Real Time Facial Emotion using CNN and Streamlit

Rahul Jha
Data Science Trainee
Almabetter,Banglore

1.Abstract:-

Emotion recognition is a major area of study. It has a wide range of uses. Two of its most exciting applications are robotic vision and interactive robotic communication. Human emotions may be detected using both verbal and visual modalities. Facial expressions are a fantastic way to discern someone's emotions. This research offers a real-time emotion detection technique and its use in robotic vision applications. The suggested technique has four stages: preprocessing, key point creation, key point selection, angular encoding, and classification.
The fundamental concept is to use Haar cascades, a real-time deep learning approach, to create crucial points. Furthermore, the created key points are encoded using a well constructed sequence of mesh generator and angular encoding modules.

Facial expression of human

2. Problem Statement:-

Detecting facial expression is one of the most difficult tasks to do since humans have more than 46 muscles that affect face expression.
The goal of this research is to recognise faces in real time and detect images in order to discern emotions.

The two tasks we must do here are:

1.Face Recognition
2.Extraction of the Face.

3. Introduction

Facial expressions are employed in nonverbal communication as well as to identify persons and play a vital part in emotion identification. They are second only to tone of voice in terms of importance in daily emotional communication.

They are also a feeling indication, helping a guy to indicate his emotional condition.
People can discern a person's emotional condition instantaneously. As a result, facial expression information is commonly employed in automated emotion identification systems.

The proposal's purpose is to recognise seven fundamental emotional states based on facial expressions: neutral, pleasure, surprise, anger, sorrow, fear, and disgust.

4. Methodology

According to research and many publications, there are several methods for detecting objects and people in a picture. Haar cascades and Deep Learning face detectors are two techniques that can be applied in our circumstance. The Haar cascades approach is used to recognise a face in a picture. If the face exists in the image, this procedure is the quickest way to locate it. Deep learning face detectors will improve accuracy. Both of these approaches will be used to the dataset to see which one performs better and produces the best results as the project progresses.

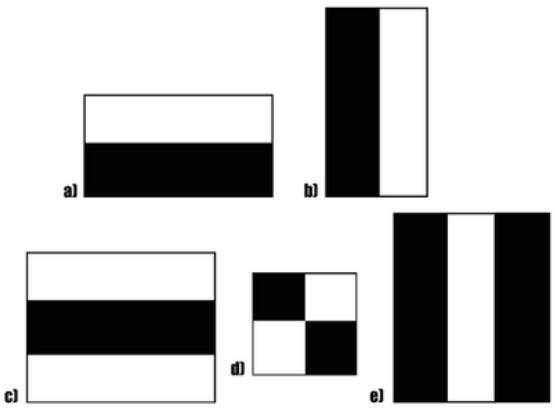
Although the Haar cascades approach is quicker, the deep learning face detector is more accurate. Because we want to identify areas with larger variances in viewing angles, using the deep learning detector makes more sense. The Haar cascades will suffice if the faces are oriented "straight on." As the project proceeds, both of these strategies will be utilised to discover which works better and offers better outcomes.

The two main application are:-
1.Face Identification.
2.Emotion Recognition

4.1. Face Identification

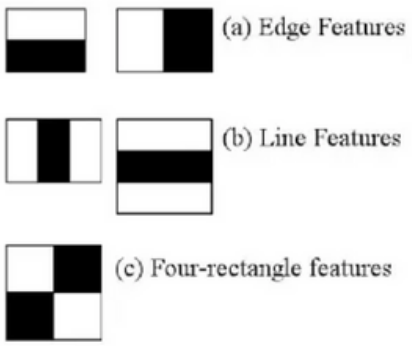
The Haar Cascade Object Detection Algorithm is used to recognise faces in pictures or real-time video. Viola and Jones introduced edge or line detection features in their 2001 study "Rapid Object Detection Using a Boosted Cascade of Simple Features."

The algorithm is given a large number of positive photos with faces and a big number of negative images without faces to train on. The model created as a consequence of this training may be found in the OpenCV GitHub repository.

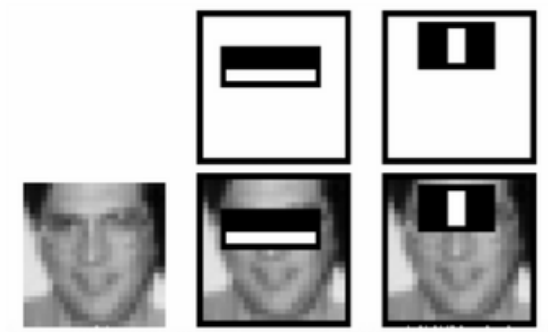


A sample of Haar features

The purpose is to compute the sum of all image pixels in the haar feature's darker region and the sum of all image pixels in the haar feature's brighter area. Then investigate what distinguishes them. The haar value will be closer to one if the image features an edge dividing dark pixels on the right from bright pixels on the left.
That is, if the haar value is near to one, we claim an edge has been discovered. In the above case, there is no edge since the haar value is distant from 1.



