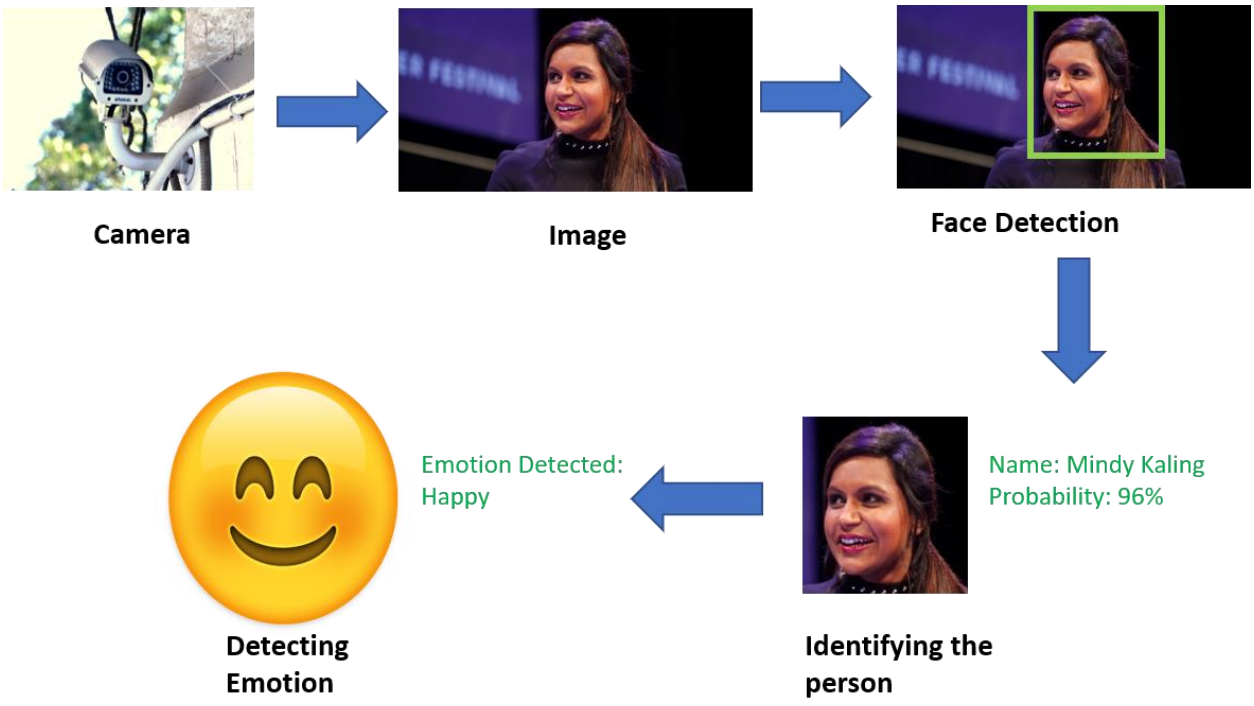
Face-Recognition-and-Emotion-recognition-for-Sentiment-Analysis

CS GY 9223: Deep Learning Fall 2020 Project Repository containing the codes and documentation and data of Face Recognition and Emotion Detection project done as a part of the deep learning course

A high level system flow is as follows:

image --> pre-process --> Face Detection model --> SVM Classifier --> Detected face output. image --> pre-process --> Emotion Detection model --> Detected emotion output



Final output --> Detected Face output + Detected emotion output.

Model Description: Below are the models used in this implementation. Facenet: - Face Recognition The facenet paper published by Google mentions the Zeiler & Fergus architecture and the Inception based one In the paper, the Inception based architecture is based on Google based GoogILeNet based Inception models. These models though have similar performance to the Zeiler & Fergus model have dramatically less computational needs. The Inception model has around 7.5M parameters with 1.6B of FLOPS(a metric for no of computations needed) compared to the Zeiler& Fergus based

model that has 140M parameters and 1.6B FLOPS. In other variations, the FLOPS of the inception model come down to around 550M. The Keras implementation of Facenet uses the inception-resnet-V1 based architecture. It has 447 layers. The json representation of the model is included for further understanding. The Facenet network is trained via a triplet loss function that encourages vectors for the same identity to become more similar (smaller distance), whereas vectors for different identities are expected to become less similar (larger distance). The focus on training a model to create embeddings directly (rather than extracting them from an intermediate layer of a model) was an important innovation in this work. These face embeddings are then used as the basis for training the classifier systems on our dataset. The FaceNet model can be used as part of the classifier itself, or we can use the FaceNet model to pre-process a face to create a face embedding that can be stored and used as input to our classifier model. This latter approach is preferred as the FaceNet model is both large and slow to create a face embedding.

SVM-(Linear)

We are using the Facenet model to get the embeddings from the input image and It is very common to use SVM while working with normalized face embedding inputs. This is because the method is very effective at separating the face embedding vectors. We can fit a linear SVM to the training data using the SVC class in scikit-learn and setting the 'kernel' attribute to 'linear'.

VGG Face - Resnet 50 - Emotion Detection Keras has a VGGFace library that provides us three provides three networks to use- the VGGFace (based on VGG16 architecture), Resnet(Based on Resnet 50 architecture) and Setnet(based on Setnet 50 architecture. We have used the Resnet 50 model in our use case. The Resnet architecture consists of a skip connection where the weights from a layer are connected to the next layer as well as next to next layer. This helps primaily to solve the problem of vanishing gradients during backpropogation.It consists of 5 stages each with a convolution and Identity block. Each convolution block has 3 convolution layers and each identity block also has 3 convolution layers. The ResNet-50 has over 23 million trainable parameters. We are using the Resnet-50 model for emotion detection and add to it three layers and ReLu activation function in and finally softmax activation function layer as a classifier at the end. We have used Adam optimizer and categorical cross entropy loss while training the model.The input to this model is through a custom data loader with a batch size of 128.0

Given an image, it goes through the following steps in our system.

We input an image to the system. Its format can be .png or .jpeg The face recognition performs face detection by cropping the image around the face and extracting only the face to be recognized from the entire image using bounding boxes. We resize the pixels of the extracted image to the model size which is 160*160 in our case. This process is labelled as face alignment and standardisation. We use a prebuilt keras model for face recognition built on the facenet framework using the keras api available as an h5 file. We train the this model on the train dataset that has images belonging to 5 different classes/people.It converts each face into an embedding which means that we extract high-quality features from each face. and produce a 128 dimensional vector representation of the same. This vector is normalized and the vector size can be changed from 128 to 1024. Faces with similar face embeddings have vectors that are closer to each other. \ We are using Linear SVM as the classifier model. The face embeddings that are similar to each other are clustered together and the model predicts the right class that the face belongs to. For emotion detection the model requires images to be of 48x48 pixel and a grayscale image. So we did some data preprocessing like image resizing, realignment etc. agian on our input image so that it can be used in our emotion detection model. We then predict the emotions that are depicted in the image.

Results of our model: The face recognition model predicts the right class that the face belongs to with an accuracy of 90+% every time. The emotion detection model gives accuracy of 67% By varying the learning rate hyper parameter we notes that the learning rate that gives the best results using Adam optimizer is 1e-4. According to Table 2.We also notices that beyond five epochs the accuracy does not vary a lot so we did early stopping after 10 epochs To get more insights about our results we plotted a confusion matrix between the predicted and true labels for our test dataset. We found that we could predict “Happy” emotion correctly 88 % of the times which was the highest and we mis predicted “Fear” emotion the most. Fear was mostly classified as Sad and disgust was classified as angry which can be understandable that these emotions could get interchanged.