Attendance taking has become an integral part in today's collegiate world. It is necessary to inculcate a sense of discipline and responsibility in young minds to motivate them to succeed. Currently a significant amount of time is spent by faculties to take record of students present in the class. There is a time-accuracy tradeoff while recording attendance manually, namely if the faculty tries to speed up the process then there is possibility of false positives (read "proxy attendance") and if the faculty tries to maximize accuracy, then he/she will stand to lose a lot of time in each class just taking attendance. This situation is further exacerbated in places such as India, where the average class sizes frequently exceed over 60 students per class. We propose a Deep Learning based solution to automatically mark attendance of students using face recognition techniques.

There are many issues with employing face recognition over a dataset which is prone to problems such as occlusion, illumination changes, pose changes etc. To combat this, we come with an end to end architecture as shown in figure 1.

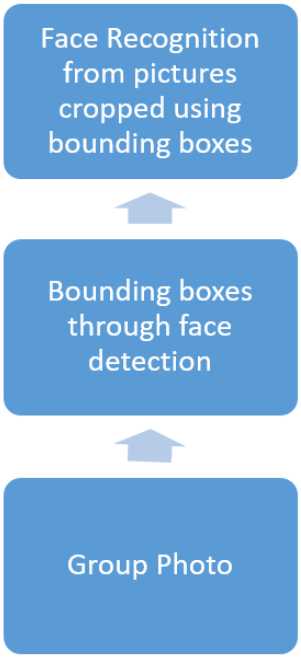

Figure 1. Architecture of the proposed effort

The input is fed from multiple cameras placed in the classroom, and first face detection is done to create bounding boxes on the basis of which we will crop the images to feed into the face recognition network.

There are multiple challenges involved in creating this pipeline, including creating a sufficiently fast face detection networks which is capable of detecting multiple bounding boxes (to a range of over hundreds of bounding boxes), creating a sufficiently accurate face detection network and integrating them together to form a seamless pipeline.

A good and sufficiently fast object detection framework is SSD (Single Shot Multi-box Detector), which can achieve near real time performance with a mean average precision of bounding box overlap in the range of 75%.

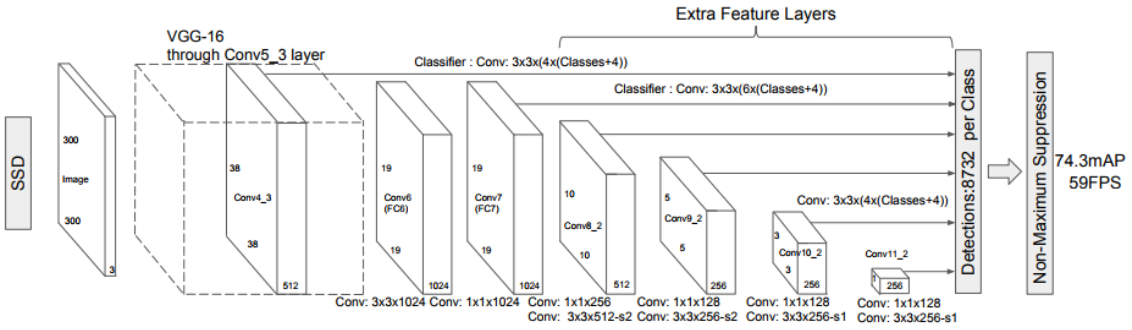

Fig 2. SSD Architecture

The SSD algorithm works by calculating a category score and offsets for fixed set of bounding boxes using small convolution filters applied to feature maps. Since the default bounding boxes are defined, this dramatically speeds up the processing time. Having different sizes of feature maps helps the network recognize bounding boxes of different dimension and different aspect ratios.

