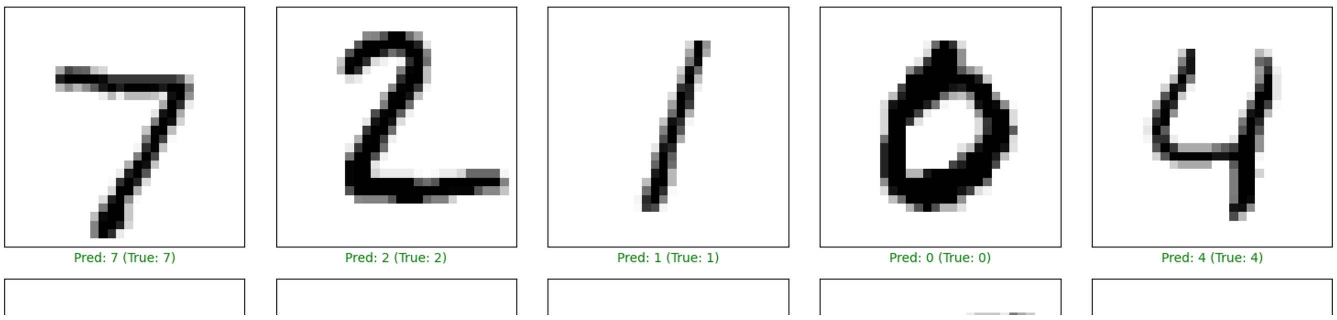
```
import numpy as np
import struct
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
import matplotlib.pyplot as plt

def read_idx(filename):
    """Proper IDX file reader that handles both images and labels"""
    with open(filename, 'rb') as f:
        zero, data_type, dims = struct.unpack('>HBB', f.read(4))
        shape = tuple(struct.unpack('>I', f.read(4))[0] for _ in range(dims))
        return np.frombuffer(f.read(), dtype=np.uint8).reshape(shape)

# Load the actual MNIST dataset (train and test separately)
try:
    train_images = read_idx('train-images-idx3-ubyte')
    train_labels = read_idx('train-labels-idx1-ubyte')
    test_images = read_idx('t10k-images-idx3-ubyte')
    test_labels = read_idx('t10k-labels-idx1-ubyte')
except FileNotFoundError:
    # Fallback to Keras dataset if files not found
    (train_images, train_labels), (test_images, test_labels) = keras.datasets.mnist.load_data()

# Normalize and reshape data
train_images = train_images.astype('float32') / 255.0
test_images = test_images.astype('float32') / 255.0
train_images = np.expand_dims(train_images, -1)  # Add channel dimension
test_images = np.expand_dims(test_images, -1)   # Add channel dimension

# Improved CNN model
model = keras.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)),
    layers.BatchNormalization(),
    layers.Conv2D(32, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Dropout(0.25),

    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.BatchNormalization(),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Dropout(0.25),

    layers.Flatten(),
    layers.Dense(256, activation='relu'),
    layers.BatchNormalization(),
    layers.Dropout(0.5),
    layers.Dense(10, activation='softmax')
])

# Improved training configuration
model.compile(
    optimizer=keras.optimizers.Adam(0.001),
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy']
)

# Add early stopping
callbacks = [
    keras.callbacks.EarlyStopping(patience=3, restore_best_weights=True)
]

# Train with validation split
history = model.fit(
    train_images, train_labels,
    batch_size=128,
    epochs=30,
    validation_split=0.1,
    callbacks=callbacks
)

# Evaluate on test set
test_loss, test_acc = model.evaluate(test_images, test_labels, verbose=0)
print(f'\nTest accuracy: {test_acc*100:.2f}%')

# Visualization
predictions = model.predict(test_images)
plt.figure(figsize=(15, 15))
```

```
for i in range(25):
    plt.subplot(5, 5, i+1)
    plt.xticks([])
    plt.yticks([])
    plt.grid(False)
    plt.imshow(test_images[i].squeeze(), cmap=plt.cm.binary)
    pred_label = np.argmax(predictions[i])
    true_label = test_labels[i]
    color = 'green' if pred_label == true_label else 'red'
    plt.xlabel(f"Pred: {pred_label} (True: {true_label})", color=color)
plt.tight_layout()
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 an
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/30
422/422 ━━━━━━━━━━━━━━━━━━━━ 137s 314ms/step - accuracy: 0.8356 - loss: 0.5509 - val_accuracy: 0.7192 - val_loss: 0.
Epoch 2/30
422/422 ━━━━━━━━━━━━━━━━━━━━ 131s 288ms/step - accuracy: 0.9740 - loss: 0.0820 - val_accuracy: 0.9890 - val_loss: 0.
Epoch 3/30
422/422 ━━━━━━━━━━━━━━━━━━━━ 144s 293ms/step - accuracy: 0.9829 - loss: 0.0561 - val_accuracy: 0.9902 - val_loss: 0.
Epoch 4/30
422/422 ━━━━━━━━━━━━━━━━━━━━ 144s 297ms/step - accuracy: 0.9841 - loss: 0.0507 - val_accuracy: 0.9918 - val_loss: 0.
Epoch 5/30
422/422 ━━━━━━━━━━━━━━━━━━━━ 142s 299ms/step - accuracy: 0.9875 - loss: 0.0379 - val_accuracy: 0.9903 - val_loss: 0.
Epoch 6/30
422/422 ━━━━━━━━━━━━━━━━━━━━ 144s 304ms/step - accuracy: 0.9888 - loss: 0.0346 - val_accuracy: 0.9918 - val_loss: 0.
Epoch 7/30
422/422 ━━━━━━━━━━━━━━━━━━━━ 141s 302ms/step - accuracy: 0.9908 - loss: 0.0300 - val_accuracy: 0.9918 - val_loss: 0.

Test accuracy: 99.28%
313/313 ━━━━━━━━━━━━━━━━━━━━ 7s 22ms/step
```


Pred: 7 (True: 7)
Pred: 2 (True: 2)
Pred: 1 (True: 1)
Pred: 0 (True: 0)
Pred: 4 (True: 4)