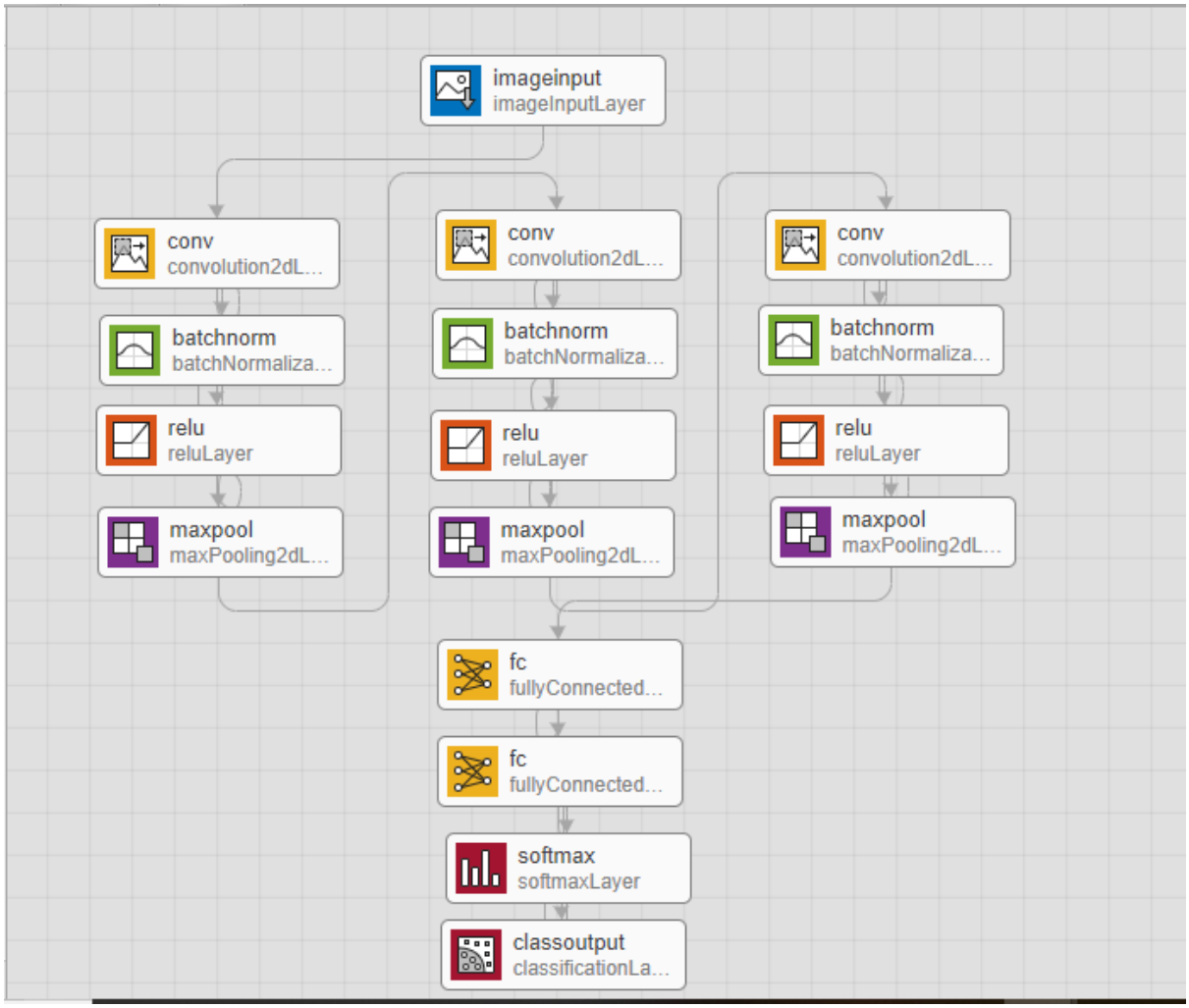
Project summary on:

