Chiranjeev
P21CS007
chiranjeev.1@iitj.ac.in

**Programming Assignment - 1**

10-02-2022

## Question1:

**Dataset Used:** Letter Recognition

**Constructed a Neural Network with various layers:**

a. Input layer
b. 3 Hidden layers
c. Output layer

**Dataset Split:**

**Split the dataset into:**

1. **Train set:** 60%
2. **Validation set:** 20%
3. **Test set:** 20%

**Xavier Weight Initialization:**

1. **ReLU:**

$$\sqrt{\frac{2}{size^{[l-1]} + size^{[l]}}}$$

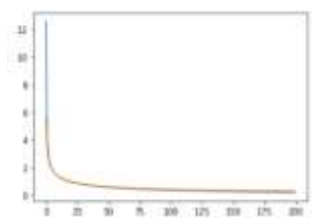
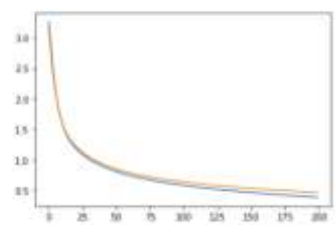
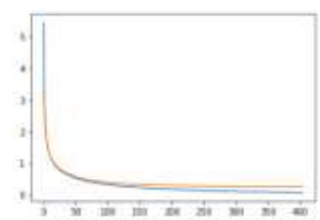
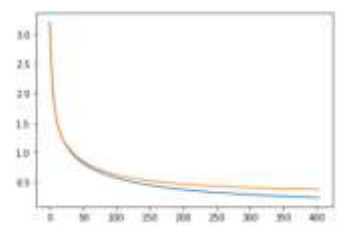
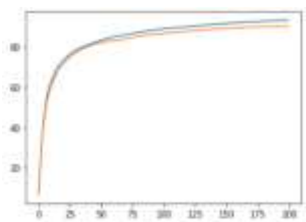
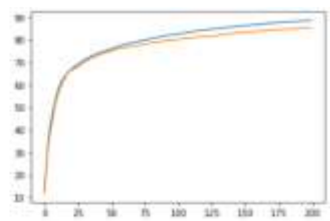
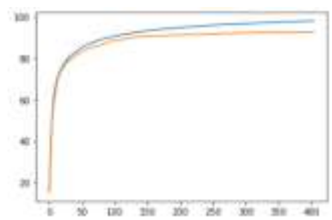
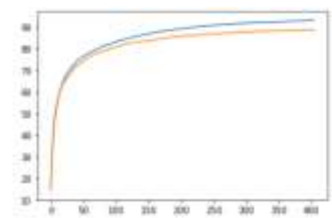
$$W^{[l]} = np.random.randn(size\_l,\ size\_l\text{-}1) * np.sqrt(2\ /\ (size\_l\text{-}1 + size\_l)\ )$$

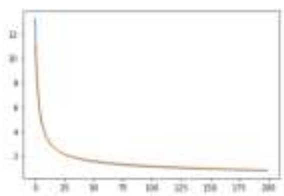
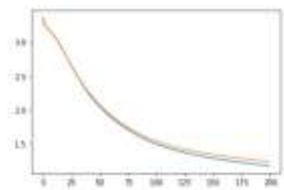
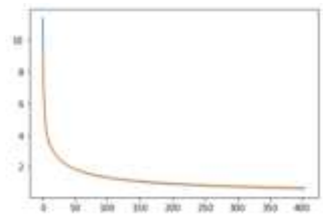
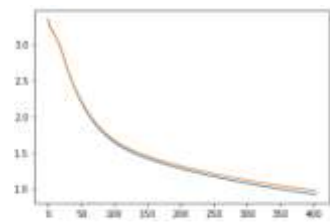
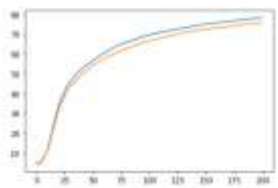
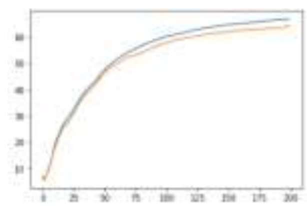
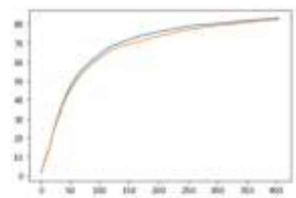
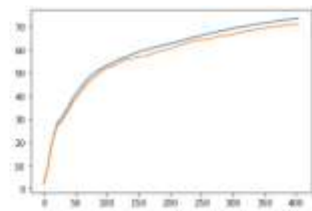
2. **Tanh:**

