FinalProject_code

June 3, 2019

In [1]:
```
import tensorflow as tf
import matplotlib.pyplot as plt
from keras import optimizers
from keras.models import Sequential
from keras.layers import Dense, Dropout, LSTM, BatchNormalization
from keras.callbacks import TensorBoard
from keras.callbacks import ModelCheckpoint
from keras.optimizers import adam
```

```
Using TensorFlow backend.
```

In [2]:
```
mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train = tf.keras.utils.normalize(x_train, axis = 1)
x_test = tf.keras.utils.normalize(x_test, axis = 1)
```

In [3]:
```
def draw_learning_curve(model):
    plt.plot(model.history['acc'])
    plt.plot(model.history['val_acc'])
    plt.title('Model Accuracy')
    plt.ylabel('Accuracy')
    plt.xlabel('Epoch')
    plt.legend(['train', 'test'], loc='upper left')
    plt.show()
    plt.plot(model.history['loss'])
    plt.plot(model.history['val_loss'])
    plt.title('Model loss')
    plt.ylabel('Loss')
    plt.xlabel('Epoch')
    plt.legend(['train', 'test'], loc='upper left')
    plt.show()
```

0.1 No normalization/reg

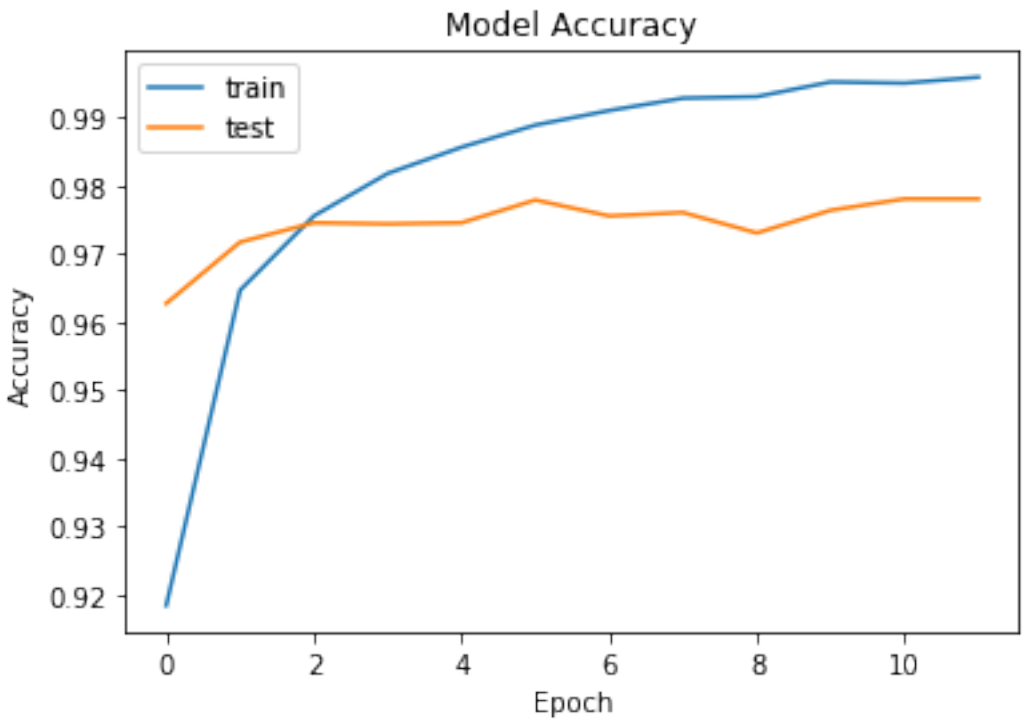
0.1.1 ReLu Activation

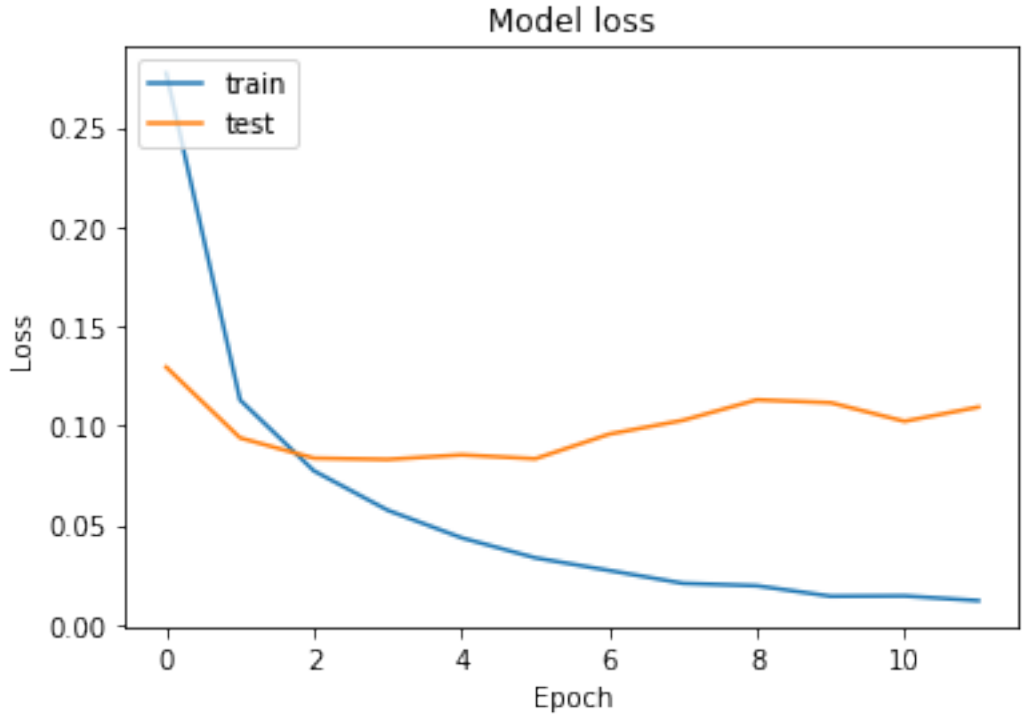
In [4]:
```
model = tf.keras.models.Sequential()
model.add(tf.keras.layers.Flatten())
```

```
        model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
        model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
        model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
        model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics = [
        model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
WARNING:tensorflow:From C:\Users\Miguel\Anaconda3\lib\site-packages\tensorflow\python\ops\resou
Instructions for updating:
Colocations handled automatically by placer.
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 4s 72us/sample - loss: 0.2773 - acc: 0.9184 - va
Epoch 2/12
54000/54000 [==============================] - 4s 66us/sample - loss: 0.1130 - acc: 0.9646 - va
Epoch 3/12
54000/54000 [==============================] - 4s 65us/sample - loss: 0.0773 - acc: 0.9755 - va
Epoch 4/12
54000/54000 [==============================] - 3s 63us/sample - loss: 0.0575 - acc: 0.9817 - va
Epoch 5/12
54000/54000 [==============================] - 3s 63us/sample - loss: 0.0436 - acc: 0.9856 - va
Epoch 6/12
54000/54000 [==============================] - 3s 63us/sample - loss: 0.0335 - acc: 0.9888 - va
Epoch 7/12
54000/54000 [==============================] - 3s 63us/sample - loss: 0.0272 - acc: 0.9909 - va
Epoch 8/12
54000/54000 [==============================] - 3s 63us/sample - loss: 0.0207 - acc: 0.9928 - va
Epoch 9/12
54000/54000 [==============================] - 3s 63us/sample - loss: 0.0195 - acc: 0.9930 - va
Epoch 10/12
54000/54000 [==============================] - 3s 63us/sample - loss: 0.0142 - acc: 0.9951 - va
Epoch 11/12
54000/54000 [==============================] - 3s 63us/sample - loss: 0.0144 - acc: 0.9950 - va
Epoch 12/12
54000/54000 [==============================] - 3s 63us/sample - loss: 0.0120 - acc: 0.9958 - va
```

```
In [5]: draw_learning_curve(model_graph)
```


Model Accuracy
train
test
Accuracy
0.99
0.98
0.97
0.96
0.95
0.94
0.93
0.92
0
2
4
6
8
10
Epoch


Model loss
train
test
Loss
0.25
0.20
0.15
0.10
0.05
0.00
0
2
4
6
8
10
Epoch

```
In [6]: val_loss, val_acc = model.evaluate(x_train, y_train)
        print("Train accuracy:", val_acc)
        val_loss, val_acc = model.evaluate(x_test, y_test)
        print("Test accuracy:", val_acc)
```