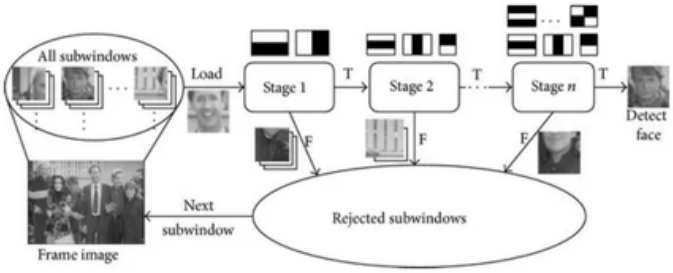
The first rectangular feature in the figure below calculates the difference in intensity between the eye and cheek areas of the face. The second rectangular feature measures the intensity difference between the two eye areas and the nasal bridge.
Using these rectangular features across an image, we can generate hundreds of feature points.



Facial identification using cascade

However, computing the total of pixels in white and black regions across the picture may be an expensive process, especially for large images, thus the authors also presented an integral image approach that can achieve the same computation by executing the operations on only four pixels.
We may utilise hundreds of tagged photos to create a face identification classifier by converting them into the HAAR cascade feature map and training it using a machine learning model. To train the classifier in a robust manner, we must ensure that we have a decent mix of pictures with and without faces. The author used Ada Boost in the study because it produces the best results.

Face Detection using HAAR Cascade Step by Step:-



- Step 1: The image (that has been sent to the classifier) is divided into small parts (or subwindows as shown in the illustration).

- Step 2: We put N no of detectors in a cascading manner where each learns a combination of different types of features from images (e.g. line, edge, circle, square) that are passed through. Supposedly when the feature extraction is done each sub-part is assigned a confidence value.

- Step 3: Images (or sub-images) with the highest confidence are detected as face and are sent to the accumulator while the rest are rejected. Thus the cascade fetches the next frame/image if remaining and starts the process again.

4.2. Emotion Recognition

Neural systems have been around for a long time. Because of neural networks, machine learning and artificial intelligence have altered considerably in the last decade. They are delivering high-quality, exact outcomes. However, training a neural system from scratch is incredibly time consuming, stressful, and expensive. As a result, we are employing a pretrained model, i.e. transfer learning.
It is a pre-trained model that can better analyse and learn the image.

Types of transfer learning model-

ALEXNET

Alex Krizhevsky and a colleague built a neural network using 1.2 million photos including over a thousand classes, which took nearly two weeks to train. It has 60,000,000 variables, 650,000 neurons, and 630,000,000 connections. It is an eight-layered network with seven hidden layers and one output layer. It has 5 convolutional layers that are followed by maxpooling layers. This network contains three completely functional

It is based upon supervised learning [6]. Fig 1 shows the architecture of AlexNet.

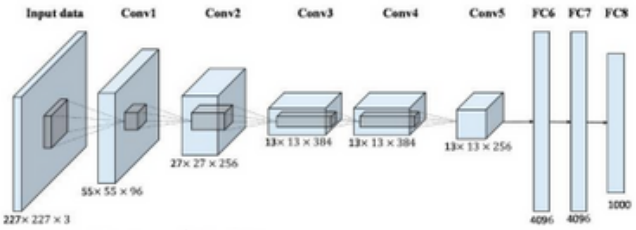


Fig. 1. Architecture of AlexNet

1) Transfer learning Feature-Based Approach:

Despite having been trained on 1.3 million object photos and being able to categorise images into hundreds of distinct item categories. However, because neural networks operate in a manner comparable to the design of the human brain, Humans have the potential to learn hundreds of distinct categories throughout their lifetimes only by looking at a few example items or things. Humans are said to develop this skill through accumulating prior information and using it to learning new objects or things. This is commonly known as Transfer of Learning [18]. The neural network might follow the same transfer learning process. However, we still require some data to train the network to do task-specific categorization. do classification specific to our tasks.