$$\sqrt{\frac{1}{size^{[l-1]}}}$$

$$W^{[l]} = np.random.randn(size\_l,\ size\_l\text{-}1) * np.sqrt(1/size\_l\text{-}1)$$

**Observations:**

| Learning rate | 0.0001 | | | |
|---|---|---|---|---|
| Epochs | 200 | | 405 | |
| Activation Function | ReLU | Tanh | ReLU | Tanh |
| Test Accuracy | 90.92% | 85.37% | 93.42% | 88.27% |
| Loss Vs Epoch |  |  |  |  |
| Accuracy Vs Epoch |  |  |  |  |

| Learning rate | 0.00001 | | | |
|---|---|---|---|---|
| **Epochs** | **200** | | **405** | |
| **Activation Function** | **ReLU** | **Tanh** | **ReLU** | **Tanh** |
| **Test Accuracy** | **77.82%** | **66.55%** | **81.89%** | **71.30%** |
| **Loss Vs Epoch** |  |  |  |  |
| **Accuracy Vs Epoch** |  |  |  |  |

## Conclusion:

- The best model according to the experiments has various hyper-parameters:
  1. **Learning rate**- 0.0001
  2. **Weights initialization**- Xavier weights initialization method for ReLU
  3. **Epochs**- 405
  4. **Activation function**- ReLU
  5. **Test accuracy**- 93.42%
  6. **Optimizer**- Adam

This is the best model because on comparing performance on Test set it gives better performance as compared to learning rate 0.00001 and Tanh activation function. Also ReLU performs better because it cuts negative values to 0. Whereas Tanh may lead to saturation of gradients if more iterated as it saturates towards +1 and -1.

<u>**Question2:**</u>

**Dataset Used:** MNIST

**Dataset Split:**

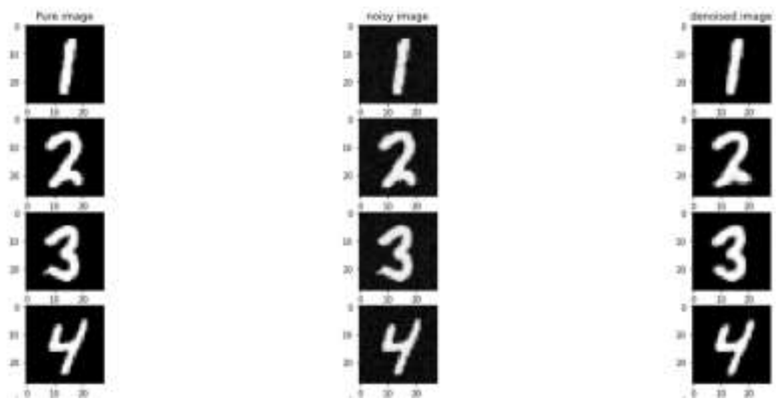
**Split the dataset into:**

1. **Train set:** 48,000 samples
2. **Validation set:** 12,000 samples
3. **Test set:** 10,000 samples