CNN for detection of hand written characters

Name : Adarsh Dubey

Reg. No. : 16BCE1336

Creating layers for the network

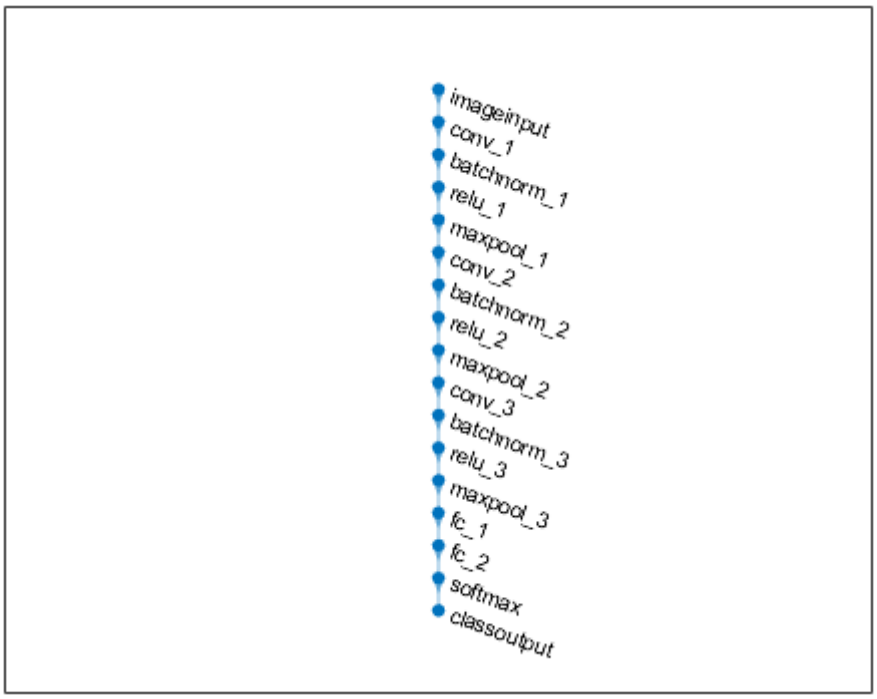



```
layers = [
    imageInputLayer([28 28 1],"Name","imageinput")
    convolution2dLayer([3 3],3,"Name","conv_1","Padding","same")
    batchNormalizationLayer("Name","batchnorm_1")
    reluLayer("Name","relu_1")
    maxPooling2dLayer([5 5],"Name","maxpool_1","Padding","same")
    convolution2dLayer([3 3],3,"Name","conv_2","Padding","same")
    batchNormalizationLayer("Name","batchnorm_2")
```

```
    reluLayer("Name","relu_2")
    maxPooling2dLayer([5 5],"Name","maxpool_2","Padding","same")
    convolution2dLayer([3 3],3,"Name","conv_3","Padding","same")
    batchNormalizationLayer("Name","batchnorm_3")
    reluLayer("Name","relu_3")
    maxPooling2dLayer([5 5],"Name","maxpool_3","Padding","same")
    fullyConnectedLayer(64,"Name","fc_1")
    fullyConnectedLayer(36,"Name","fc_2")
    softmaxLayer("Name","softmax")
    classificationLayer("Name","classoutput")];
```

Plot the Layers

```
plot(layerGraph(layers));
```



Loading the images

Load the images using imageDatastore function and lable the images as their respective folder names. The images are devided into the folders with names as the lable as to what the image is.

```
imds=imageDatastore('C:\Users\ALOK DUBEY\Desktop\New
folder\handwritten_alpha_numeric_chars',"IncludeSubfolders",true,"LabelSource"
,"foldernames");
```

Test train split of the data

The image data is split into test and training data. We will use 80% of the data for training the network and 20% data to test the performance of the network.

```
[train_imgs,test_images]=splitEachLabel(imds,0.9,"randomized");
```