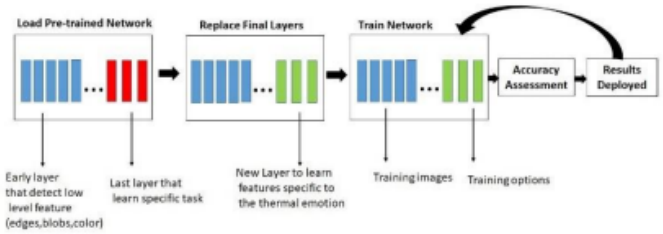
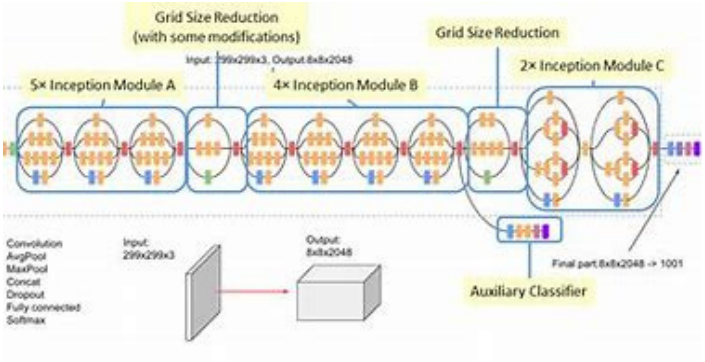


Fig. 2. Transfer Learning Feature-Based Approach

- **INCEPTION V3**

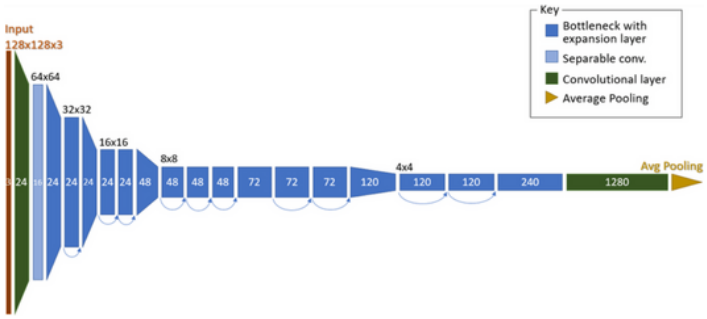
Inception v3 is an image recognition model that has been shown to attain greater than 78.1% accuracy on the ImageNet dataset. The model is the culmination of many ideas developed by multiple researchers over the years. It is based on the original paper: "Rethinking the Inception Architecture for Computer Vision" by Szegedy, et. al.
The model itself is made up of symmetric and asymmetric building blocks, including convolutions, average pooling, max pooling, concatenations, dropouts, and fully connected layers. Batch normalization is used extensively throughout the model and applied to activation inputs. Loss is computed using Softmax.



Architecture of inception v3

- **MOBILENET V2**

MobileNetV2 is a convolutional neural network architecture that seeks to perform well on mobile devices. It is based on an inverted residual structure where the residual connections are between the bottleneck layers. The intermediate expansion layer uses lightweight depthwise convolutions to filter features as a source of non-linearity. As a whole, the architecture of MobileNetV2 contains the initial fully convolution layer with 32 filters, followed by 19 residual bottleneck layers.



Architecture of Mobile Net V2

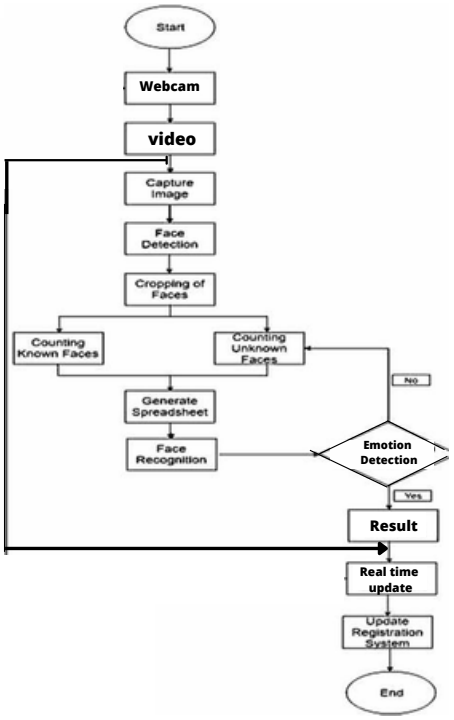
The MobileNet v2 architecture is based on an inverted residual structure where the input and output of the residual block are thin bottleneck layers opposite to traditional residual models which use expanded representations in the input. MobileNet v2 uses lightweight depth-wise convolutions to filter features in the intermediate expansion layer. Additionally, non-linearities in the narrow layers were removed in order to maintain representational power.

5.Frontend with Streamlit

Streamlit is an open-source python framework for building web apps for Machine Learning and Data Science. We can instantly develop web apps and deploy them easily using Streamlit. Streamlit allows you to write an app the same way you write a python code. Streamlit makes it seamless to work on the interactive loop of coding and viewing results in the web app.