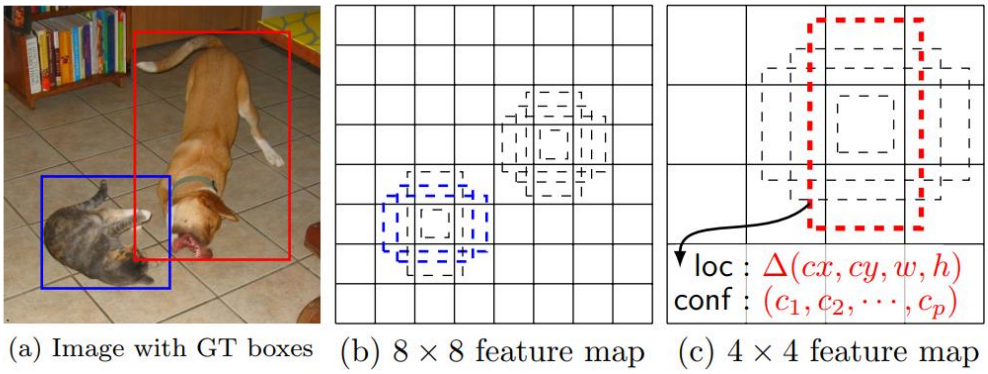

Fig 3. Feature Maps and Bounding boxes in SSD

We crop the images on the basis of bounding boxes generated by the SSD network, to get the faces in the dataset. Now, we pass it to the Face Recognition pipeline. We chose VGG 16 network because it provides a high accuracy, reaching to the tune of ~94% in validation tests, while at the same time managing low time for inference.

The VGG 16 network employs 16 layers of convulsion filters and pooling layers to achieve this feat.

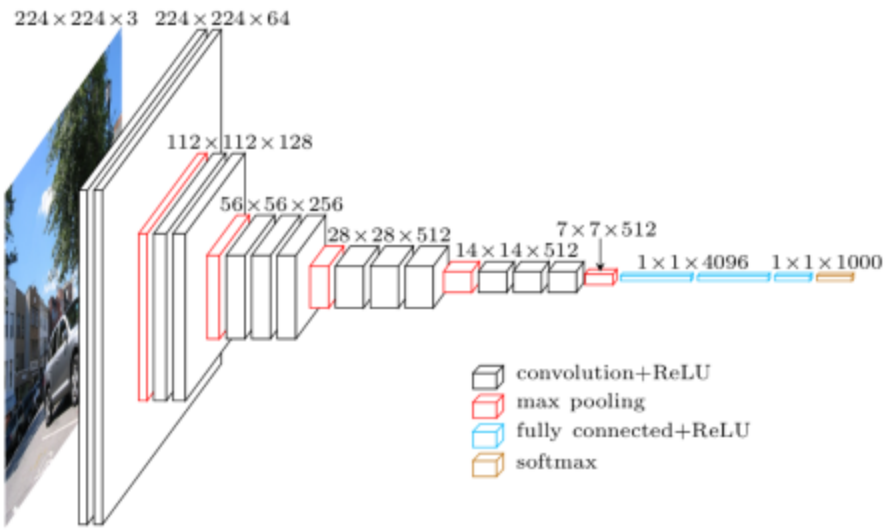

Fig 4. VGG 16 Architecture

There are multiple problems with such an architecture, namely the huge amount of training time, and the need of large datasets to get sufficiently high accuracy. For this we use Transfer Learning mechanism, where we take a pre-trained network with high accuracy and detach the last few layers, fitting in new layers more suited to our purpose. We freeze the top layers, and only training the newly added bottom layers for getting good results in a shorter period of time.

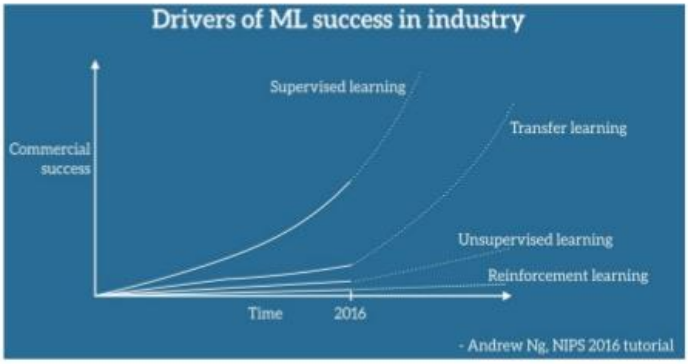

Fig 5. Transfer Learning

Using Transfer Learning, we take the models trained on extensive datasets and then retrain them to suit our purpose. We use model weights obtained from training COCO dataset, and then retrain them to fit WIDER dataset in the SSD network. WIDER dataset contains images annotated with appropriate bounding boxes, and also contains images which have illumination variance, occlusion, blur, pose changes and scale changes. These images help the dataset increase the prediction accuracy. We also retrain the dataset the VGG-16 model which is trained on LFW dataset (Lone-Faces in Wild Dataset), and then retrain them to fit Indian Actors dataset (this dataset will be replaced with a dataset of all images in the attendance repository, we use this dataset due to a lack of access to such a dataset).

