Open in Colab

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!"

Things to Note:

1. When coding the `class myCallback`, Python 3 will run into an error

   ```
   TypeError: '>' not supported between instances of 'NoneType' and 'float'
   ```

   when using the code

   ```
   if(logs.get('accuracy')>0.99):
   ```

   For Python 3, use the following equivalent code line

   ```
   if logs.get('accuracy') is not None and logs.get('accuracy') > 0.99:
   ```

2. You can run the notebook using TensorFlow 2.5.0

```
import tensorflow as tf

print(tf.__version__)
```

```
2.6.0
```

```
# GRADED FUNCTION: train_mnist_conv
```

```
def train_mnist_conv():

    class myCallback(tf.keras.callbacks.Callback):
        def on_epoch_end(self, epoch, logs={}):
            if logs.get('accuracy') is not None and logs.get('accuracy') > 0.998:
                print("\nReached 99.8% accuracy so cancelling training!")
                self.model.stop_training = True

    mnist = tf.keras.datasets.mnist
    (training_images, training_labels), (test_images, test_labels) = mnist.load_data()


    callbacks = myCallback()

    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

    model = tf.keras.models.Sequential([
      tf.keras.layers.Conv2D(32, (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')
    ])

    model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

    # model fitting
    history = model.fit(training_images, training_labels, epochs=10, callbacks=[callbacks]
    )
    # model fitting

    return history.epoch, history.history['accuracy'][-1]


_, _ = train_mnist_conv()
```

```
 Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
 11493376/11490434 [==============================] - 0s 0us/step
 11501568/11490434 [==============================] - 0s 0us/step
 Epoch 1/10
 1875/1875 [==============================] - 38s 4ms/step - loss: 0.1429 - accuracy: 0.9585
```

```
Epoch 2/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0484 - accuracy: 0.9851
Epoch 3/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0308 - accuracy: 0.9905
Epoch 4/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0191 - accuracy: 0.9937
Epoch 5/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0135 - accuracy: 0.9958
Epoch 6/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0089 - accuracy: 0.9971
Epoch 7/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0067 - accuracy: 0.9977
Epoch 8/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0066 - accuracy: 0.9978
Epoch 9/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0050 - accuracy: 0.9983

Reached 99.8% accuracy so cancelling training!
```