

If not mentioned , Learning rate = 0.1, epochs = 3 , Batch size = 1 for SGD or full dataset for batch GD, Step activation function for perceptron

Q1 Task 1

Implement the OR Boolean logic gate using perceptron Neural Network. Inputs = x_1, x_2 and bias, weights should be fed into the perceptron with single Output = y . Display final weights and bias of each perceptron.

Task 2

- Use the iris dataset Encode the input and show the new representation
 - Decode the lossy representation for the output
 - Map the input to reconstruction and visualize
-

Task 1

Implement the OR Boolean logic gate using perceptron Neural Network. Inputs = x_1, x_2 and bias, weights should be fed into the perceptron with single Output = y . Display final weights and bias of each perceptron.

Q2 Task 2

- Use the heart disease Dataset
 - Create an Auto Encoder and fit it with our data using 3 neurons in the dense layer
 - Display new reduced dimension values
 - Plot loss for different Auto encoders
-

Q3 Task 1

Implement the OR Boolean logic gate using perceptron Neural Network. Inputs = x_1, x_2 and bias, weights should be fed into the perceptron with single Output = y . Display final weights and bias of each perceptron.

Task 2

- Load the Intel Image dataset
 - Train and test the dataset
 - Create a model using CNN
 - Evaluate the model using confusion matrix.
-

Q4 Task 1

Implement the OR Boolean logic gate using perceptron Neural Network. Inputs = x_1, x_2 and bias, weights should be fed into the perceptron with single Output = y . Display final weights and bias of each perceptron.

Task 2

- Implement autoencoder
 - Use the Iris Dataset
 - Create an autoencoder and fit it with our data using 2 neurons in the dense layer
 - Plot loss w.r.t. epoch
 - Calculate reconstruction error using Mean Squared Error (MSE).
-

Q5 Task 1

Implement the NOT Boolean logic gate using perceptron Neural Network. Inputs = x_1, x_2 and bias, weights should be fed into the perceptron with single Output = y . Display final weights and bias of each perceptron.

Task 2

1. Use the Iris Dataset
 2. Create an Auto Encoder and fit it with our data using 3 neurons in the dense layer
 3. Display new reduced dimension values
 4. Plot loss for different encoders
-

Task 1

Implement the OR Boolean logic gate using perceptron Neural Network. Inputs = x_1, x_2 and bias, weights should be fed into the perceptron with single Output = y . Display final weights and bias of each perceptron.

Task 2

Use the heart disease dataset and do the following

- Use the Dataset
 - Create an autoencoder and fit it with our data using 2 neurons in the dense layer
 - Plot loss w.r.t. epochs
 - Calculate reconstruction error using Mean Squared Error (MSE).
-

Q6

- Load California Housing dataset and select 2 features (e.g., Median Income, House Age) and 1 target (Median House Value).
 - Normalize inputs and initialize a single-layer NN with random weights and bias.
 - Perform forward propagation, calculate prediction error, Squared Error, and MSE.
 - Update weights and bias using gradient descent.
 - Plot Loss vs Weight, Loss vs Bias, and Error Surface.
-

Take the dataset of Life expectancy

Initialize a neural network with random weights.

- a) Calculate output of Neural Network
 - b) Calculate squared error loss
 - c) Plot the mean squared error for each iteration in stochastic Gradient Descent.
-

Task 1

Implement the OR Boolean logic gate using perceptron Neural Network. Inputs = x_1, x_2 and bias, weights should be fed into the perceptron with single Output = y . Display final weights and bias of each perceptron.

Task 2

Implement autoencoder

- Use the Wine Dataset
 - Create an autoencoder and fit it with our data using 3 neurons in the dense layer
 - Calculate loss w.r.t to different epochs and plot using line graph.
-

Q7 Implement Self Organizing Map for anomaly Detection

- Use Credit Card Applications Dataset:

- Detect fraud customers in the dataset using SOM and perform hyperparameter tuning
 - Show map and use markers to distinguish frauds.
-

Q8 Train a small neural network (dataset - MNIST classification)

Compare the optimizers:

1. SGD
2. SGD + Momentum
3. Adam

Plot:

1. Training loss vs epochs
 2. Accuracy vs epochs
-

Q9 1. Use MNIST or IRIS/ Cifar-10 Dataset

2. Train a model with and without data augmentation (horizontal flip, rotation, noise).
 3. Compare generalization performance on the validation set. (Accuracy & Error)
 4. Evaluate the model using confusion matrix, precision, recall
-

Q10 1. Implement a tiny SimCLR framework using a small dataset (e.g., CIFAR-10 subset).