## I. Denoising Autoencoder

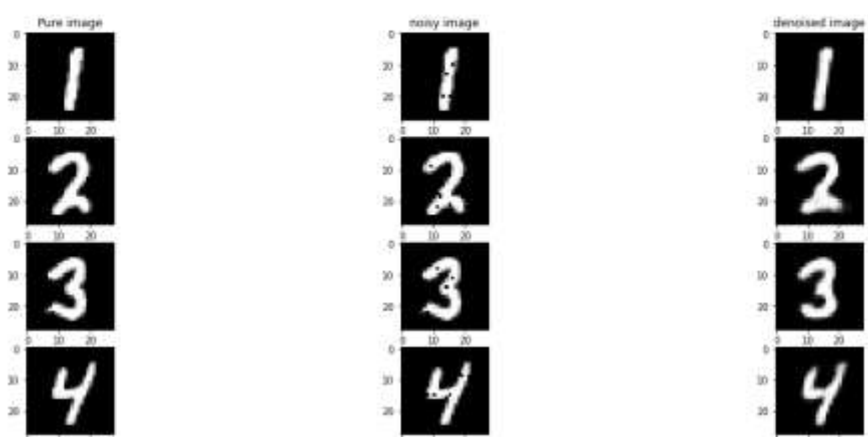
**Experiments:**

1. **Tried with introducing Gaussian noise to the MNIST Dataset:**

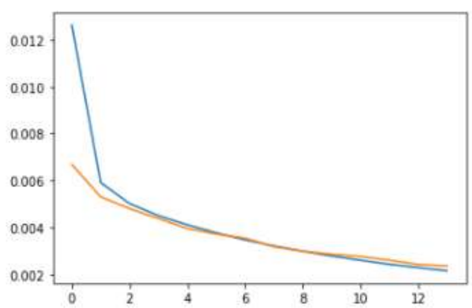
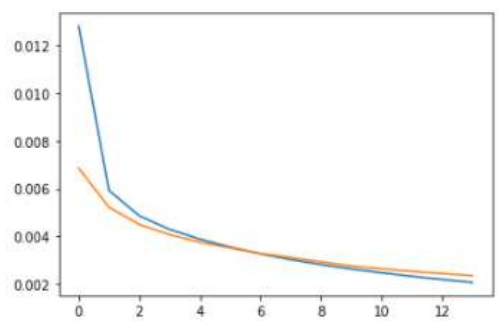
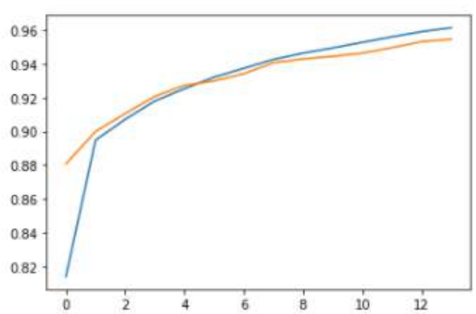
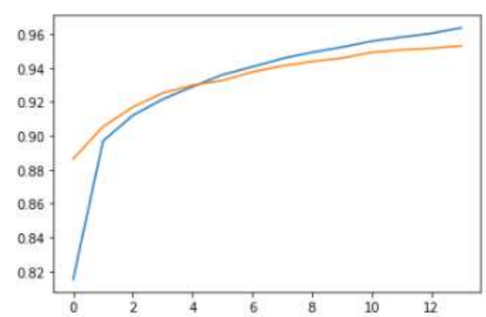


