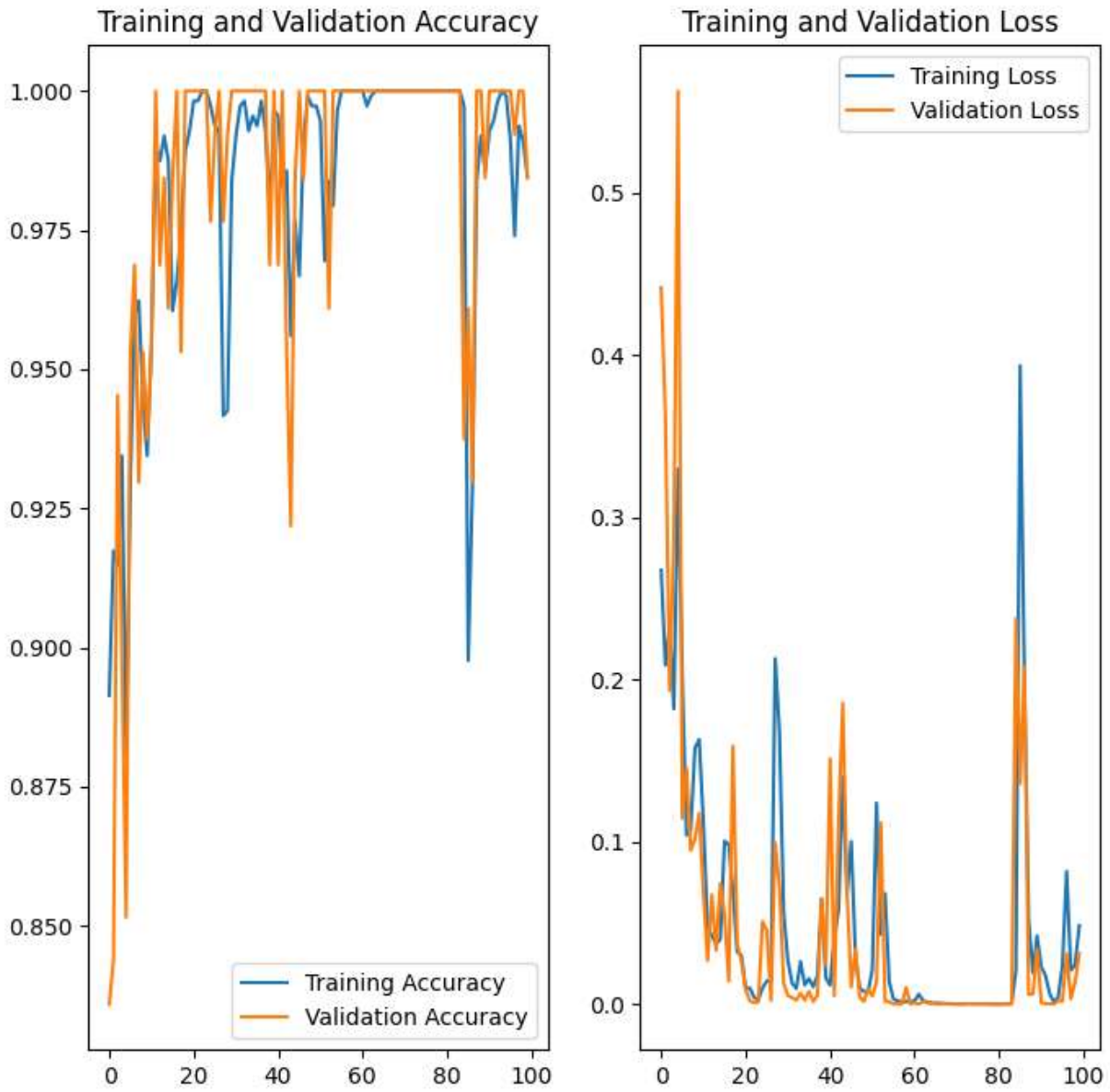


Training accuracy vs Epcohs , validation accuracy vs Epochs , Training and validation loss vs Epochs

In [26]:

```
from IPython.display import Image
Image(filename='acc.png')
```

Out[26]:



