**Design and implement a CNN model (with 2 layers of convolutions) to classify multi category image datasets. Record the accuracy corresponding to the number of epochs. Use the MNIST, CIFAR-10 datasets.**

import tensorflow as tf

from tensorflow.keras import layers, models

import matplotlib.pyplot as plt

import numpy as np

# Load MNIST dataset

(x\_train\_mnist, y\_train\_mnist), (x\_test\_mnist, y\_test\_mnist) = tf.keras.datasets.mnist.load\_data()

x\_train\_mnist, x\_test\_mnist = x\_train\_mnist / 255.0, x\_test\_mnist / 255.0

# Reshape MNIST images to (28,28,1) for CNN input

x\_train\_mnist = x\_train\_mnist.reshape(-1, 28, 28, 1)

x\_test\_mnist = x\_test\_mnist.reshape(-1, 28, 28, 1)

# Load CIFAR-10 dataset

(x\_train\_cifar, y\_train\_cifar), (x\_test\_cifar, y\_test\_cifar) = tf.keras.datasets.cifar10.load\_data()

x\_train\_cifar, x\_test\_cifar = x\_train\_cifar / 255.0, x\_test\_cifar / 255.0

# Define CNN model

def create\_cnn\_model(input\_shape, num\_classes):

model = models.Sequential([

layers.Conv2D(32, (3,3), activation='relu', input\_shape=input\_shape),

layers.MaxPooling2D((2,2)),

layers.Conv2D(64, (3,3), activation='relu'),

layers.MaxPooling2D((2,2)),

layers.Flatten(),

layers.Dense(64, activation='relu'),

layers.Dense(num\_classes, activation='softmax')

])

model.compile(optimizer='adam',

loss='sparse\_categorical\_crossentropy',

metrics=['accuracy'])

return model

# Train and evaluate on MNIST

epochs = 10

mnist\_model = create\_cnn\_model((28, 28, 1), 10)

history\_mnist = mnist\_model.fit(x\_train\_mnist, y\_train\_mnist, epochs=epochs, validation\_data=(x\_test\_mnist, y\_test\_mnist))

# Train and evaluate on CIFAR-10

cifar\_model = create\_cnn\_model((32, 32, 3), 10)

history\_cifar = cifar\_model.fit(x\_train\_cifar, y\_train\_cifar, epochs=epochs, validation\_data=(x\_test\_cifar, y\_test\_cifar))

# Plot accuracy vs. epochs

def plot\_accuracy(history, dataset\_name):

plt.plot(history.history['accuracy'], label='Train Accuracy')

plt.plot(history.history['val\_accuracy'], label='Validation Accuracy')

plt.xlabel('Epochs')

plt.ylabel('Accuracy')

plt.title(f'Accuracy vs. Epochs for {dataset\_name}')

plt.legend()

plt.show()

plot\_accuracy(history\_mnist, "MNIST")

plot\_accuracy(history\_cifar, "CIFAR-10")