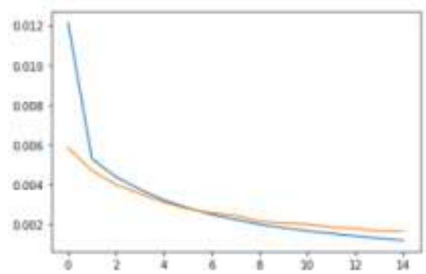
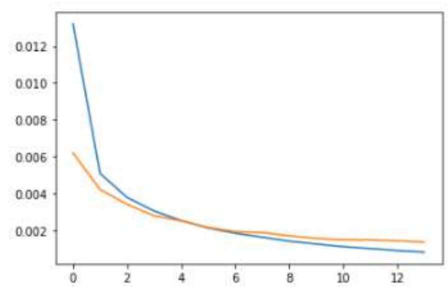
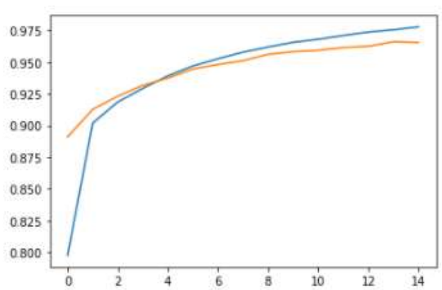
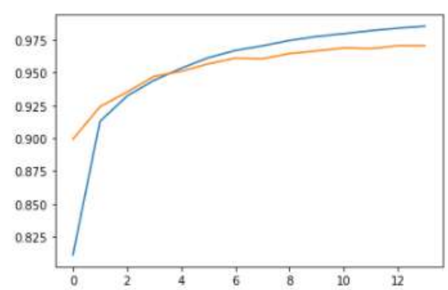

2. **Tried with introducing Salt-pepper noise to the MNIST Dataset:**

   **Output:**



## II. Classification Network

| Model | 1 FC Layer | |
|---|---|---|
| **Network** | • Input Layer(784)<br>• Relu(FC Layer (128))<br>• Output Layer(10) | • Input Layer(784)<br>• tanh(FC Layer (128))<br>• Output Layer(10) |
| **Activation Function** | **ReLU** | **Tanh** |
| **Test Accuracy** | **95.39%** | **95.50%** |
| **Loss Vs Epochs** |   |   |
| **Accuracy Vs Epochs** |   |   |

| Model | 3 FC Layers | |
|---|---|---|
| **Network** | • Input Layer(784)<br>• tanh(FC Layer (256))<br>• tanh(FC Layer (128))<br>• tanh(FC Layer (64))<br>• Output Layer(10) | • Input Layer(784)<br>• tanh(FC Layer (256))<br>• tanh(FC Layer (128))<br>• tanh(FC Layer (64))<br>• Output Layer(10) |
| **Activation Function** | **ReLU** | **Tanh** |
| **Test Accuracy** | 96.58% | 96.94% |
| **Loss<br>Vs<br>Epochs** |  |  |
| **Accuracy<br>Vs<br>Epochs** |  |  |

## Conclusion:

- From the above tables we can conclude that the 3 FC Layer network performs better than 1 FC Layer network on the basis of test accuracy.
- Test accuracy on for 3 FC layer network is 95.39% for Relu activation function and 95.50% for Tanh activation function whereas for 3 FC Layer network is 96.58% for Relu activation function and 96.94% for Tanh activation function. There is improvement in accuracy for 3Layer network

because more FC layer helps in learning more complex features then features learnt by 1 FC layer network.

- Also we can observe that here Tanh activation function outperforms Relu activation function.

## Question3:

**Dataset Used:** Cifar10

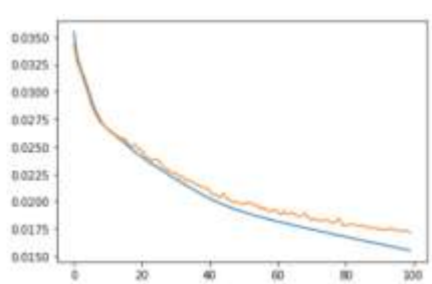
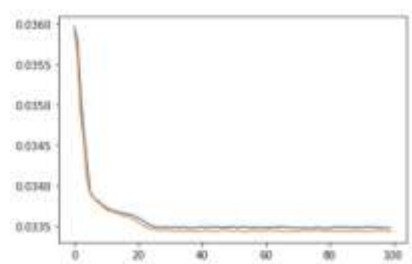
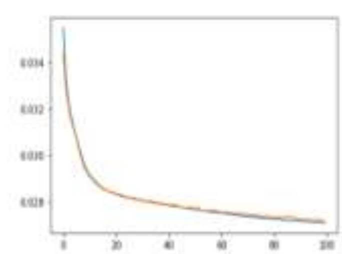
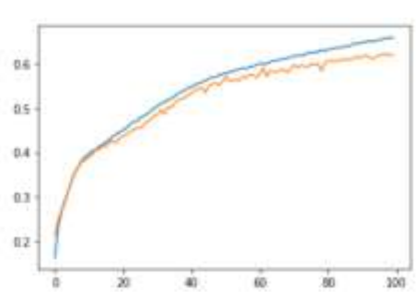
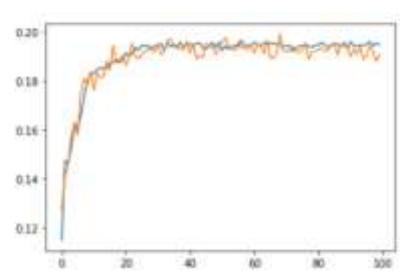
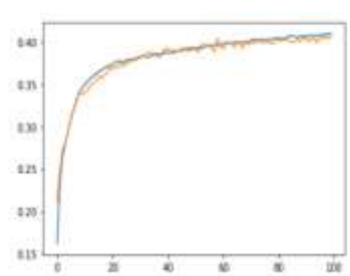
**Dataset Split:**