- **Streamlit runs the python script from top to bottom.**
- **Each time the user interacts the script is a rerun from top to bottom.**
- **Streamlit allows you to use caching for costly operations like loading large datasets.**

- **Workflow of Web App**



Streamlit is a great framework for data scientists, machine learning researchers and developers, and streamlit-webrtc extends it to be able to deal with real-time video (and audio) streams.
It means we can implement your computer vision algorithms only in Python on server-side and users can use them as web apps with real-time video inputs from their webcams or smartphones.

- **Streamlit webtc**

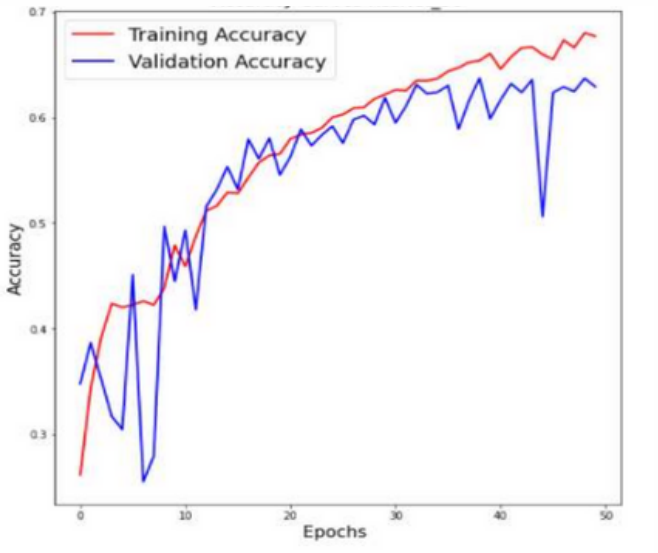
Streamlit-webrtc , which sends and receives video (and audio, but it's only partially supported now) streams between frontend and backend via WebRTC.
This should be helpful for creating, for example, performant real-time computer vision WebApps.

This week I tried Steamlit, found it interesting and useful, and then wanted to run real-time computer vision apps on Streamlit, on which users can try some computer vision models with video input from their webcams.
So I tried to develop a component to achieve it using WebRTC. Fortunately, there is a great WebRTC library for Python, aiortc 105.

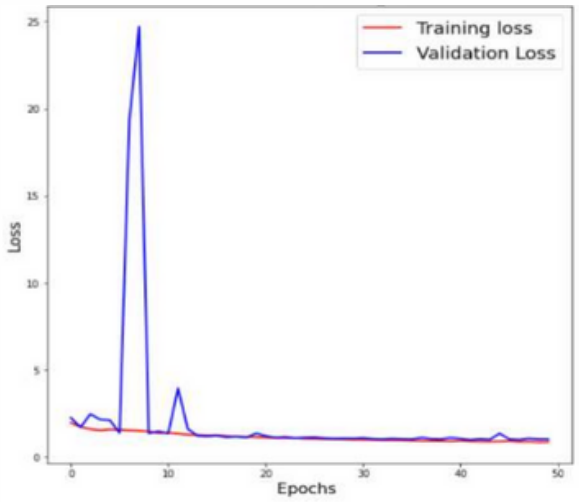
One interesting thing about this component is that the input video streams are sourced from users' webcams and transferred to the server-side Python process via the network. Therefore, the server does not need the access to the camera, unlike the usual approach using OpenCV (cv2.VideoCapture). It means the Python code can be hosted on a remote server (actually, I hosted a sample app on Heroku and it worked, as stated below).

In addition, WebRTC provides good performance with sending and receiving video/audio frames.

6. Model Evaluation



- **The training gave the accuracy of 72.3% and val_accuracy of 63.8%.**
- **And in real life scenario it identifies great but lags between neutral and sad.**



- **The training loss is slightly higher than the validation loss for the first epoch**

7. Conclusion:-

- **Our model is giving an accuracy of 72.3% and validation accuracy of of 63.8% . It is robust in that it works well even in a dim light environment.**
- **The application is able to detect face location and predict the right expression while checking it on a local webcam.**

- The front-end of the model was made using streamlit for webapp and running well on local webapp link.
- Finally, we successfully deployed the Streamlit WebApp on Heroku and Streamlit Cloud , that runs on a web server.
- And we believe that through this model teachers can understand the students' perception during online classes and change the way of teaching if needed by understanding the students' motive.

8. Challenges

- Large dataset folder containing lot of images to handle
- Model training take lots of time and system resource
- Continuous Runtime and RAM Crashes many time till we get the best model
- Code to access webcam using opencv
- Carefully tuned Hyper parameters .
- Size of web app should be lesser than 500 mb.