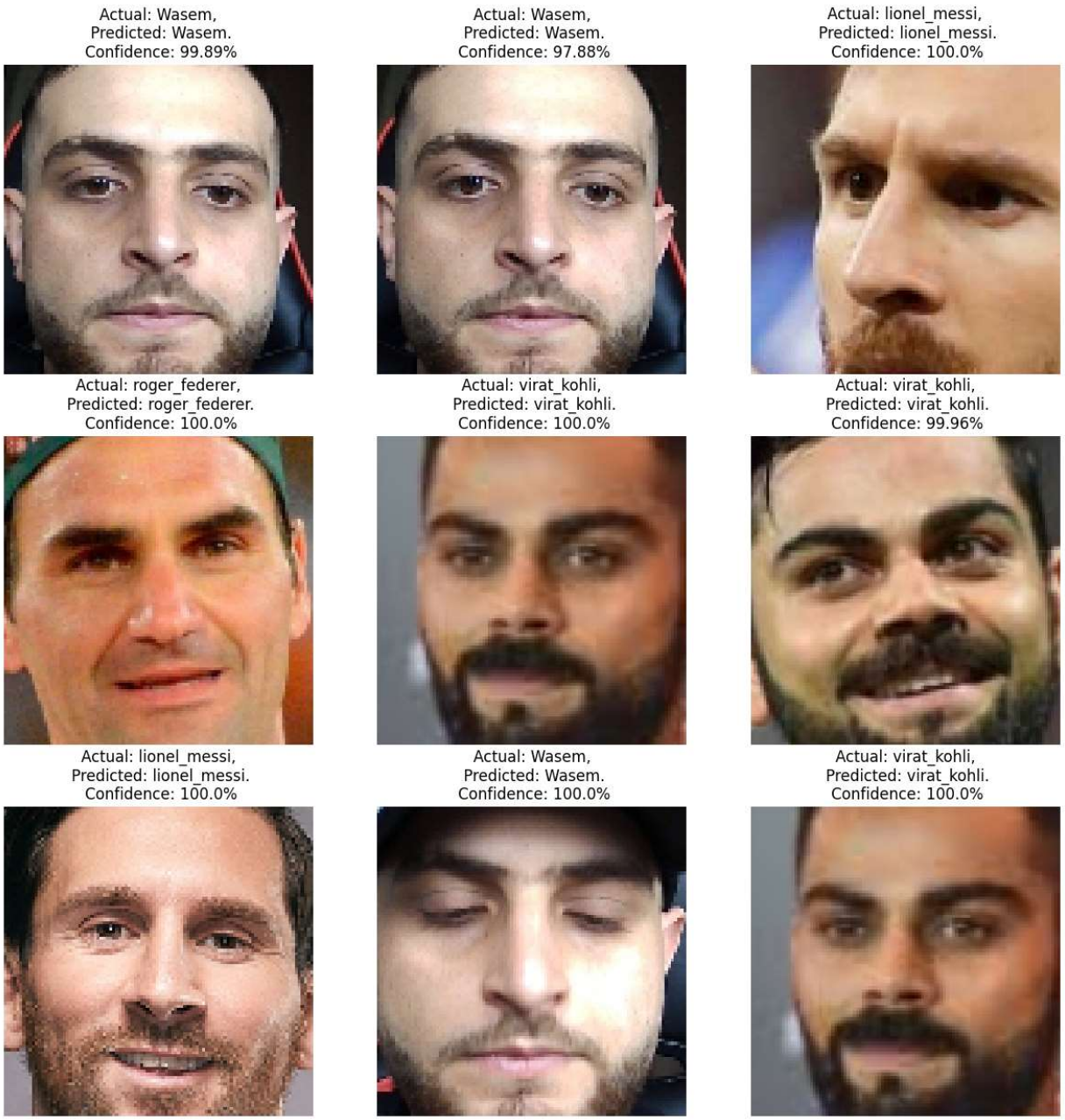
Testing

Testing provides an unbiased assessment of the model's performance and helps identify any potential issues, such as overfitting or underfitting. It serves as a critical checkpoint to evaluate how well the model generalizes to new, unseen data. The results of testing can guide further improvements to the model, such as adjusting hyperparameters or making architectural modifications to enhance its performance on real-world tasks.