**Split the dataset into:**

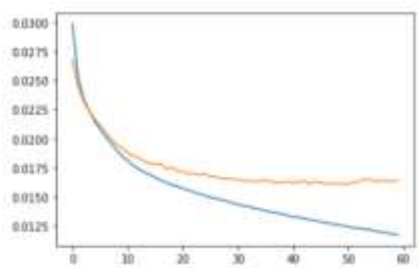
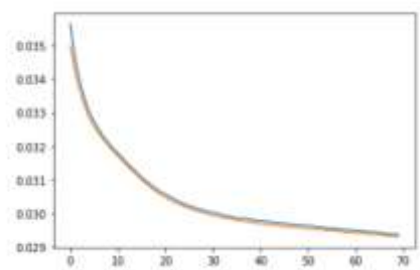
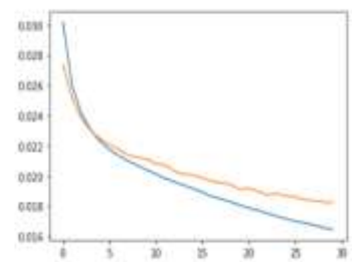
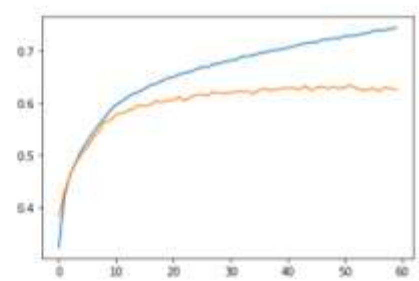
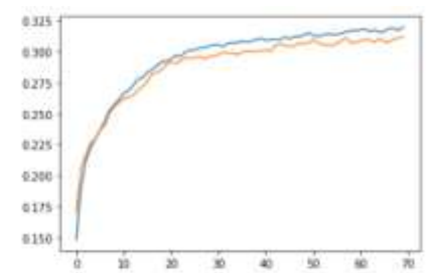
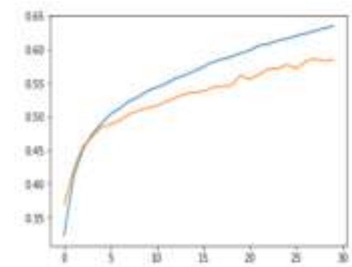
1. **Train set:** 45,000 samples
2. **Validation set:** 5,000 samples
3. **Test set:** 10,000 samples

**Architectures:**

**I. Conv-Conv-Pool-Conv-Conv-Pool-FC-FC**

| **Hyper-parameters** | • **Maxpool - (2x2)**<br>• **Activation - Softmax**<br>• **Kernel size for Conv - (5x5)**<br>• **Stride - 1** | • **Maxpool - (1x1)**<br>• **Activation - Softmax**<br>• **Kernel size for Conv- (1x1)**<br>• **Stride - 2** | • **Avgpool - (2x2)**<br>• **Activation - Softmax**<br>• **Kernel size for Conv- (5x5)**<br>• **Stride - 1** |
|---|---|---|---|
| **Test Accuracy** | **62.53%** | 18.94% | 39.46% |
| **Loss Vs Epoch** |  |  |  |
| **Accuracy Vs Epoch** |  |  |  |

