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:

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

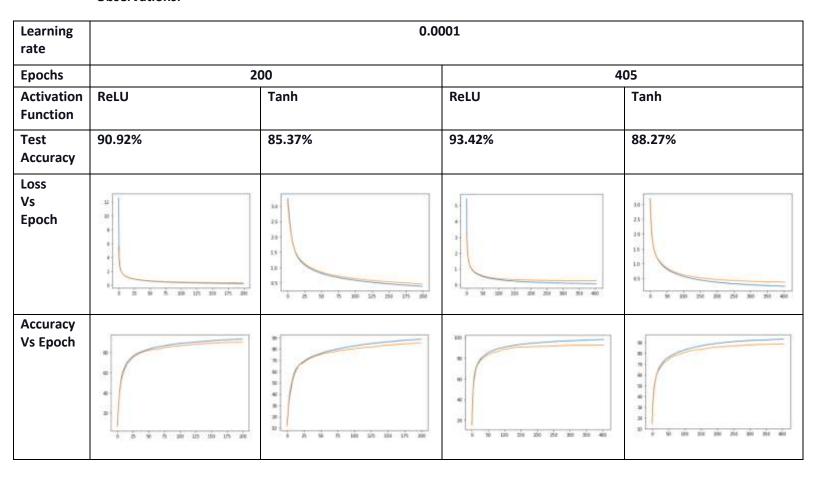
$$W^{[l]} = np.random.randn(size_l, size_l-1) * np.sqrt(2 / (size_l-1 + size_l))$$

2. **Tanh:**

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

$$W^{[l]} = np.random.randn(size_l, size_l-l) * np.sqrt(l/size_l-l)$$

Observations:



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	12 10 10 10 10 10 10 10 10 10 10 10 10 10	29 28 20 15	0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	10 21 29 11 11 18 18 50 300 150 300 251 300 201 400
Accuracy Vs Epoch	30 00 00 00 100 100 100 100 100 100 100	10 10 10 10 10 10 10 10 10 10 10 10 10 1	# 10 10 10 200 200 60 No 601	20 Me min

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.

Question2:

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:

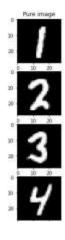


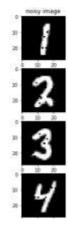


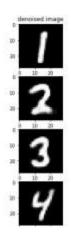


2. Tried with introducing Salt-pepper noise to the MNIST Dataset:

Output:







II. Classification Network

Model	1 FC Layer			
Network	Input Layer(784)Relu(FC Layer (128))Output Layer(10)	Input Layer(784)tanh(FC Layer (128))Output Layer(10)		
Activation Function	ReLU	Tanh		
Test Accuracy	95.39%	95.50%		
Loss Vs Epochs	0.012 0.010 0.008 0.004 0.002 0 2 4 6 8 10 12	0.012 - 0.008 - 0.006 - 0.004 - 0.002 - 0 2 4 6 8 10 12		
Accuracy Vs Epochs	0.96 0.94 0.92 0.90 0.88 0.86 0.84 0.82	0.96 - 0.94 - 0.92 - 0.90 - 0.88 - 0.86 - 0.84 - 0.82 - 0 2 4 6 8 10 12		

Model	3 FC Layers		
Network	 Input Layer(784) tanh(FC Layer (256)) tanh(FC Layer (128)) tanh(FC Layer (64)) Output Layer(10) 	 Input Layer(784) tanh(FC Layer (256)) tanh(FC Layer (128)) tanh(FC Layer (64)) Output Layer(10) 	
Activation Function	ReLU	Tanh	
Test Accuracy	96.58%	96.94%	
Loss Vs Epochs	0.013 0.018 0.008 0.004 0.004	0.012 0.010 0.008 0.006 0.004 0.002	
Accuracy Vs Epochs	0.975 0.950 0.925 0.900 0.875 0.850 0.825 0.800	0.975 - 0.950 - 0.925 - 0.990 - 0.875 - 0.850 - 0.825 - 0.925	

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:

Train set: 45,000 samples
 Validation set: 5,000 samples
 Test set: 10,000 samples

Architectures:

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

Hyper- parameters Test Accuracy	 Maxpool - (2x2) Activation - Softmax Kernel size for Conv - (5x5) Stride - 1 	 Maxpool - (1x1) Activation - Softmax Kernel size for Conv- (1x1) Stride - 2 	 Avgpool - (2x2) Activation - Softmax Kernel size for Conv- (5x5) Stride - 1 39.46%
Loss Vs Epoch	0.0350 0.0325 0.0390 0.0275 0.0250 0.0250 0.0179 0.0150 0 20 40 60 80 306	0.0255 0.0258 0.0245 0.0245 0.0235	8:002 8:002 8:008 8:20 40 60 80 208
Accuracy Vs Epoch	0.6 - 0.3 - 0.3 - 0.0 -	0.10 0.16 0.14 0.12	6.40 6.33 6.23 6.23 6.35 6.25 6.25 6.25 6.25 6.25 6.25 6.25 6.2

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

Hyper- parameters	 Maxpool - (2x2) Activation - Softmax Relu Kernel size for Conv - (5x5) Stride - 1 	 Maxpool - (1x1) Activation - Softmax Relu Kernel size for Conv- (1x1) Stride - 2 	 Avgpool - (2x2) Activation - Softmax Relu Kernel size for Conv- (5x5) Stride - 1
Test Accuracy	63.59%	32.13%	39.46%
Loss Vs Epoch	0.0350 0.0350 0.0255 0.0350 0.0175 0.0150 0.0125	0.015 0.013 0.013 0.011 0.011 0.029 0 10 20 30 40 50 40 70	0.038 0.038 0.034 0.022 0.029 0.029 0.029 0.036 0.036 0.036 0.036
Accuracy Vs Epoch	0.5 0.5 5.4 0 10 20 30 40 50 60	0 325 0 300 0 275 0 250 0 275 0 250 0 150 0 150 0 150 0 30 30 40 50 80 70	845- 855- 846- 846- 846- 846- 846- 846- 846- 846

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

Hyper-parameters Test Accuracy	 Maxpool - not used Activation - Softmax Relu Kernel size for Conv - (5x5) Stride - 1 61.43% 	 Maxpool - not used Activation - Softmax Relu Kernel size for Conv- (5x5) Stride - 2 52.47%
Loss Vs Epoch	0030 0029 0024 0022 0030 0018 0016	0.034 0.032 0.036 0.026 0.024 0.022 0.022
Accuracy Vs Epoch	0.85 0.90 0.35 0.45 0.40 0.35 0.30 0 5 30 25 30 25 30	0.55 0.50 0.45 0.40 0.35 0.30 0.25 0.30 0.30 0.30 0.30 0.30 0.40

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) Activation - Softmax Kernel size for Conv - (5x5) Stride - 1 	 Maxpool - (2x2) Activation - Softmax	 MAxpool - (2x2) Activation - Softmax Relu Kernel size for Conv- (5x5) Stride - 1 	
Test Accuracy	61.31%	63.77%	61.61%	
Loss Vs Epoch	0.0325 0.0325 0.0325 0.0250 0.0250 0.0250 0.0175	0.030 - 0.025	0.0025 0.0000 0.0025 0.0025 0.0025 0.0025	
Accuracy Vs Epoch	06 05 04 03 02 01 20 40 60 80 200	0.7 - 0.6 - 0.5 - 0.4 - 0.3 - 0.10 20 30 40 50 60	0.65 - 0.60 - 0.55 - 0.50 - 0.45 - 0.40 - 0.35 - 0.30 - 0 5 20 15 20 25 20	

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.