**Architecture used in the Paper**

Convolutional Neural Networks are similar to ordinary neural networks, but with an explicit assumption that the inputs are images, allowing designers to encode certain properties into the architecture. CNN architecture comprises of a sequence of layers with the simplest architecture being [INPUT−CONV−RELU−POOL−FC]. INPUT layer holds the raw pixel values of the images, CONV layer consists of a kernel or filter of a fixed size which slides in a window fashion to perform the convolution operation on the windowed image to extract features. Padding is applied onto the size of input image to overcome uneven mapping with filter size. RELU stands for rectified linear units, which is an element wise activation function that assigns zero value to hidden units. POOL denotes the pooling layer, which is responsible for down sampling and dimensionality reduction that in turn reduces the computational power required to process data. Pooling layer also has a kernel or function which slides like a window onto the input to extract dominant features that are rotational and positional invariant. Max pooling and Average pooling are the two common functions used. FC is the fully connected layer where each neuron in the input is connected to each neuron in the output and this layer is responsible in computing the score of a particular class, resulting in N outputs where N denotes the number of classes/categories to be classified. The class with maximum score is decided as the predicted class of the CNN architecture. FC layer is also referred to as DENSE layer. It may be noted that the CNN architecture can be modified based on the design requirements and performance of the system. Some of the other layers that are used in CNN architecture include DROPOUT and FLATTEN. DROPOUT layer is a regularization technique to prevent over fitting of CNN, wherein a fraction of inputs (referred to as dropout rate) are dropped out by setting their values to 0 at each update during training. The values of inputs that are retained are scaled up, so that their sum is unchanged during training. FLATTEN layers are introduced before FC layer to convert the two dimensional features into one dimension.

**Dataset Details**

Our dataset talks about Face Recognization It contains two folders (folder for train and folder for test) it consists of 16 different Labels

In training:

15 in folder face1 and 20 in folder face10 16 in folder face11 and 14 in folder face12 13 in folder face13 and 12 in folder face14 15 in folder face15 and 17 in folder face16 15 in folder face2 and 14 in folder face3 17 in folder face4 and 16 in folder face5 16 in folder face6 and 14 in folder face7 14 in folder face8 and 16 in folder face9 For Testing

4 in folder face1 and 4 in folder face10 and 4 in folder face11 and 4 in folder face12 and 4 in folder face13 and 4 in folder face14 and 4 in folder face15 and 4 in folder face16 and 4 in folder face2 and 4 in folder face3 and 4 in folder face4 and 4 in folder face5 and 4 in folder face6 and 4 in folder face7 and 4 in folder face8 and 4 in folder face9

Total Images in Training 244 and 64 Image In Testing

**Implementation Details**

We used CNN Model Conv 2D with Kernal Size 5 X 5 and Activation Function Relu then Max Pooling 2D 2 X 2 then we use with Kernal Size 5 X 5 and Stride 1 X 1 and Activation Function Relu and Max Pooling 2D 2 X 2 then we flattened Matrix and Entered it in ANN with 100 Layer then the output layer 16 Neurnand Activation Function is Softmax

We used adam optimizer and loss is 'sparse\_categorical\_crossentropy' and metrics=['accuracy'])

Total params: 4,924,248

Trainable params: 4,924,248

We Used 10 Epochs and Validation Steps equals 10

Results and visualizations