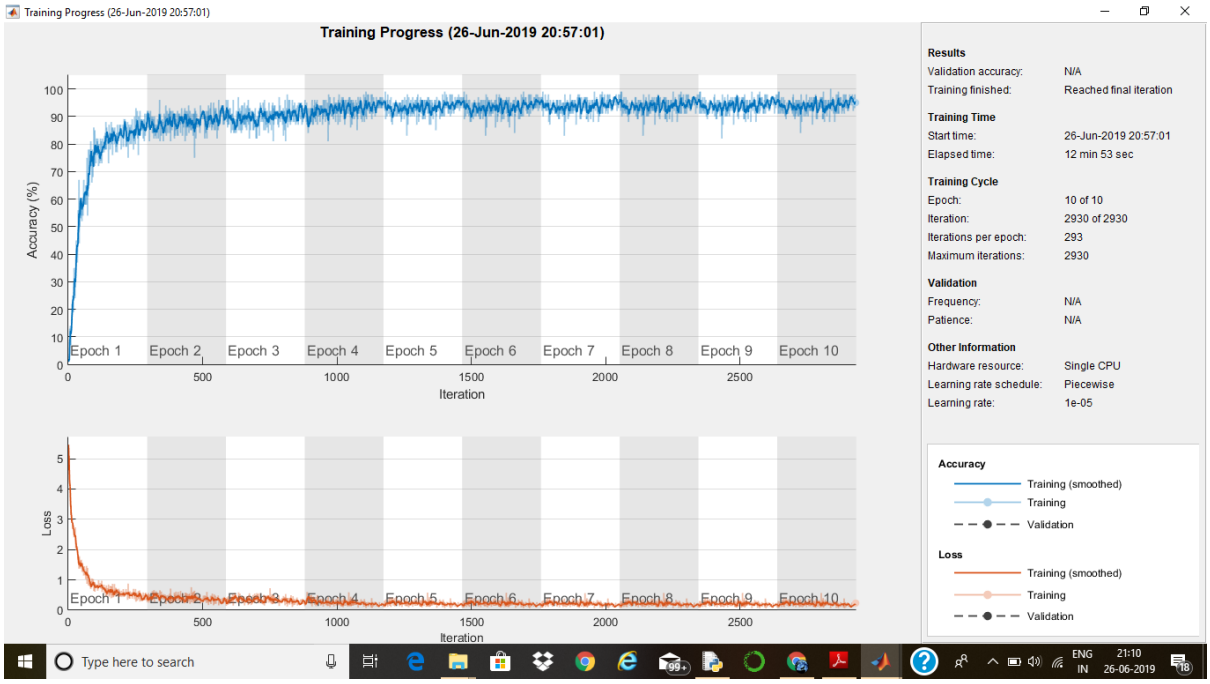
Training Options and Training the network

We set up the parameters to be used by the network for the training . We then pass the training images to the network along with the lables of the images and train the network to recognize them. We then test the network in next section and calculate it's accuracy.

```
%"MiniBatchSize",100,...\
%'InitialLearnRate', 0.001, ...
%'LearnRateSchedule', 'piecewise', ...
%'LearnRateDropFactor', 0.1, ...
%'LearnRateDropPeriod', 3,...
%"Verbose",true
layers=convNet.Layers;
layers(15)=fullyConnectedLayer(36,"Name","fc");
layers(16)=softmaxLayer();
layers(17)=classificationLayer();
opts=trainingOptions('sgdm',...
"MaxEpochs",4,...
"MiniBatchSize",100,...
'InitialLearnRate', 0.01, ...
'LearnRateSchedule', 'piecewise', ...
'LearnRateDropFactor', 0.1, ...
'LearnRateDropPeriod', 3,...
'Plots','training-progress',...
"Verbose",true...
);
%training
```

```
convNet=trainNetwork(train_imgs,layers,opts);
```

Plot:



Training progress:

```
Initializing input data normalization.
|=========================================================================================|
|  Epoch  |  Iteration  |  Time Elapsed  |  Mini-batch  |  Mini-batch  |  Base Learning  |
|         |             |   (hh:mm:ss)   |   Accuracy   |     Loss     |      Rate       |
|=========================================================================================|
|       1 |           1 |       00:00:01 |       1.00% |       5.4600 |          0.0100 |
|       1 |          50 |       00:00:21 |      58.00% |       1.5107 |          0.0100 |
|       1 |         100 |       00:00:35 |      76.00% |       0.8841 |          0.0100 |
|       1 |         150 |       00:00:48 |      73.00% |       0.8490 |          0.0100 |
|       1 |         200 |       00:01:02 |      83.00% |       0.4512 |          0.0100 |
|       1 |         250 |       00:01:13 |      80.00% |       0.7026 |          0.0100 |
|       2 |         300 |       00:01:23 |      86.00% |       0.4782 |          0.0100 |
|       2 |         350 |       00:01:34 |      89.00% |       0.3984 |          0.0100 |
|       2 |         400 |       00:01:44 |      91.00% |       0.3239 |          0.0100 |
|       2 |         450 |       00:01:54 |      86.00% |       0.4876 |          0.0100 |
|       2 |         500 |       00:02:05 |      92.00% |       0.2675 |          0.0100 |
|       2 |         550 |       00:02:15 |      93.00% |       0.2117 |          0.0100 |
|       3 |         600 |       00:02:25 |      89.00% |       0.3350 |          0.0100 |
```

