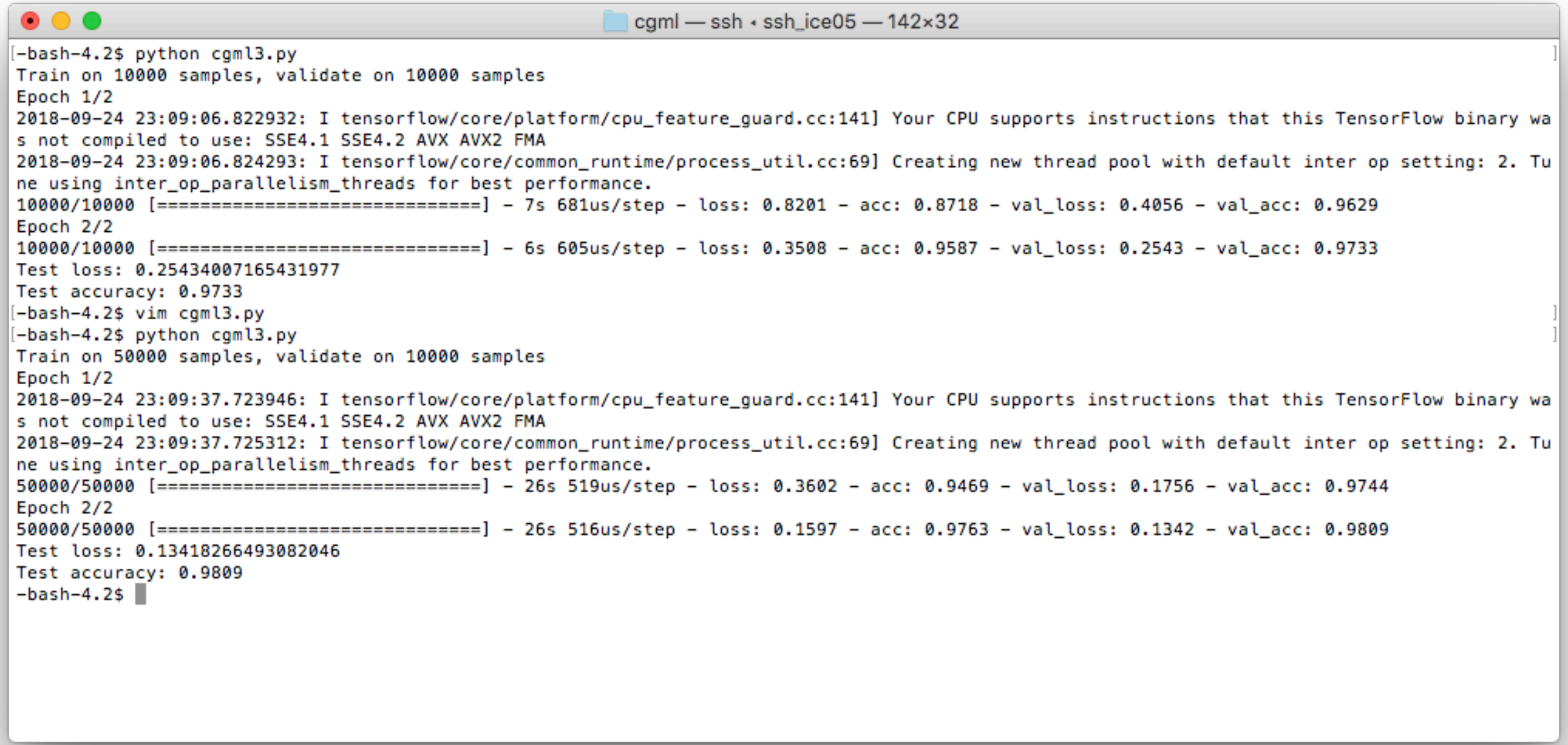
```
#Camille Chow
#ECE 471 Assignment 3
import numpy as np
import tensorflow as tf

#dimensional constants
num_classes = 10
image_h = 28
image_w = 28
channels = 1
input_shape = (image_h, image_w, channels)
val_set_size = 10000

#tunable hyperparams
batch_size = 50
epochs = 2
kernel_size = 3
pool_size = 2
a_fcn = 'relu'
dropout = .35
lam = .001

#get data
mnist = tf.keras.datasets.mnist
(x_train, y_train),(x_test, y_test) = mnist.load_data()

#shape test data
x_test = x_test.reshape(x_test.shape[0], image_h, image_w, channels)
x_test = x_test.astype('float32')
x_test /= 255.0
y_test = tf.keras.utils.to_categorical(y_test, num_classes)

#shape training data
x_train = x_train.reshape(x_train.shape[0], image_h, image_w, channels)
x_train = x_train.astype('float32')
x_train /= 255.0
y_train = tf.keras.utils.to_categorical(y_train, num_classes)

#split into training/validation sets
x_val = x_train[:val_set_size]
y_val = y_train[:val_set_size]
x_train = x_train[val_set_size:]
y_train = y_train[val_set_size:]

#build cnn
model = tf.keras.Sequential()
model.add(tf.keras.layers.Conv2D(32, kernel_size=kernel_size, strides=(1, 1), activation=
a_fcn, input_shape=input_shape))
model.add(tf.keras.layers.MaxPooling2D(pool_size=pool_size, strides=pool_size))
model.add(tf.keras.layers.Conv2D(64, kernel_size=kernel_size, activation=a_fcn))
model.add(tf.keras.layers.MaxPooling2D(pool_size=pool_size))
model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(500, activation=a_fcn, kernel_regularizer=tf.keras.regula
rizers.l2(lam)))
model.add(tf.keras.layers.Dropout(dropout))
model.add(tf.keras.layers.Dense(num_classes, activation='softmax'))

#train model
model.compile(loss=tf.keras.losses.categorical_crossentropy, optimizer=tf.keras.optimizer
s.Adam(), metrics=['accuracy'])
model.fit(x_train, y_train, batch_size=batch_size, epochs=epochs, verbose=1, validation_d
ata=(x_test, y_test))
#model.fit(x_val, y_val, batch_size=batch_size, epochs=epochs, verbose=1, validation_data
=(x_test, y_test))

#test model
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```



```
[-bash-4.2$ python cgml3.py
Train on 10000 samples, validate on 10000 samples
Epoch 1/2
2018-09-24 23:09:06.822932: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlow binary wa
s not compiled to use: SSE4.1 SSE4.2 AVX AVX2 FMA
2018-09-24 23:09:06.824293: I tensorflow/core/common_runtime/process_util.cc:69] Creating new thread pool with default inter op setting: 2. Tu
ne using inter_op_parallelism_threads for best performance.
10000/10000 [==============================] - 7s 681us/step - loss: 0.8201 - acc: 0.8718 - val_loss: 0.4056 - val_acc: 0.9629
Epoch 2/2
10000/10000 [==============================] - 6s 605us/step - loss: 0.3508 - acc: 0.9587 - val_loss: 0.2543 - val_acc: 0.9733
Test loss: 0.25434007165431977
Test accuracy: 0.9733
[-bash-4.2$ vim cgml3.py
[-bash-4.2$ python cgml3.py
Train on 50000 samples, validate on 10000 samples
Epoch 1/2
2018-09-24 23:09:37.723946: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlow binary wa
s not compiled to use: SSE4.1 SSE4.2 AVX AVX2 FMA
2018-09-24 23:09:37.725312: I tensorflow/core/common_runtime/process_util.cc:69] Creating new thread pool with default inter op setting: 2. Tu
ne using inter_op_parallelism_threads for best performance.
50000/50000 [==============================] - 26s 519us/step - loss: 0.3602 - acc: 0.9469 - val_loss: 0.1756 - val_acc: 0.9744
Epoch 2/2
50000/50000 [==============================] - 26s 516us/step - loss: 0.1597 - acc: 0.9763 - val_loss: 0.1342 - val_acc: 0.9809
Test loss: 0.13418266493082046
Test accuracy: 0.9809
-bash-4.2$
```