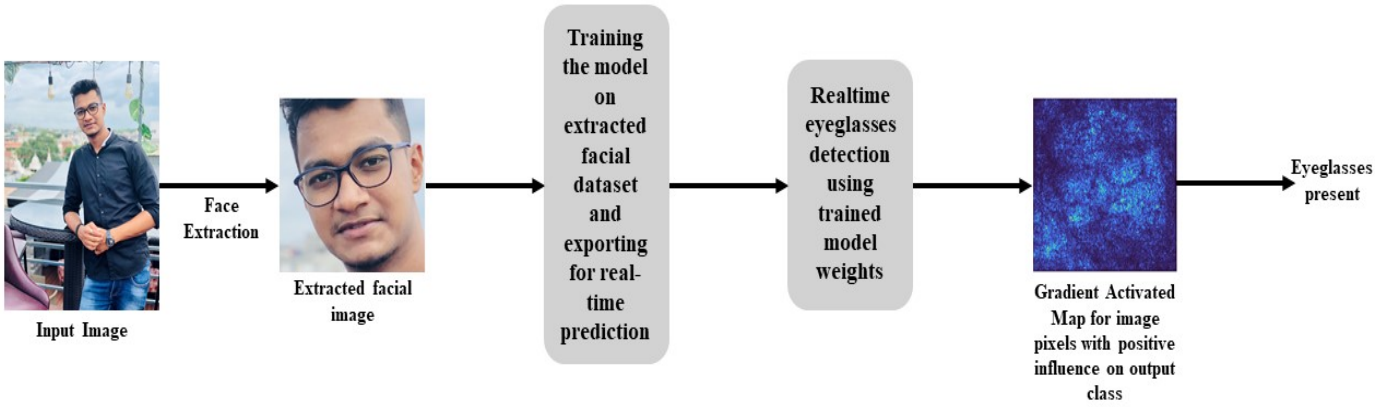
METHODOLOGY



The input images need to be pre-processed before being used for training. The first step as shown in pipeline in Figure 2 is detecting and extracting the face in each input image, which is done by using a technique based on the renowned Voila-Jones Face Detection algorithm called the Haar-Cascades. OpenCV supports object recognition using Haar features which can recognize faces in an image and return a rectangular region of interest (ROI) which can be cropped and saved as a new image. Using this approach, a new dataset is thus generated consisting of close shots of faces in the base dataset. This new dataset is then preprocessed based on the requirements of the transfer learning model i.e., resizing to 299,299 and normalizing each pixel.

Transfer Learning is nowadays the most popular machine learning technique used for training classification models wherein a pre-trained model is used which is trained on significantly large datasets with high accuracy and then refactored on our concerned dataset to fit more. A typical transfer learning algorithms involves using a pre-trained model and recycling a portion of its weights while reinitializing and updating the weights of shallower layers. The elemental instance of this approach requires modifying or updating the weights on the final classification dense layer of a fully trained neural network and using it to classify a different dataset.

For an input image passing through the first convolutional layer filters, almost all the features in the image are retained although some filters are still not active at this layer which are marked by solid-colored boxes. Comparing this with the filters of our last convolutional layer, it is inferred that the activations become abstract being less interpretable visually. At this layer certain higher-level concepts are encoded like specific border, corners, and angles. Next step involves training of the model consisting of 262,146 trainable parameters over a total of 22,064,930 parameters present in the proposed model. Multiple models were trained on the batch image dataset while varying the values of different hyper parameters to find the optimized version of proposed model and hence fine tuning the model. After training all these models, the optimized model was exported and saved as .h5 files which will be used for loading the model to make predictions in real-time using a custom framework.

After this the following hyper-parameters of the model were optimized by selecting the best performing value for each parameter:

1. Train-Test Split Ratio
2. Optimizer Function
3. Learning Rate
4. Activation Function (for fully connected layer)

For real time detection framework, the model trained on the fine-tuned hyper parameters is used and loaded through saved model in .h5 file. This loaded model is used to make predictions on frame captured in real-time by OpenCV.