3	650	00:02:36	93.00%	0.2679	0.0100
3	700	00:02:46	88.00%	0.4057	0.0100
3	750	00:02:56	92.00%	0.3358	0.0100
3	800	00:03:07	93.00%	0.3098	0.0100
3	850	00:03:17	88.00%	0.2645	0.0100
4	900	00:03:27	89.00%	0.4998	0.0010
4	950	00:03:37	93.00%	0.2280	0.0010
4	1000	00:03:48	91.00%	0.2125	0.0010
4	1050	00:03:58	89.00%	0.2626	0.0010
4	1100	00:04:09	98.00%	0.1170	0.0010
4	1150	00:04:25	96.00%	0.1632	0.0010
5	1200	00:04:41	96.00%	0.1127	0.0010
5	1250	00:04:56	89.00%	0.3224	0.0010
5	1300	00:05:12	96.00%	0.2017	0.0010
5	1350	00:05:25	94.00%	0.1573	0.0010
5	1400	00:05:39	90.00%	0.2572	0.0010
5	1450	00:05:52	96.00%	0.1666	0.0010
6	1500	00:06:06	91.00%	0.2611	0.0010
6	1550	00:06:19	91.00%	0.2944	0.0010
6	1600	00:06:33	94.00%	0.1462	0.0010
6	1650	00:06:46	92.00%	0.2445	0.0010
6	1700	00:06:59	97.00%	0.1298	0.0010
6	1750	00:07:15	97.00%	0.1636	0.0010
7	1800	00:07:32	92.00%	0.3487	1.0000e-04
7	1850	00:07:52	88.00%	0.2776	1.0000e-04
7	1900	00:08:03	94.00%	0.2752	1.0000e-04
7	1950	00:08:15	93.00%	0.1992	1.0000e-04
7	2000	00:08:27	91.00%	0.1676	1.0000e-04
7	2050	00:08:39	95.00%	0.2198	1.0000e-04
8	2100	00:08:53	93.00%	0.1866	1.0000e-04
8	2150	00:09:07	97.00%	0.1280	1.0000e-04
8	2200	00:09:23	93.00%	0.2038	1.0000e-04
8	2250	00:09:38	96.00%	0.1231	1.0000e-04
8	2300	00:09:51	95.00%	0.2033	1.0000e-04
9	2350	00:10:06	99.00%	0.0788	1.0000e-04
9	2400	00:10:19	94.00%	0.2364	1.0000e-04
9	2450	00:10:33	93.00%	0.1914	1.0000e-04
9	2500	00:10:49	92.00%	0.3376	1.0000e-04
9	2550	00:11:03	96.00%	0.1625	1.0000e-04
9	2600	00:11:19	97.00%	0.1117	1.0000e-04
10	2650	00:11:32	96.00%	0.1345	1.0000e-05
10	2700	00:11:45	95.00%	0.1496	1.0000e-05
10	2750	00:11:59	93.00%	0.1817	1.0000e-05
10	2800	00:12:12	96.00%	0.1538	1.0000e-05
10	2850	00:12:26	96.00%	0.1614	1.0000e-05
10	2900	00:12:42	92.00%	0.2543	1.0000e-05
10	2930	00:12:53	95.00%	0.2369	1.0000e-05

Evaluation of the network

We use the 20% of the images from the data for the calculation of accuray of the trained net.

```
%make predictions

predictedLabels=classify(convNet,test_images);

%get labels of the images in test set

y_test=test_images.Labels;
```

```
%score=number of times the prediction by the system is correct

score= sum(predictedLabels==y_test);

%accuracy is the percentage of times system makes correct prediction

accuracy=score/numel(y_test);

disp(accuracy*100);
```

```
    92.7841
```

```
img=imread('C:\Users\ALOK DUBEY\Desktop\sample.jpeg');
img=rgb2gray(img);
img=medfilt2(img);
img=imbinarize(img);
img=bwlabel(img);
img=~img;
figure,imshow(img);
iProp=regionprops(img,'BoundingBox','Area');
for i =1:length(iProp)
    if iProp(i).Area>100
        im=imcrop(img,iProp(i).BoundingBox);
        im=imresize(im,[28 28]);
        imwrite(im,'C:\Users\ALOK DUBEY\Desktop\New folder\temp\char.jpg');
        j=imread('C:\Users\ALOK DUBEY\Desktop\New folder\temp\char.jpg');
        ch=convNet.classify(j);
        disp(ch);
        clear ch;
        figure,imshow(j);
    end
end
```

Output:

a

d

2