**II. Conv-Pool-BatchNormalization-ReLU-Conv-Pool-BatchNormalization-ReLU-FC**

| | | | |
|---|---|---|---|
| **Hyper-parameters** | • **Maxpool - (2x2)**<br>• **Activation – Softmax Relu**<br>• **Kernel size for Conv - (5x5)**<br>• **Stride - 1** | • **Maxpool - (1x1)**<br>• **Activation – Softmax Relu**<br>• **Kernel size for Conv- (1x1)**<br>• **Stride - 2** | • **Avgpool - (2x2)**<br>• **Activation - Softmax Relu**<br>• **Kernel size for Conv- (5x5)**<br>• **Stride - 1** |
| **Test Accuracy** | **63.59%** | 32.13% | 39.46% |
| **Loss Vs Epoch** |  |  |  |
| **Accuracy Vs Epoch** |  |  |  |

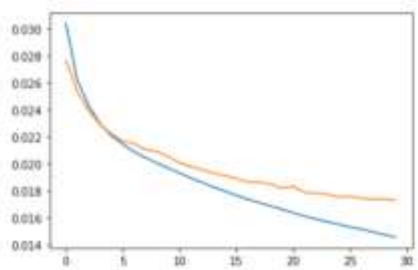
**III. Conv-BatchNormalization-ReLU-Conv-BatchNormalization-ReLU-FC**

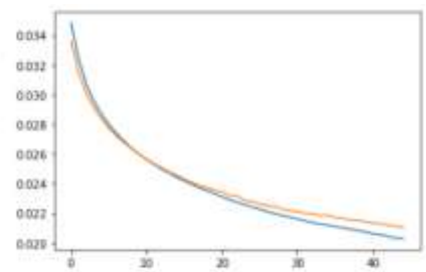
<table>
<tr>
<td>Hyper-parameters</td>
<td>

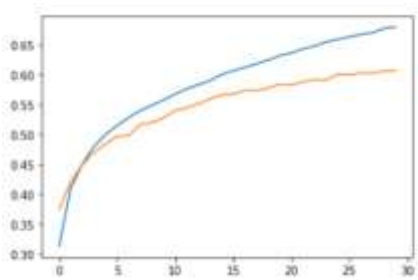
- **Maxpool - not used**
- **Activation - Softmax**<br>**Relu**
- **Kernel size for Conv - (5x5)**
- **Stride - 1**

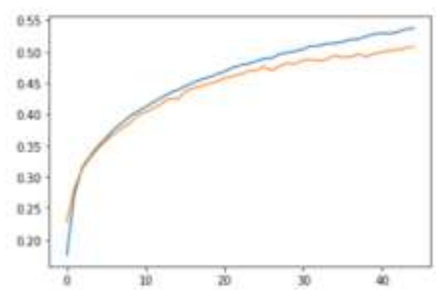
</td>
<td>

- **Maxpool - not used**
- **Activation - Softmax**<br>**Relu**
- **Kernel size for Conv- (5x5)**
- **Stride - 2**

</td>
</tr>
<tr>
<td>Test Accuracy</td>
<td>61.43%</td>
<td>52.47%</td>
</tr>
<tr>
<td>Loss<br>Vs<br>Epoch</td>
<td>


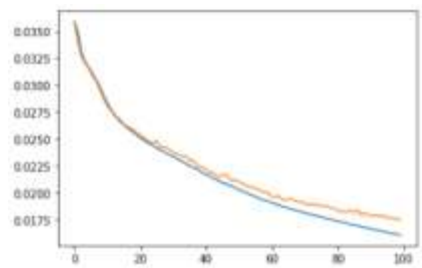
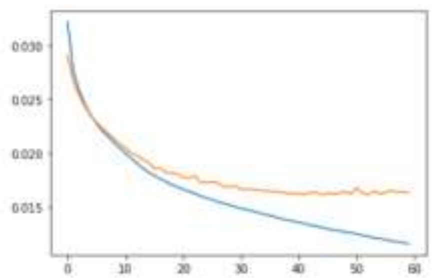
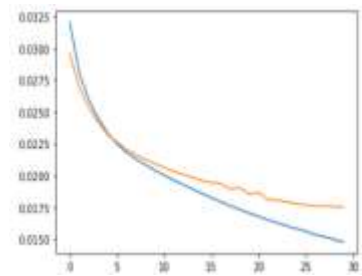
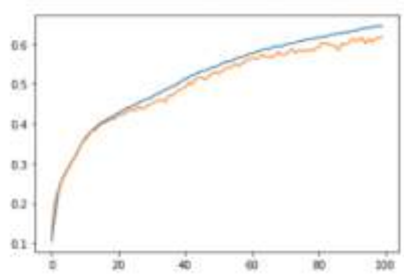
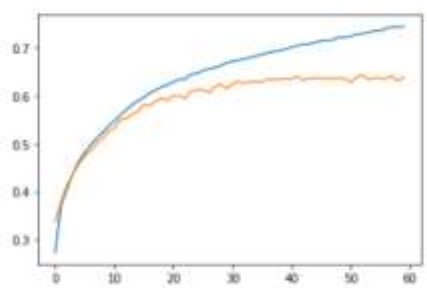
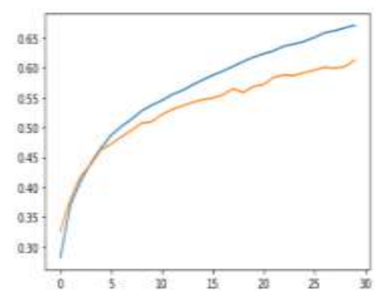
</td>
<td>


</td>
</tr>
<tr>
<td>Accuracy<br>Vs<br>Epoch</td>
<td>


</td>
<td>


</td>
</tr>
</table>

**IV. Increasing 1FC layer in each best architecture got in above tables**

| **Architecture** | **Conv-Conv-Pool-Conv-Conv-Pool-FC-FC-FC** | **Conv-Pool-BatchNormalization-ReLU-Conv-Pool-BatchNormalization-ReLU-FC-FC** | **Conv-BatchNormalization-ReLU-Conv-BatchNormalization-ReLU-FC-FC** |
|---|---|---|---|
| **Hyper-parameters** | • **Maxpool - (2x2)**<br>• **Activation – Softmax**<br>• **Kernel size for Conv - (5x5)**<br>• **Stride - 1** | • **Maxpool - (2x2)**<br>• **Activation – Softmax Relu**<br>• **Kernel size for Conv- (5x5)**<br>• **Stride - 1** | • **MAxpool - (2x2)**<br>• **Activation - Softmax Relu**<br>• **Kernel size for Conv- (5x5)**<br>• **Stride - 1** |
| **Test Accuracy** | 61.31% | **63.77%** | 61.61% |
| **Loss Vs Epoch** |  |  |  |
| **Accuracy Vs Epoch** |  |  |  |

## Conclusion:

- On changing Network size the network performs better according to the observations as it can learn more complex features. Also on introducing batch normalization layer helps in normalizing the batches for layers so that they can come in similar range.

- We can observe from above tables that MaxPool performs better than Average pool as maxpool helps in getting the best features from its window size.

- On changing the stride from 1 to 2 it reduces the convolution output much as compared to stride = 1. We can see from formula of the shape of output we get after convolution operation there is division by Stride(S). Thus on dividing by stride 2 it reduces the output size much as compared to dividing by stride = 1.

- On increasing Layer we can observe that the accuracy is increase as the network can now learn more complex features.