Considering that each image captured by the camera has to be transferred from the camera unit to the remote server for processing, there is possibility of a network clog. To circumvent this, we propose a thick client based model in which significant inferencing is processed in-situ and only the results are pushed to the server for logging. This can be accomplished by setting up a mobile processing unit capable of running the neural networks remotely with sufficiently high speed to push the results to the server.

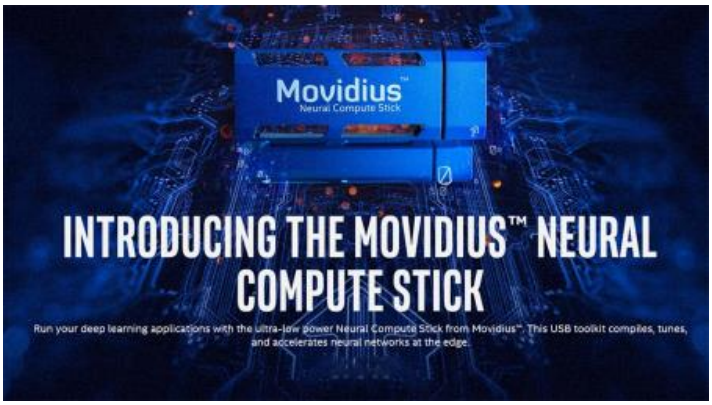

Fig 6. Introducing Movidus NCS

A mobile processing unit capable of this task is the Movidus Neural Compute Stick, which contains a Visual Processing Unit which can process images and run neural networks on them with a speed equivalent to nearly hundreds of GFlops/s. This will enable faster inferencing, and also reduce the amount of workload on the communication infrastructure such as LAN , which is necessary in the developing world where high speed network connections may not be an ubiquitous utility.

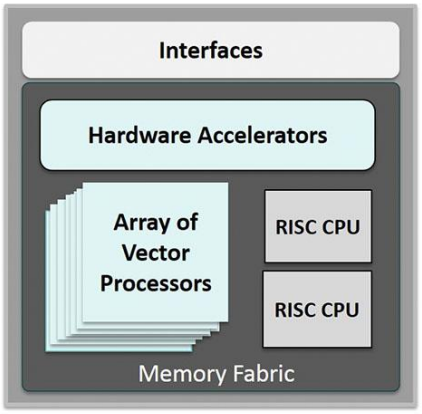

Fig 7. Movidus VPU

The Movidus NCS can run Tensorflow and Caffe frameworks, and requires a raspberry pi or a remote computer to run. First the model needs to be trained on the host computer, then the model weights need to be transformed to the format prescribed by the NCS SDK, and then inferencing can be performed on the network fed into the device.

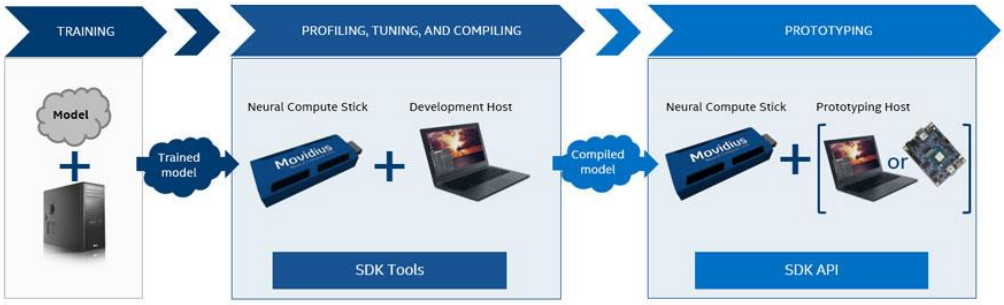

Fig 8. Movidus NCS Workflow

Hence, we have demonstrated how a pipeline is setup for automated attendance system, what are the necessary component and how it can be run remotely to reduce the network overhead.