Technical Planning (Face Identification based Attendance System) Group-16

Attendance is an important part of classroom evaluation. In every class, attendance is usually checked by teacher but sometimes teacher may miss someone. The traditional attendance system consists of every student signing on a sheet of paper against their names where the sheet is passed around to each student or biometric based system or tag-based attendance system. The concept of face recognition-based attendance system is to give a computer system the ability of finding and recognizing human faces fast and precisely in images or videos.

Face recognition can be done in two ways. One way of doing this is by training a neural network model, which can classify faces accurately. The second and best way of doing this is using one shot learning method by using a pretrained model to identify and extract basic features of the face and then add new images (of student only three required to train on the images). The goal of one-shot learning is to learn information about object categories from one, or few training images. We have decided to implement this using one shot learning technique.

Using Siamese network, one-shot learning technique can be implemented. It's nothing but, two identical neural networks with exact same weights, but taking two distinct inputs. These networks are optimized based on the contrastive loss between their outputs. This loss can vary and depends on inputs. If inputs are similar, then loss is minimal, and it maximizes when inputs are different. So, in this way, optimized Siamese networks can differentiate between their inputs. The network would learn to encode face image 128 bit encoding which would be used to identify whether the image of the person is present in the database or not. This way we could uniquely identify the person entering the class and update his /her attendance.

Tools:

We would first use Keras library for initial prototyping of the architecture to know weather the architecture would work functionally. Keras is opensource neural network library used to prototype the neural networks as it is simple to use and support other low-level libraries like TensorFlow etc. After we are through to the architecture we would like to use TensorFlow library to further fine tune the network if needed .

TensorFlow is an open source software library for high performance numerical computation. Its flexible architecture allows easy deployment of computation across a variety of platforms. Tensor flow has been around from 2017. This tool has thus evolved over time and has really helped developers to build their deep learning solutions effectively in short span of time. There are bunch of high level as well as low level API's, we are using some of them to build our network. This can be useful for the following model which we are developing for this system.

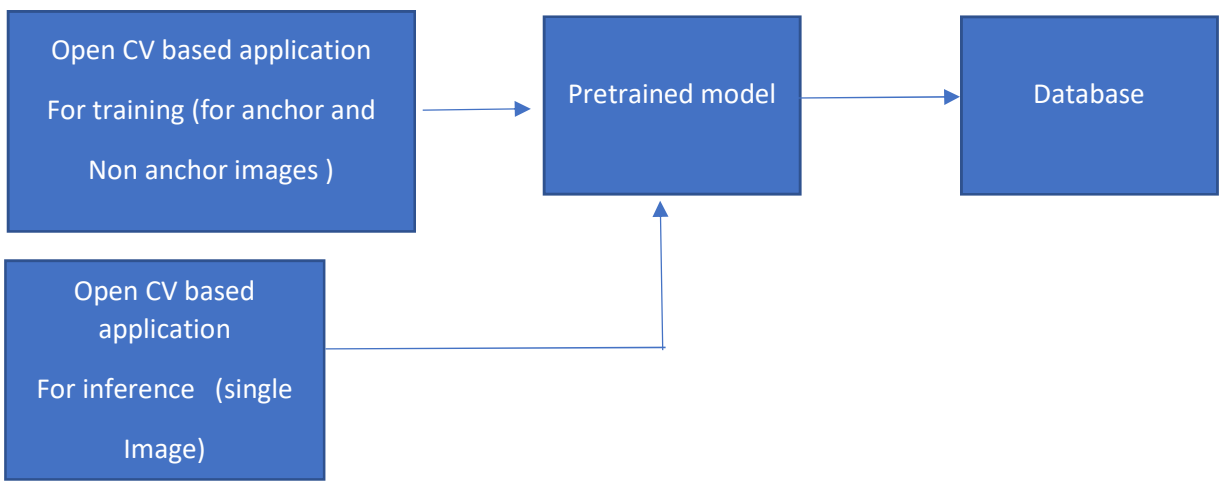
To build the facenet model which is a one-shot model, that directly learns a mapping from face images to a compact Euclidean space where distances directly correspond to a measure of face similarity. Here comes the concept of triplet loss. Triplet loss would help the Siamese network learn the parameters to understand the level of similarity or dissimilarity between two images. We will be using 3 images such as anchor image, positive image and negative image. By using the difference between anchor and positive image as well as anchor and negative image, we decide the loss. To achieve our goal, we would be implementing the triplet loss function. The steps for the triplet loss functions implementation would be:

- Compute the distance between the encodings of "anchor" and "positive"
- Compute the distance between the encodings of "anchor" and "negative
- Compute the formula per training example $
- Compute the full formula by taking the max with zero and summing over the training

We are not going to train any neural network model since it would need lot of face images. Rather we are going to use the transfer learning by using a pretrained model to map face images into 128 bits encoding as discussed before. And Lastly, we will use these encoding to perform face verification and recognition. The network architecture we are going to use for this is inception model. The input to this model is tensor of 96*96*3 image and output is (m,128) which is the encodings for the image. Each of the student would have single 128 bit encoding for their face. Thus to uniquely identify the student we would check encoding of each of the student in the database against the live student face and measuring the distance between the encoding (1: k problem). The distance we would use is L2 distance. We would then use threshold to verify the student. This threshold would be an empirical calculation with respect to the network.

The system level implementation would also include framework which would help us to add new images to the database so that a new user can be added. We are planning to use Linux based system for this as it is easy to implement and integrate the system level function in Linux. We would use open CV libraries to capture the image and do the preprocessing of the image for the pre-processing which is required to pass the face image to the network.

The overall system architecture would look like the following:



Work distribution:

Sr.No	Name of student	Associated Work
1	Jasmine Corea	**Paper Reading**: • Face net, • Inception Network **Implementation:** • Triplet Loss Function • Verification using Keras • Evaluating pretrained face detection network to load the weights.
2	Adarsh Sawant	**Paper Reading**: • Face net, • Keras libraries **Implementation:** • Triplet Loss Function Verification using TensorFlow • System level implementation using open CV to capture the live feed. • Understating the Keras libraries and knowledge transfer to the team
3	Ying Wang	**Literature Reading:** • Deep face net • Tensor flow **Implementation** • Understanding the TensorFlow library and knowledge transfer across the team. • Setting up the database from the encoding and code to add the new image to the database
4	Prashant Desai	**Literature Reading:** • Deep face net • Inception Network **Implementation:** • Implementation of inception architecture model in TensorFlow • Setting up Tensor flow library • System level implementation

References:

- Florian Schroff, Dmitry Kalenichenko, James Philbin (2015). FaceNet: A Unified Embedding for Face Recognition and Clustering
- Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, Lior Wolf (2014). DeepFace: Closing the gap to human-level performance in face verification
- The pretrained model we use is inspired by Victor Sy Wang's implementation and was loaded using his code https://github.com/iwantooxxoox/Keras-OpenFace.
- Our implementation also took a lot of inspiration from the official FaceNet github repository https://github.com/davidsandberg/facenet