```
'''
ECE471 Selected Topics in Machine Learning - Assignment 2
Submit by Sept. 26, 10pm
tldr: Classify mnist digits with a convolutional neural network. Get at least
95.5% accuracy on the test test.
'''

import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.datasets.mnist import load_data

# Fix seed
np.random.seed(1618)
tf.set_random_seed(1618)

# Data parameters
NUM_CLASSES = 10
HEIGHT = 28
WIDTH = 28
NUM_CHANNELS = 1

# Load data and split into train, val, test sets
(x_train, y_train), (x_test, y_test) = load_data()
(x_train, y_train), (x_val, y_val) = \
    (x_train[:50000], y_train[:50000]), (x_train[50000:], y_train[50000:])

# Normalize and reshape data and labels
x_train, x_val, x_test = \
    map(lambda x: (x / 255.0).reshape([-1, HEIGHT, WIDTH, NUM_CHANNELS]),
        [x_train, x_val, x_test])
y_train, y_val, y_test = \
    map(lambda y: keras.utils.to_categorical(y, NUM_CLASSES),
        [y_train, y_val, y_test])

# Hyperparameters
BATCH_SIZE = 32
NUM_EPOCHS = 2

# Create CNN using Keras API
activation = keras.activations.relu
regularizer = keras.regularizers.l2(l=0.05)

model = keras.Sequential()
model.add(keras.layers.Conv2D(32, 5, activation=activation,
                              kernel_regularizer=regularizer))
model.add(keras.layers.Conv2D(64, 3, activation=activation,
                              kernel_regularizer=regularizer))
model.add(keras.layers.MaxPooling2D(3))
model.add(keras.layers.Dropout(0.25))
model.add(keras.layers.Flatten())
model.add(keras.layers.Dense(128, activation=activation))
model.add(keras.layers.Dropout(0.5))
model.add(keras.layers.Dense(NUM_CLASSES, activation='softmax'))

model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

model.fit(x_train, y_train,
          batch_size=BATCH_SIZE,
          epochs=NUM_EPOCHS,
          verbose=1,
          validation_data=[x_val, y_val])

loss, acc = model.evaluate(x_test, y_test, verbose=1)
num_params = np.sum([np.prod(v.get_shape().as_list())
                     for v in tf.trainable_variables()])
```

```
print('Test loss:', loss)
print('Test accuracy:', acc)
print('Number of parameters:', num_params)

''' Output
Train on 50000 samples, validate on 10000 samples
Epoch 1/2
2018-09-22 20:49:57.609976: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions t
hat this TensorFlow binary was not compiled to use: AVX2 FMA
50000/50000 [==============================] - 125s 2ms/step - loss: 0.5212 - acc: 0.8933 - val_loss: 0.2
130 - val_acc: 0.9696
Epoch 2/2
50000/50000 [==============================] - 125s 2ms/step - loss: 0.2682 - acc: 0.9471 - val_loss: 0.1
701 - val_acc: 0.9752
10000/10000 [==============================] - 6s 625us/step
Test loss: 0.164031094706
Test accuracy: 0.9751
Number of parameters: 1266475.0
'''
```