2. Use data augmentations.
 3. Implement the NT-Xent loss function to compute similarity between pairs.
 4. Compare Data with and without Augmentation.
-

Q11 1. Use a pretrained model (e.g., ResNet-50 or MobileNet).

2. Freeze its encoder.
 3. Train a classifier head on a different dataset (e.g., Flowers dataset).
 4. Compare accuracy with fine tuning
-

Choose two datasets with different distributions (dogs & cats , cars).

1. Resize images to the required input size of the chosen pre-trained model.
 2. Load Pre-trained Model (LeNet-5 or VGG-16)
 3. Compare the performances of all the models and visualize
 4. Write down your observations and conclusions
-

Choose two datasets with different distributions (dogs & cats , cars).

1. Resize images to the required input size of the chosen pre-trained model.
 2. Load Pre-trained Model (AlexNet or ResNet-50)
 3. Compare the performances of all the models and visualize
 4. Write down your observations and conclusions
-

1. Choose two datasets with different distributions (dogs & cats , cars).

2. Resize images to the required input size of the chosen pre-trained model.
 3. Load Pre-trained Model (ResNet-50)
 4. Compare the performances of all the models and visualize
 5. Write down your observations and conclusions
-

- Take the dataset of Breast cancer
- Initialize a neural network with random weights.
- Calculate output of Neural Network:
- Calculate MSE
- Plot error surface using loss function versus weight, bias

Q12

- Perform this cycle in step c for every input output pair
- Perform 5 epochs of step d.
- Update weights accordingly using stochastic gradient descend.
- Plot the mean squared error for each iteration in stochastic Gradient Descent.
- Similarly plot accuracy for iteration and note the results

Q13

- Take the dataset of Iris.
- Initialize a neural network with random weights.
- Calculate output of Neural Network:
- Calculate MSE
- Plot error surface using loss function verses weight, bias
- Perform this cycle in step c for every input output pair
- Perform 10 epochs of step d.
- Update weights accordingly using stochastic gradient descend.
- Plot the mean squared error for each iteration in stochastic Gradient Descent.
- Similarly plot accuracy for iteration and note the results

Q14

- Implement batch gradient descent optimizer function
Take the dataset of Titanic
- Initialize a neural network with random weights.
- Calculate output of Neural Network:
- Calculate squared error loss
- Update network parameter using batch gradient descent optimizer function Implementation.
- Display updated weight and bias values
- Plot loss w.r.t. Iterations

Q15

1. Implement the NOR Boolean logic gate using perceptron Neural Network. Inputs = x_1 , x_2 and bias, weights should be fed into the perceptron with single Output = y . Display final weights and bias of each perceptron.
2. Take the dataset of Diabetes 2
 - a) Initialize a neural network with random weights.
 - b) Calculate output of Neural Network:
 - i. Calculate squared error loss
 - ii. Update network parameter using batch Mini Batch gradient descent optimizer function Implementation.
 - iii. Display updated weight and bias values
 - iv. Plot loss w.r.t. bias values

- Take the dataset of Diabetes
- Initialize a neural network with random weights.
 - c)Calculate output of Neural Network:
 - Calculate Mean squared error loss
 - Update network parameter using batch momentum based gradient descent optimizer function Implementation.
 - Display updated weight and bias values
 - Plot loss w.r.t. iterations

1. Implement the XOR Boolean logic gate using perceptron Neural Network. Inputs = x_1 , x_2 and bias, weights should be fed into the perceptron with single Output = y . Display final weights and bias of each perceptron.

Q16

2. Take the dataset of Penguin
3. b) Initialize a neural network with random weights.
c) Calculate output of Neural Network:
 - i. Calculate squared error loss
 - ii. Update network parameter using batch Adaptive delta gradient descent optimizer function Implementation.
 - iii. Display updated weight and bias values
 - iv. Plot accuracy w.r.t. epoch values

Q17

1. Implement backpropagation algorithm from scratch.
- a) Take Iris Dataset
- b) Initialize a neural network with random weights.
- c) Calculate Squared Error (SE)
- d) Perform multiple iterations.
- e) Update weights accordingly.
- f) Plot accuracy for iterations and note the results.

Build a multiclass image categorization CNN network which correctly classifies different categories of images in the dataset.(handwritten digits from Mnist digit dataset)

- Split original dataset to train and test set
- Build CNN Model
- Generate the accuracy of the built model using Adam Optimizer and Adagrad Optimizer.
- Compare performance of different optimizer on Digit categorization.
- Plot training vs validation accuracy
- Evaluate the model using confusion matrix, precision, recall.