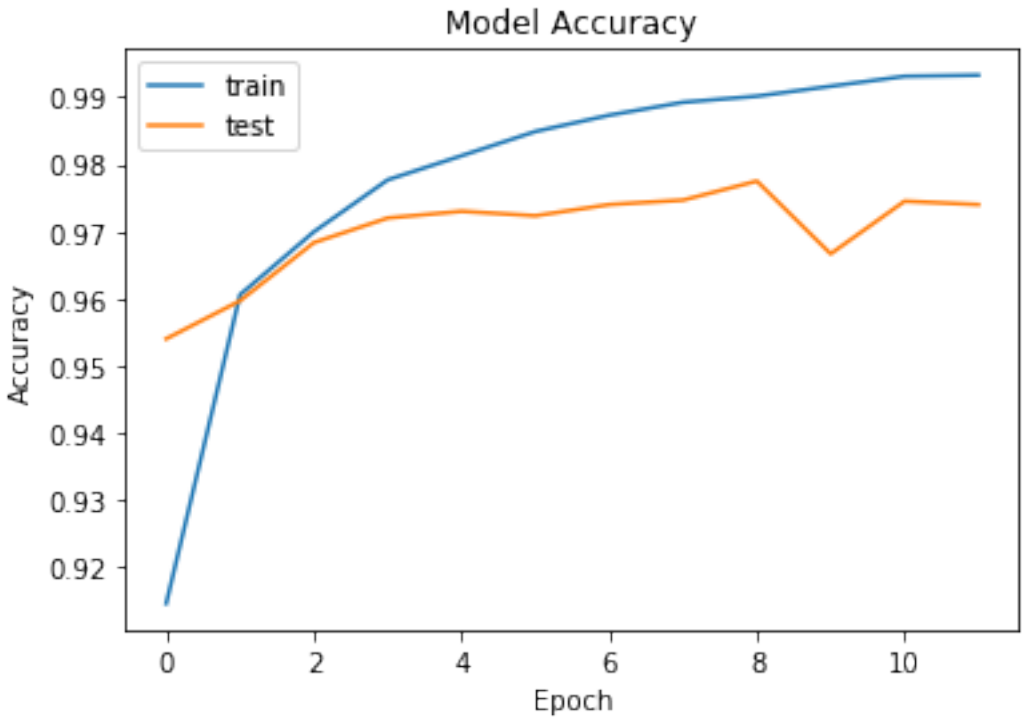
```
60000/60000 [==============================] - 2s 31us/sample - loss: 0.0187 - acc: 0.9951
Train accuracy: 0.99513334
10000/10000 [==============================] - 0s 32us/sample - loss: 0.1160 - acc: 0.9753
Test accuracy: 0.9753
```

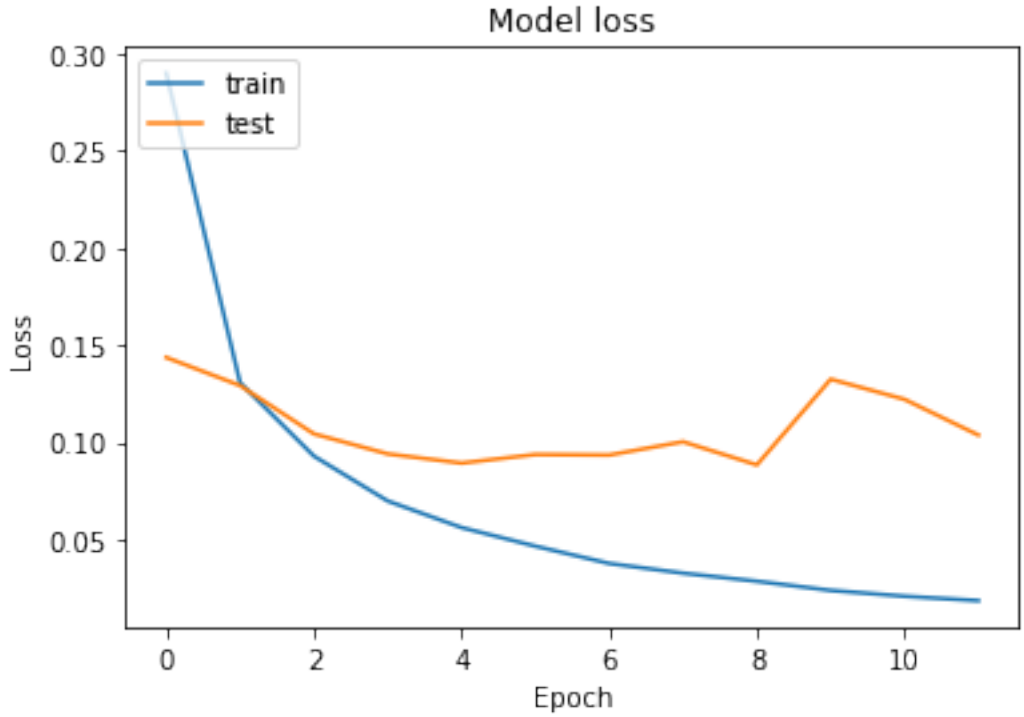
0.1.2 Leaky ReLu Activation

```
In [7]: model = tf.keras.models.Sequential()
        model.add(tf.keras.layers.Flatten())
        model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
        model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
        model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
        model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics = [
        model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.2899 - acc: 0.9146 - va
Epoch 2/12
54000/54000 [==============================] - 3s 63us/sample - loss: 0.1306 - acc: 0.9606 - va
Epoch 3/12
54000/54000 [==============================] - 3s 63us/sample - loss: 0.0928 - acc: 0.9699 - va
Epoch 4/12
54000/54000 [==============================] - 3s 63us/sample - loss: 0.0697 - acc: 0.9777 - va
Epoch 5/12
54000/54000 [==============================] - 4s 65us/sample - loss: 0.0560 - acc: 0.9813 - va
Epoch 6/12
54000/54000 [==============================] - 4s 70us/sample - loss: 0.0465 - acc: 0.9849 - va
Epoch 7/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.0376 - acc: 0.9873 - va
Epoch 8/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.0326 - acc: 0.9892 - va
Epoch 9/12
54000/54000 [==============================] - 3s 63us/sample - loss: 0.0284 - acc: 0.9901 - va
Epoch 10/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.0237 - acc: 0.9916 - va
Epoch 11/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.0207 - acc: 0.9931 - va
Epoch 12/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.0184 - acc: 0.9933 - va
```

```
In [8]: draw_learning_curve(model_graph)
```


