

RECOGNIZING HANDWRITTEN DIGITS WITH DEEP LEARNING FOR SMARTER AI APPLICATIONS

```
# Import necessary libraries

import numpy as np

import matplotlib.pyplot as plt

from tensorflow.keras import datasets, layers, models

from tensorflow.keras.utils import to_categorical

from sklearn.model_selection import train_test_split


# Load the MNIST dataset

(train_images, train_labels), (test_images, test_labels) = datasets.mnist.load_data()


# Preprocess the data

# Reshape images to 28x28x1 (grayscale)

train_images = train_images.reshape((train_images.shape[0], 28, 28, 1))

test_images = test_images.reshape((test_images.shape[0], 28, 28, 1))


# Normalize the pixel values to be between 0 and 1

train_images, test_images = train_images / 255.0, test_images / 255.0


# Convert labels to one-hot encoding

train_labels = to_categorical(train_labels, 10)

test_labels = to_categorical(test_labels, 10)


# Define the CNN model

model = models.Sequential([

    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)),

    layers.MaxPooling2D((2, 2)),

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

    layers.MaxPooling2D((2, 2)),

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

RECOGNIZING HANDWRITTEN DIGITS WITH DEEP LEARNING FOR SMARTER AI APPLICATIONS

```
layers.Flatten(),  
layers.Dense(64, activation='relu'),  
layers.Dense(10, activation='softmax') # 10 classes for 0-9 digits  
)
```

Compile the model

```
model.compile(optimizer='adam',  
              loss='categorical_crossentropy',  
              metrics=['accuracy'])
```

Train the model

```
history = model.fit(train_images, train_labels, epochs=5, batch_size=64, validation_split=0.1)
```

```
Epoch 1/5  
844/844 ————— 51s 57ms/step - accuracy: 0.8581 - loss: 0.4493 - val_accuracy: 0.9835 - val_loss: 0.0558  
Epoch 2/5  
844/844 ————— 81s 56ms/step - accuracy: 0.9823 - loss: 0.0584 - val_accuracy: 0.9872 - val_loss: 0.0473  
Epoch 3/5  
844/844 ————— 82s 56ms/step - accuracy: 0.9879 - loss: 0.0399 - val_accuracy: 0.9913 - val_loss: 0.0343  
Epoch 4/5  
844/844 ————— 81s 55ms/step - accuracy: 0.9899 - loss: 0.0299 - val_accuracy: 0.9873 - val_loss: 0.0428  
Epoch 5/5  
844/844 ————— 82s 55ms/step - accuracy: 0.9930 - loss: 0.0214 - val_accuracy: 0.9907 - val_loss: 0.0341
```

Evaluate the model on the test set

```
test_loss, test_acc = model.evaluate(test_images, test_labels)  
print(f"Test accuracy: {test_acc * 100:.2f}%")
```

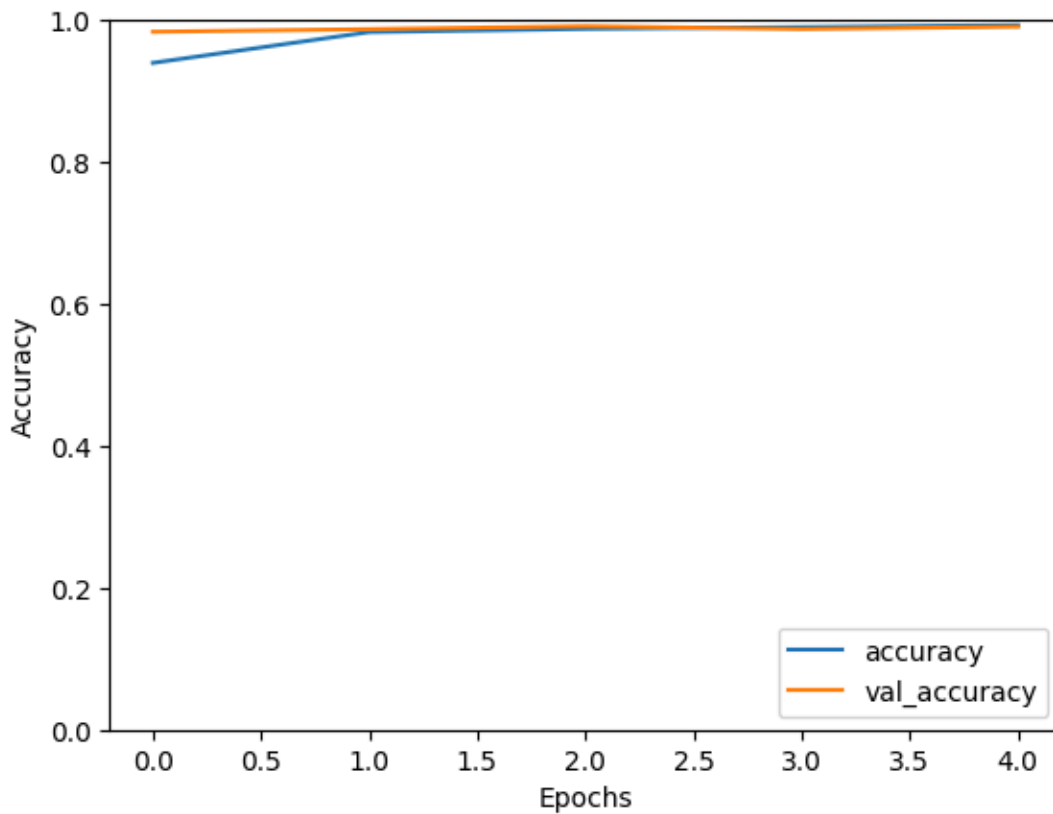
```
313/313 ————— 3s 9ms/step - accuracy: 0.9803 - loss: 0.0575  
Test accuracy: 98.38%
```

Save the model (optional)

```
model.save('mnist_cnn_model.h5')
```

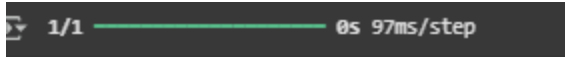
RECOGNIZING HANDWRITTEN DIGITS WITH DEEP LEARNING FOR SMARTER AI APPLICATIONS

```
# Plot the training history (optional)
plt.plot(history.history['accuracy'], label='accuracy')
plt.plot(history.history['val_accuracy'], label = 'val_accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.ylim([0, 1])
plt.legend(loc='lower right')
plt.show()
```



```
# Predictions (optional)
predictions = model.predict(test_images[:5])
```

RECOGNIZING HANDWRITTEN DIGITS WITH DEEP LEARNING FOR SMARTER AI APPLICATIONS



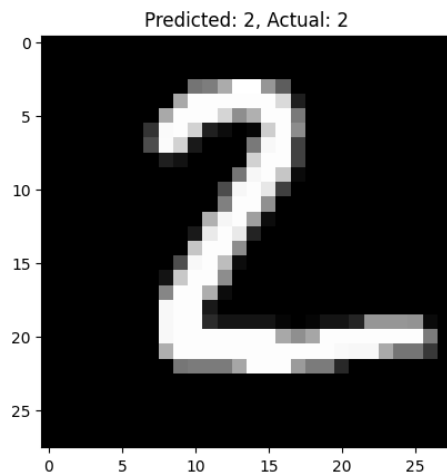
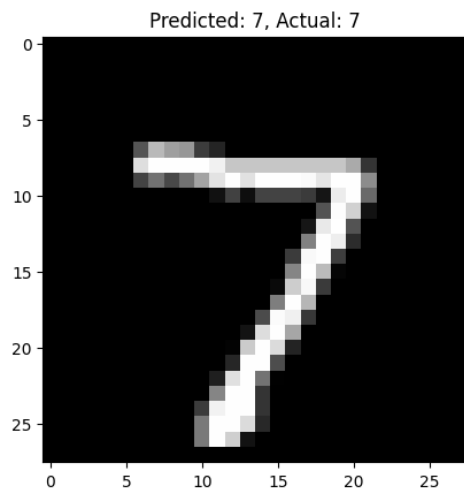
Show the predicted digits and the actual digits

for i in range(5):

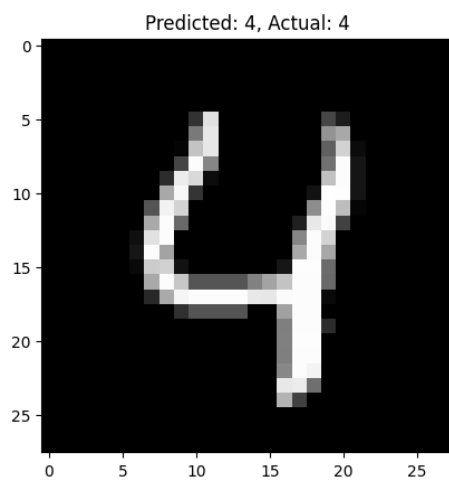
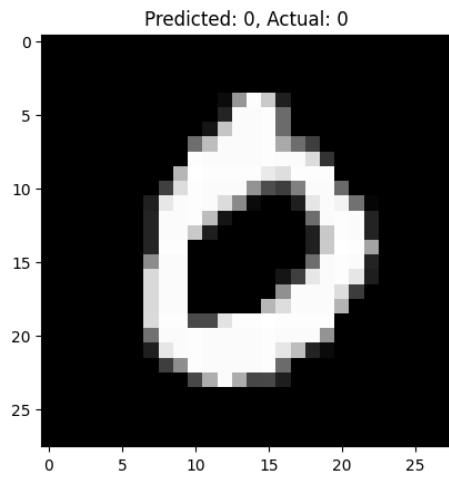
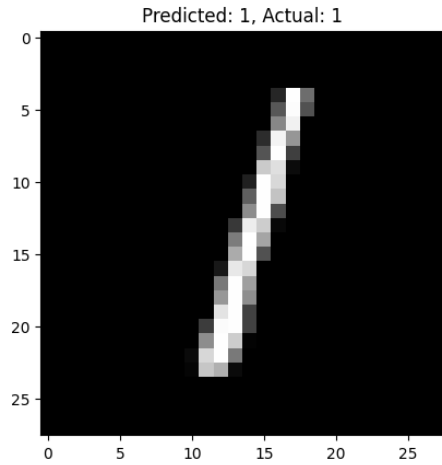
```
    plt.imshow(test_images[i].reshape(28, 28), cmap='gray')
```

```
    plt.title(f"Predicted: {np.argmax(predictions[i])}, Actual: {np.argmax(test_labels[i])}")
```

```
    plt.show()
```



RECOGNIZING HANDWRITTEN DIGITS WITH DEEP LEARNING FOR SMARTER AI APPLICATIONS



GOOGLE COLAB PROJECT LINK:

https://colab.research.google.com/drive/1KyJoM5cLHNjfw59Fmw30DSdAID-isg_T