Ayush Chanchal

Sap:500097569

Batch: B6

EXPERIMENT:05

Description:

Observing how learning is achieved in the MNIST handwritten digits dataset involves training a neural network on this dataset and analyzing its performance over time.

Load the MNIST Dataset:

Download the MNIST dataset. It consists of a training set and a test set of handwritten digits.

Each image is 28x28 pixels, and each pixel value is between 0 and 255, representing grayscale intensity.

Preprocess the Data:

Normalize the pixel values to be between 0 and 1. This helps in faster convergence during training.

Design a Neural Network:

Create a neural network architecture suitable for image classification. A simple feedforward neural network or a convolutional neural network (CNN) is commonly used for this task.

Compile the Model:

Define the loss function (typically categorical cross-entropy for classification tasks) and the optimizer (e.g., stochastic gradient descent).

Train the Model:

Train the model using the training data. This involves forward pass (predicting) and backward pass (backpropagating the error) through the network.

Evaluate the Model:

Use the test set to evaluate the model's performance. Look at metrics like accuracy, precision, recall, etc.

Visualize the Learning Curve:

Plot the training and validation loss and accuracy over epochs. This shows how the model's performance changes over time.

Inspect Misclassified Samples:

Look at some of the samples the model got wrong. This can give insights into what types of images are challenging for the model.

Interpretation:

Analyze what the network is learning. For example, in convolutional layers, you can look at the learned filters.

Experiment:

Try different network architectures, regularization techniques, or even different deep learning frameworks to see how they affect learning.

Q). Implement the competitive learning algorithm using a simple neural network and train it using data from the MNIST Handwritten digits dataset, observing how learning is achieved.

Source code:

import matplotlib.pyplot as plt

import numpy as np

import pandas as pd

train\_df=pd.read\_csv("mnist\_train.csv")

train\_image=dict()

for id,row in train\_df.iterrows():

x=row[1:785].to\_numpy()

train\_image[id]={'label': row[0],'data' : row[1:785].to\_numpy() / np.sqrt(np.dot(x, x))}

def normalize(vec):

return vec / np.sqrt(vec.dot(vec))

alpha= 0.20

iterations = 100000

n\_output = 10

n\_pixels = 784

n\_samples = len(train\_image)

rng = np.random.default\_rng()

w\_chg\_ls = list()

W = np.random.rand(n\_output, n\_pixels)

for x in range(n\_output):

W[x] = normalize(W[x])

def training():

for t in range(iterations):

if t % 1000:

w\_chg\_ls.append(np.copy(W))

rand\_i = rng.integers(n\_samples)

input\_vec=train\_image[rand\_i]['data']

win\_index=np.argmax(np.dot(W,input\_vec))

W[win\_index]+=alpha\*(input\_vec-W[win\_index])

return W

weights = training()

def visualization():

fig, ax = plt.subplots(nrows=10, ncols=5, figsize=(15, 15))

for x in range(10):

for w in range(5):

ax[x,w].imshow(w\_chg\_ls[w\*100][x].reshape((28,28)))

ax[x,w].axis('off')

plt.show()

ScreenShot:

