Exercise 3

In the videos you looked at how you would improve Fashion MNIST using Convolutions. For your exercise see if you can improve MNIST to 99.8% accuracy or more using only a single convolutional layer and a single MaxPooling 2D. You should stop training once the accuracy goes above this amount. It should happen in less than 20 epochs, so it's ok to hard code the number of epochs for training, but your training must end once it hits the above metric. If it doesn't, then you'll need to redesign your layers.

I've started the code for you -- you need to finish it!

When 99.8% accuracy has been hit, you should print out the string "Reached 99.8% accuracy so cancelling training!"

In [22]:
```
import tensorflow as tf
from os import path, getcwd, chdir

# DO NOT CHANGE THE LINE BELOW. If you are developing in a local
# environment, then grab mnist.npz from the Coursera Jupyter Notebo
ok
# and place it inside a local folder and edit the path to that loca
tion
path = f"{getcwd()}/../tmp2/mnist.npz"
```

In [23]:
```
config = tf.ConfigProto()
config.gpu_options.allow_growth = True
sess = tf.Session(config=config)
```

In [37]:

```
# GRADED FUNCTION: train_mnist_conv
def train_mnist_conv():
    # Please write your code only where you are indicated.
    # please do not remove model fitting inline comments.

    # YOUR CODE STARTS HERE
    class myCallback(tf.keras.callbacks.Callback):
        def on_epoch_end(self, epoch, logs={}):
            if(logs.get('acc')>= 0.998):
                print("\nReached 99% accuracy so cancelling trainin
g!")
                self.model.stop_training = True
    # YOUR CODE ENDS HERE

    mnist = tf.keras.datasets.mnist
    (training_images, training_labels), (test_images, test_labels)
= mnist.load_data(path=path)
    # YOUR CODE STARTS HERE
    training_images=training_images.reshape(60000, 28, 28, 1)
    training_images=training_images / 255.0
    test_images = test_images.reshape(10000, 28, 28, 1)
    test_images=test_images/255.0
    callbacks = myCallback()
    # YOUR CODE ENDS HERE

    model = tf.keras.models.Sequential([
        # YOUR CODE STARTS HERE
        tf.keras.layers.Conv2D(32, (3,3), activation='relu', input_
shape=(28, 28, 1)),
        tf.keras.layers.Conv2D(64, (3,3), activation='relu', input_
shape=(28, 28, 1)),
        tf.keras.layers.MaxPooling2D(2, 2),
        tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
        tf.keras.layers.MaxPooling2D(2,2),
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(128, activation='relu'),
        tf.keras.layers.Dense(10, activation='softmax')
        # YOUR CODE ENDS HERE
    ])
    model.compile(optimizer='adam', loss='sparse_categorical_crosse
ntropy', metrics=['accuracy'])
    # model fitting
    history = model.fit(
        # YOUR CODE STARTS HERE
        training_images, training_labels, epochs=10, callbacks=[cal
lbacks]
        # YOUR CODE ENDS HERE
    )
    # model fitting
    return history.epoch, history.history['acc'][-1]
```

In [38]:
```
_, _ = train_mnist_conv()
```

```
Epoch 1/10
60000/60000 [==============================] - 18s 293us/sample -
loss: 0.1102 - acc: 0.9657
Epoch 2/10
60000/60000 [==============================] - 17s 288us/sample -
loss: 0.0354 - acc: 0.9888
Epoch 3/10
60000/60000 [==============================] - 18s 294us/sample -
loss: 0.0246 - acc: 0.9922
Epoch 4/10
60000/60000 [==============================] - 17s 285us/sample -
loss: 0.0184 - acc: 0.9940 - loss: 0.0181 - ETA: 2s - loss:
Epoch 5/10
60000/60000 [==============================] - 16s 272us/sample -
loss: 0.0145 - acc: 0.9955
Epoch 6/10
60000/60000 [==============================] - 16s 269us/sample -
loss: 0.0112 - acc: 0.9965
Epoch 7/10
60000/60000 [==============================] - 16s 273us/sample -
loss: 0.0090 - acc: 0.9971
Epoch 8/10
60000/60000 [==============================] - 16s 270us/sample -
loss: 0.0085 - acc: 0.9975
Epoch 9/10
60000/60000 [==============================] - 16s 270us/sample -
loss: 0.0076 - acc: 0.9975
Epoch 10/10
59808/60000 [============================>.] - ETA: 0s - loss: 0.0
059 - acc: 0.9981
Reached 99% accuracy so cancelling training!
60000/60000 [==============================] - 16s 270us/sample -
loss: 0.0058 - acc: 0.9981
```

In [35]:
```
# Now click the 'Submit Assignment' button above.
# Once that is complete, please run the following two cells to save
your work and close the notebook
```

In [ ]:
```
%%javascript
<!-- Save the notebook -->
IPython.notebook.save_checkpoint();
```

In [ ]:
```
%%javascript
IPython.notebook.session.delete();
window.onbeforeunload = null
setTimeout(function() { window.close(); }, 1000);
```