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 [1]:

```python
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 Notebook
# and place it inside a local folder and edit the path to that location
path = f"{getcwd()}/../tmp2/mnist.npz"
```

In [2]:

```python
config = tf.ConfigProto()
config.gpu_options.allow_growth = True
sess = tf.Session(config=config)
```

In [15]:

```
# 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 ImplementedCallback(tf.keras.callbacks.Callback):
        def on_epoch_end(self, epoch, logs={}):
            if logs.get('acc') > 0.998:
                print("\nReached 99.8% accuracy so cancelling training!")
                self.model.stop_training = True
    callback = ImplementedCallback()
    # 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, test_images = training_images.reshape(60000, 28, 28, 1), test_images.reshape(10000, 28, 28, 1)
    training_images, test_images = training_images/255.0, test_images/255.0
    # YOUR CODE ENDS HERE

    model = tf.keras.models.Sequential([
            # YOUR CODE STARTS HERE
            tf.keras.layers.Conv2D(64, (3, 3), activation='relu', input_shape=(28, 28, 1)),
            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_crossentropy', metrics=['accuracy'])
    # model fitting
    history = model.fit(
        # YOUR CODE STARTS HERE
        training_images,
        training_labels,
        epochs=10,
        callbacks=[callback]
        # YOUR CODE ENDS HERE
    )
    # model fitting
    return history.epoch, history.history['acc'][-1]
```

In [16]:

```
_, _ = train_mnist_conv()
```

```
Epoch 1/10
60000/60000 [==============================] - 15s 255us/sample - loss: 0.143
1 - acc: 0.9573
Epoch 2/10
60000/60000 [==============================] - 15s 248us/sample - loss: 0.047
0 - acc: 0.9856
Epoch 3/10
60000/60000 [==============================] - 15s 243us/sample - loss: 0.029
0 - acc: 0.9912
Epoch 4/10
60000/60000 [==============================] - 14s 227us/sample - loss: 0.019
9 - acc: 0.9936
Epoch 5/10
60000/60000 [==============================] - 14s 230us/sample - loss: 0.013
6 - acc: 0.9955
Epoch 6/10
60000/60000 [==============================] - 14s 227us/sample - loss: 0.008
6 - acc: 0.9974
Epoch 7/10
60000/60000 [==============================] - 14s 227us/sample - loss: 0.007
8 - acc: 0.9973
Epoch 8/10
59936/60000 [============================>.] - ETA: 0s - loss: 0.0057 - acc:
0.9980
Reached 99.8% accuracy so cancelling training!
60000/60000 [==============================] - 14s 230us/sample - loss: 0.005
7 - acc: 0.9980
```

In [ ]:

```
# 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);
```