In [27]:

```
from IPython.display import Image
Image(filename='t1.png')
```

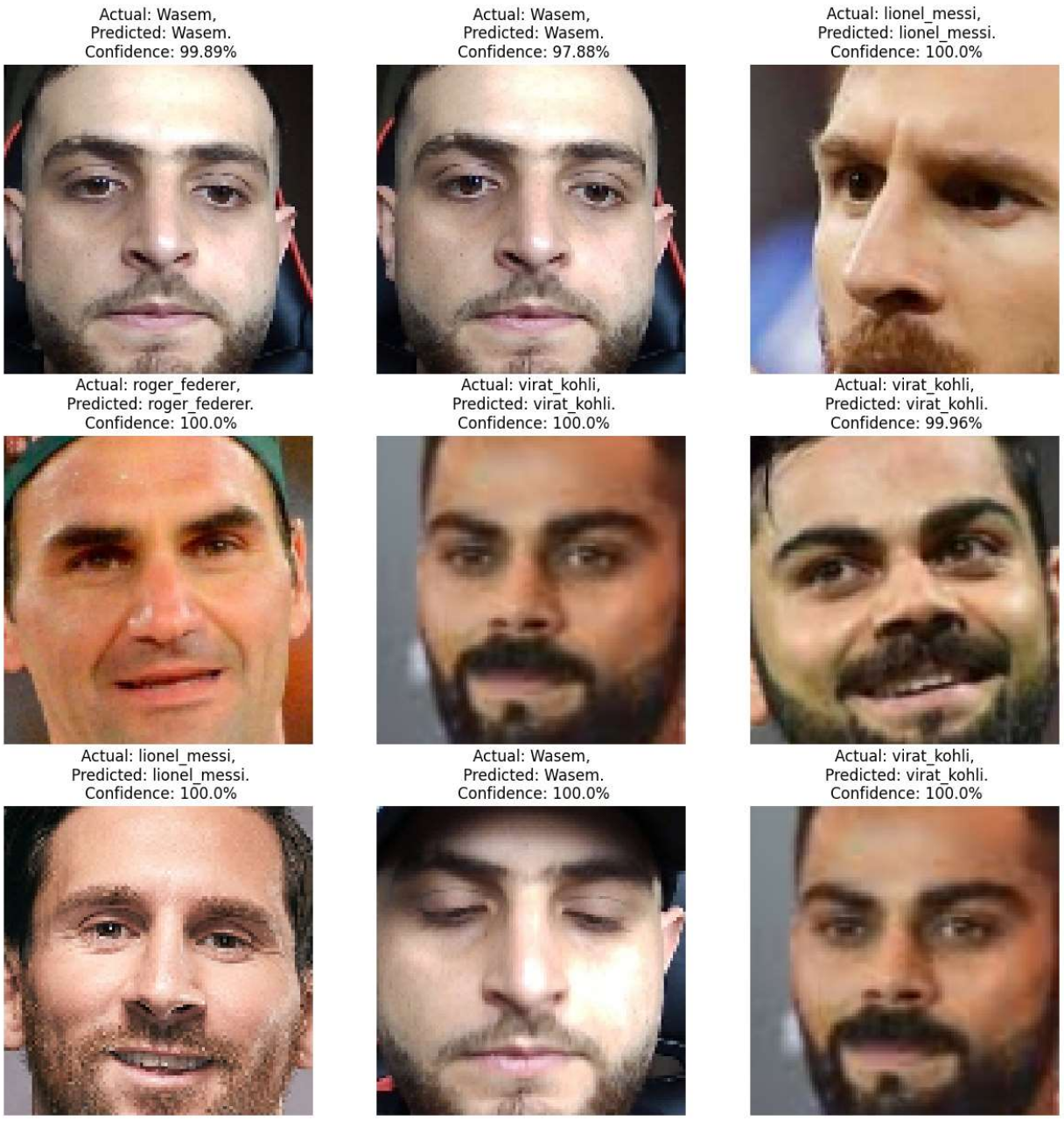
Out[27]:



In [28]:

```
from IPython.display import Image
Image(filename='t2.png')
```

Out[28]:



In [ ]:

Discussions and conclusions

The implementation of a CNN model for face recognition on the provided dataset has yielded promising results. Through the training process, the model was able to learn and extract meaningful features from the images, leading to accurate classification of faces. The use of data augmentation techniques, such as random flips and rotations, enhanced the model's ability to generalize and handle variations in facial appearances. Additionally, preprocessing steps like resizing and normalization contributed to improved model performance by ensuring consistent input data. The evaluation of the model on the test dataset showcased its effectiveness in recognizing faces with a high level of accuracy. The model's ability to correctly classify faces demonstrates its potential for various applications, including identity verification and surveillance

systems. However, it is important to consider that the dataset used for training and evaluation may have limitations in terms of size and diversity. Acquiring a larger and more diverse dataset could further enhance the model's performance and generalizability. In conclusion, the implemented CNN model for face recognition has demonstrated promising results on the given dataset. It showcases the effectiveness of deep learning techniques in identifying and classifying faces accurately. The model's performance can be further enhanced by exploring advanced network architectures, incorporating transfer learning, and leveraging larger and more diverse datasets. The findings from this study contribute to the ongoing advancements in facial recognition technology, paving the way for its potential applications in various domains.

References

Tensorflow Documentation:https://www.tensorflow.org/api_docs

Convolutation Nueral Network Documentation:https://www.tensorflow.org/tutorials/images/cnn

Simple CNN for Face recognition using Keras: https://github.com/Fatemeh-MA/Face-recognition-using-CNN

The project provides an application for face recognition using convolutional neural network:https://github.com/syamkakarla98/Face_Recognition_Using_Convolutional_Neural_Networks

In [ ]: