Recognizing handwritten digits with deep learning for smarter AI applications

Googlecolab workspace

New Section

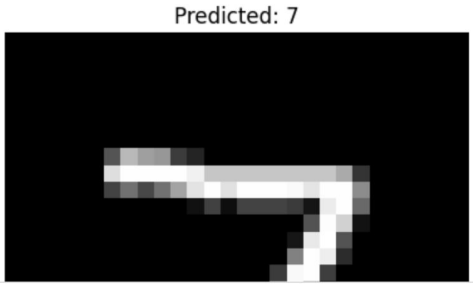
7m

```python
import tensorflow as tf
from tensorflow.keras import layers, models
import matplotlib.pyplot as plt

# Load MNIST dataset
(x_train, y_train), (x_test, y_test) = tf.keras.datasets.mnist.load_data()

# Normalize the data
x_train, x_test = x_train / 255.0, x_test / 255.0

# Reshape data to add channel dimension (needed for CNN)
x_train = x_train.reshape(-1, 28, 28, 1)
x_test = x_test.reshape(-1, 28, 28, 1)

# Build the 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.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dense(10, activation='softmax')
])

# Compile the model
model.compile(optimizer='adam',
```

```python
# Compile the model
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

# Train the model
model.fit(x_train, y_train, epochs=5, validation_split=0.1)

# Evaluate the model
test_loss, test_acc = model.evaluate(x_test, y_test)
print(f"Test accuracy: {test_acc:.2f}")

# Predict and show a sample
import numpy as np
sample = np.expand_dims(x_test[0], axis=0)
prediction = model.predict(sample)
print("Predicted digit:", np.argmax(prediction))

# Show the image
plt.imshow(x_test[0].reshape(28,28), cmap='gray')
plt.title(f"Predicted: {np.argmax(prediction)}")
plt.axis('off')
plt.show()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
/usr/local/lib/python3.11/dist-packages/keras/src/layers/convolutional/base_conv.py:107: UserWarning: Do not pass a
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 55s 31ms/step - accuracy: 0.8938 - loss: 0.3474 - val_accuracy: 0.9818 - val_loss: 0
Epoch 2/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 84s 32ms/step - accuracy: 0.9838 - loss: 0.0518 - val_accuracy: 0.9880 - val_loss: 0
Epoch 3/5
```

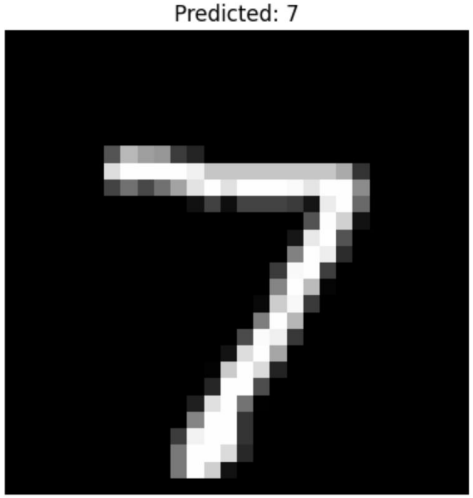
```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
/usr/local/lib/python3.11/dist-packages/keras/src/layers/convolutional/base_conv.py:107: UserWarning: Do not pass a
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 55s 31ms/step - accuracy: 0.8938 - loss: 0.3474 - val_accuracy: 0.9818 - val_loss: 0
Epoch 2/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 84s 32ms/step - accuracy: 0.9838 - loss: 0.0518 - val_accuracy: 0.9880 - val_loss: 0
Epoch 3/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 79s 31ms/step - accuracy: 0.9886 - loss: 0.0342 - val_accuracy: 0.9887 - val_loss: 0
Epoch 4/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 52s 30ms/step - accuracy: 0.9927 - loss: 0.0234 - val_accuracy: 0.9893 - val_loss: 0
Epoch 5/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 83s 31ms/step - accuracy: 0.9938 - loss: 0.0188 - val_accuracy: 0.9915 - val_loss: 0
313/313 ━━━━━━━━━━━━━━━━━━━━ 3s 9ms/step - accuracy: 0.9876 - loss: 0.0352
Test accuracy: 0.99
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 110ms/step
Predicted digit: 7
```



```
print("Predicted digit:", np.argmax(prediction))

# Show the image
plt.imshow(x_test[0].reshape(28,28), cmap='gray')
plt.title(f"Predicted: {np.argmax(prediction)}")
plt.axis('off')
plt.show()
```

```
s/mnist.npz

.py:107: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential models, pre

 val_accuracy: 0.9818 - val_loss: 0.0620

 val_accuracy: 0.9880 - val_loss: 0.0380

 val_accuracy: 0.9887 - val_loss: 0.0423

 val_accuracy: 0.9893 - val_loss: 0.0372

 val_accuracy: 0.9915 - val_loss: 0.0328
```

```
Test accuracy: 0.99
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 110ms/step
Predicted digit: 7
```




```
[ ]  Start coding or generate with AI.
```