**AML Algorithm #13 : neural network implementation to solve MNIST**

import tensorflow as tf

from tensorflow.keras import layers, models

from tensorflow.keras.datasets import mnist

from tensorflow.keras.utils import to\_categorical

from PIL import Image

import numpy as np

# Load and preprocess the MNIST dataset

(train\_images, train\_labels), (test\_images, test\_labels) = mnist.load\_data()

train\_images = train\_images.reshape((60000, 28, 28, 1)).astype('float32') / 255

test\_images = test\_images.reshape((10000, 28, 28, 1)).astype('float32') / 255

train\_labels = to\_categorical(train\_labels)

test\_labels = to\_categorical(test\_labels)

# Build the neural network model

model = models.Sequential()

model.add(layers.Conv2D(32, (3, 3), activation='relu', input\_shape=(28, 28, 1)))

model.add(layers.MaxPooling2D((2, 2)))

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

model.add(layers.MaxPooling2D((2, 2)))

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

model.add(layers.Flatten())

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

model.add(layers.Dense(10, activation='softmax'))

# Compile the model

model.compile(optimizer='adam',

loss='categorical\_crossentropy',

metrics=['accuracy'])

# Train the model only if it hasn't been trained before

try:

model.load\_weights('mnist\_model.h5')

except (OSError, ValueError):

model.fit(train\_images, train\_labels, epochs=5, batch\_size=64, validation\_split=0.2)

model.save\_weights('mnist\_model.h5')

# Load your own image for prediction

image\_path = 'image.jpg' # Replace with the path to your image

input\_image = Image.open(image\_path).convert('L') # Convert to grayscale

input\_image = input\_image.resize((28, 28)) # Resize to 28x28 pixels

input\_data = np.array(input\_image).reshape((1, 28, 28, 1)).astype('float32') / 255

# Make predictions

predictions = model.predict(input\_data)

predicted\_class = np.argmax(predictions)

print("Predicted Class:", predicted\_class)

**Input :**

****

**OUTPUT :**

1/1 [==============================] - 0s 130ms/step

Predicted Class: 7