Build a multiclass image categorization CNN network which correctly classifies different categories of images in the dataset.(Fashion Mnist dataset.)

- Split original dataset to train and test set
- Build CNN Model
- Generate the accuracy of the built model using RMSProp and SGDOptimizer.
- Perform hyperparameter tuning to increase the accuracy of the CNN.
- Compare performance of different optimizer on Cloth categorization.
- Plot training vs validation loss
- Evaluate the model using confusion matrix, precision.

1. Implement CNN and compare its performance using different optimizers

Take the MNIST dataset

- a) Initialize a neural network basic layers with random weights.
- c) Perform practical analysis of optimizers on MNIST dataset keeping batch size, and epochs same but with different optimizers.
- d) Compare the results by choosing 5 different optimizers [SGD, Adadelta, Adagrad, Adam, RMSprop] on a simple neural network

- Load the CIFAR-10 image dataset.
- Split the dataset into training and testing sets.

Q19

- Design a CNN model for object classification.
- Train and evaluate the model.
- Plot accuracy and loss curves.

- Use alpaca dataset
- CNN must include : Convolution layer, Pooling layer, Flatten layer,Dense layer
Plot:
 - Accuracy vs Epochs
 - Loss (Error) vs Epochs

Q20

1. Load the Corn 3-Classes image dataset.
2. Preprocess the images:
 - a. Resize images to a fixed size (e.g., 224×224)
 - b. Normalize pixel values.
3. Split the dataset into training and testing sets.
4. Create a CNN model using:
 - a. Convolution layer
 - b. Max Pooling layer
 - c. Flatten layer
 - d. Dense layer
5. Train the CNN model for multi-class classification.
6. Test the model on unseen images.
7. Plot graphs:
 - a. Training vs Validation Accuracy
 - b. Training vs Validation Loss (Error)

Q7 Implement Self Organizing Map for anomaly Detection

1. Use Credit Card Applications Dataset:
2. Detect fraud customers in the dataset using SOM and perform hyperparameter tuning
3. Show map and use markers to distinguish frauds.

Task 1

Implement the NAND Boolean Logic Gate using a Perceptron Neural Network.

- Inputs: x1, x2, bias
- Train using perceptron learning rule
- Output: y
- Display final weights and bias
- Verify truth table results

Task 2

Use the Iris Dataset

- Normalize the input features
- Perform Min–Max scaling
- Visualize original vs normalized features

Task 1

Implement Multi-output Perceptron for

- AND gate
- OR gate
- Display weight matrix and bias vector

Task 2

Load Flowers Dataset

- Train CNN model with 3 kernels.
 - Plot training and validation accuracy using graph.
-

Implement perceptron

- Train on AND,OR gate
- Compare convergence with normal perceptron

Task 2

Use Iris Dataset

- Encode using Autoencoder (3 neurons)
 - Decode and reconstruct
 - Plot original vs reconstructed data
-

Use dataset with initial values $X = [1.0, 2.0]$, $Y = [0.5, 1.5]$

- Initialize neural network with random weights
 - Compute output using linear activation
 - Calculate MAE and MSE
 - Plot loss surface (weight vs loss)
-

Use CIFAR-10 Dataset

- Train CNN with and without data augmentation
 - Augmentations: rotate, zoom, distort images
 - Compare validation accuracy and loss and plot using Graph
-

Implement XOR gate using 2-layer Neural Network

- Use Adadelta optimizer
- Plot accuracy vs epoch

Fashion-MNIST Classification

- CNN with RMSProp & Adam
 - Compare confusion matrices
-

Implement the backpropagation algorithm.

1. Take Iris Dataset
 2. Initialize a neural network with random weights.
 3. Calculate error
 4. Perform multiple iterations of NN
 5. Update weights accordingly.
 6. Plot accuracy for iterations and note the results.
-

Build a multiclass image categorization of CNN network which correctly classifies different categories of images in the dataset.

1. Take Flower dataset
2. Split original dataset to train and test set

-
3. Build CNN Model
 4. Generate the accuracy of the built model using any optimizer.
 5. Compare performance of different optimizers on Flower categorization.
-

Train a small neural network (dataset – Cifar- 100 Classification)

Compare the optimizers:

- Adagrad
- SGD
- Adam

Plot:

- Training loss vs epochs
 - Accuracy vs epochs
-

- Use IRIS Dataset
 - Train a model with and without data augmentation (horizontal flip, rotation, noise).
 - Compare generalization performance on the validation set. (Accuracy & Error)
 - Plot accuracy vs epochs
 - Plot loss vs epochs
-

Task 1: Build a small CNN for MNIST digits dataset

- Split dataset into train/test
- Use 2 convolution layers + pooling + dense
- Metrics / Plots: Accuracy, Confusion Matrix, Precision & Recall, plot training vs validation accuracy and loss

Task 2: Compare Adam vs SGD optimizer

- Metrics / Plots: Plot training loss & accuracy for each optimizer
 - Calculate and plot all activation functions (Sigmoid and tanh) for input ranging in (-10, +10)
 - Calculate and plot Derivative of given Activation function and plot also observe the behaviour of curves.
 - Consider a target vector Y and prediction vector \hat{Y} . Calculate MSE and MAE.
 - Calculate and plot all activation functions (Tanh and Relu) for input ranging in (-5, +5)
 - Calculate and plot Derivative of given Activation function and plot also observe the behaviour of curves.
 - Consider a target vector Y and prediction vector \hat{Y} . Calculate MSE and MAE.
 - 1. Calculate and plot all activation functions (Sigmoid, Relu and softmax) for input ranging in (-10 to +10)
 - 2. Calculate and plot Derivative of given Activation function and plot also observe the behaviour of curves.
 - 3. Consider a target vector Y and prediction vector \hat{Y} . Calculate MSE and MAE.
-

Task 1

Implement the AND Boolean logic gate using perceptron Neural Network. Inputs = x_1, x_2 and bias, weights should be fed into the perceptron with single Output = y . Display final weights and bias of each perceptron.

Task 2

1. Use the titanic Dataset

-
- 2.Create an Auto Encoder and fit it with our data using 3 neurons in the dense layer
 3. Display new reduced dimension values
 - 4.Plot loss for different encoders [Sparse Autoencoder, Noise Autoencoder]
-

Use dataset – MNIST digit classification

Create a neural network and apply following optimizers

- SGD
- SGD + Momentum
- Adam

Plot the comparison using ROC curve

1. Use MNIST or IRIS/ Cifar-10 Dataset
 2. Train a model with and without data augmentation (horizontal flip, rotation, noise).
 3. Compare generalization performance on the validation set. (Accuracy & Error)
 4. Observe improvements and plot the graph.
-

- Load California Housing dataset, select 2 features (e.g., Median Income, House Age) and 1 target (Median House Value).
 - Normalize inputs and initialize a single-layer neural network with random weights and bias.
 - Perform forward propagation and calculate prediction error, Squared Error, and MSE.
 - Update weights and bias using gradient descent.
 - Visualize or analyze how loss changes with weight and bias and observe the error surface.
-

- Load Iris dataset
 - Normalize inputs and create an autoencoder with 2–3 neurons in the dense layer.
 - Encode and decode the inputs.
 - Calculate reconstruction error (MSE) and analyze the results.
 - Observe loss progression over epochs and compare with different encoder configurations.
-

- Load a small image dataset (Intel Image or CIFAR-10 subset).
 - Preprocess images and split into training and testing sets.
 - Build and train a CNN model.
 - Evaluate the model using confusion matrix, precision, recall, and accuracy.
 - Analyze training vs validation metrics over epochs to check generalization.
-

- Load MNIST or Fashion-MNIST dataset.
 - Train a small neural network with different optimizers: SGD, SGD+Momentum, Adam.
 - Evaluate accuracy and ROC curve for each optimizer.
 - Compare training loss, validation loss, and overall performance across optimizers.
-

- Load California Housing dataset and select 2 features (e.g., Median Income, House Age) and 1 target (Median House Value).
- Normalize inputs and initialize a single-layer neural network with random weights and bias.
- Perform forward propagation and calculate prediction error, Squared Error, and MSE.

- Update weights and bias using gradient descent.
 - Visualize or analyze how loss changes with weight and bias and observe the error surface.
-

Task 1: Implement a small CNN for CIFAR-10 subset

- Preprocess & normalize images
- Convolution + Pooling + Flatten + Dense layers
- Metrics / Plots: Accuracy, Confusion Matrix, Precision & Recall, plot training vs validation accuracy/loss

Task 2: Apply data augmentation (flip, rotate, noise)

- Compare performance with original data
 - Metrics / Plots: Plot accuracy vs epochs and loss vs epochs
-