Aim: To perform Face detection on Video

Objective: Performing face recognition Generating the data for face recognition Recognizing faces preparing the training data Loading the data and recognizing faces.

Theory:

Generating the data for face recognition:

Creating a dataset of facial pictures from diverse sources, such as photos or video frames, is normally how face recognition data is generated. To guarantee reliable model training, this dataset should include a wide variety of people, expressions, lighting settings, and backgrounds. To increase dataset diversity and enhance model generalization, data augmentation techniques including rotation, scaling, and introducing noise may also be used.

Recognizing faces:

The automated technique of locating and differentiating human faces in pictures or videos is referred to as "recognizing faces." This entails examining facial features like the eyes, nose, and mouth to identify people based on characteristics like age, gender, or emotions. Although commonly utilized for security, authentication, and personalization, facial recognition technology also presents privacy and ethical questions.

Preparing the training data:

Training data preparation entails a number of crucial processes. Gathering a broad collection of photographs or videos pertinent to the objective is the first step in data collection. Following that, these images are annotated with ground truth data such as item bounding boxes or semantic segmentation masks.

Data augmentation is the process of performing modifications, such as rotation or scaling, to a dataset to erroneously increase its size and diversity. Last but not least, data splitting separates the dataset into training, validation, and test sets to efficiently train and assess machine learning models.

These actions are necessary for creating reliable computer vision models.

Loading the data and recognizing faces:

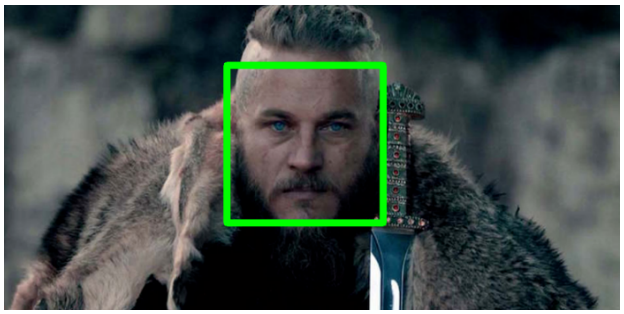
The steps involved in gathering and preprocessing picture or video data, then employing facial detection and recognition algorithms, are known as "loading the data" and "recognizing faces." In order to enable applications like facial authentication, emotion detection, or person identification, this often entails processes like data ingestion, image scaling, feature extraction, and the use of deep learning models to recognize and classify faces within the data.

Code

```
!pip install opencv-python

import cv2

import datetime

from google.colab.patches import cv2_imshow

face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +

'haarcascade_frontalface_default.xml')

video_path = '/content/sample.mp4'

cap = cv2.VideoCapture(video_path)

while True:

   ret, frame = cap.read()

   if not ret:

     break

   gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

   faces = face_cascade.detectMultiScale(gray, scaleFactor=1.3,

   minNeighbors=5, minSize=(30, 30))

   timestamp = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

   cv2.putText(frame, timestamp, (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.7,

   (0, 0, 255), 2)
```

```
for (x, y, w, h) in faces:
  cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)
cv2_imshow(frame)
if cv2.waitKey(1) & 0xFF == ord('q'):
  break
```

Output



Conclusion :

Face recognition on video entails recognizing and monitoring faces over a number of frames to allow for real-time analysis of facial expressions and movements. To effectively match and identify faces, face recognition relies on collecting and arranging data about people. The preparation of training data is essential for the creation of reliable recognition models. This data can be loaded and processed to effectively recognize faces in a variety of applications, such as security systems and social media tagging. These actions work together to create and implement efficient facial recognition systems.