Model Accuracy
train
test
Accuracy
0.99
0.98
0.97
0.96
0.95
0.94
0.93
0.92
0
2
4
6
8
10
Epoch


Model loss
train
test
Loss
0.30
0.25
0.20
0.15
0.10
0.05
0
2
4
6
8
10
Epoch

```
In [9]: val_loss, val_acc = model.evaluate(x_train, y_train)
        print("Train accuracy:", val_acc)
        val_loss, val_acc = model.evaluate(x_test, y_test)
        print("Test accuracy:", val_acc)
```

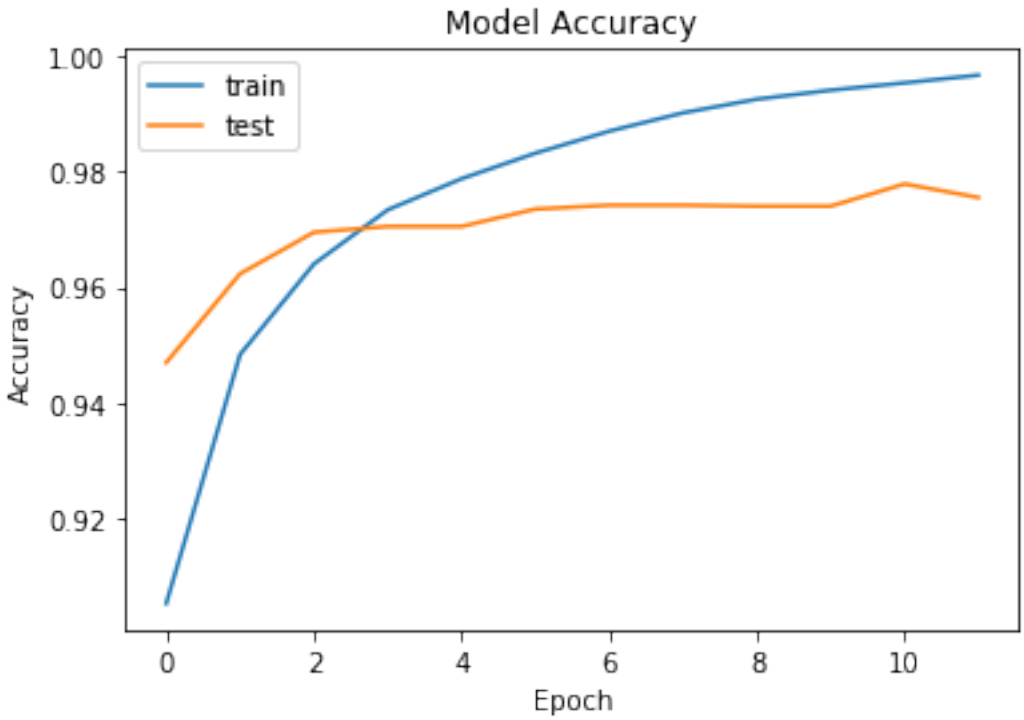
```
60000/60000 [==============================] - 2s 32us/sample - loss: 0.0244 - acc: 0.9928
Train accuracy: 0.9927667
10000/10000 [==============================] - 0s 32us/sample - loss: 0.1201 - acc: 0.9722
Test accuracy: 0.9722
```

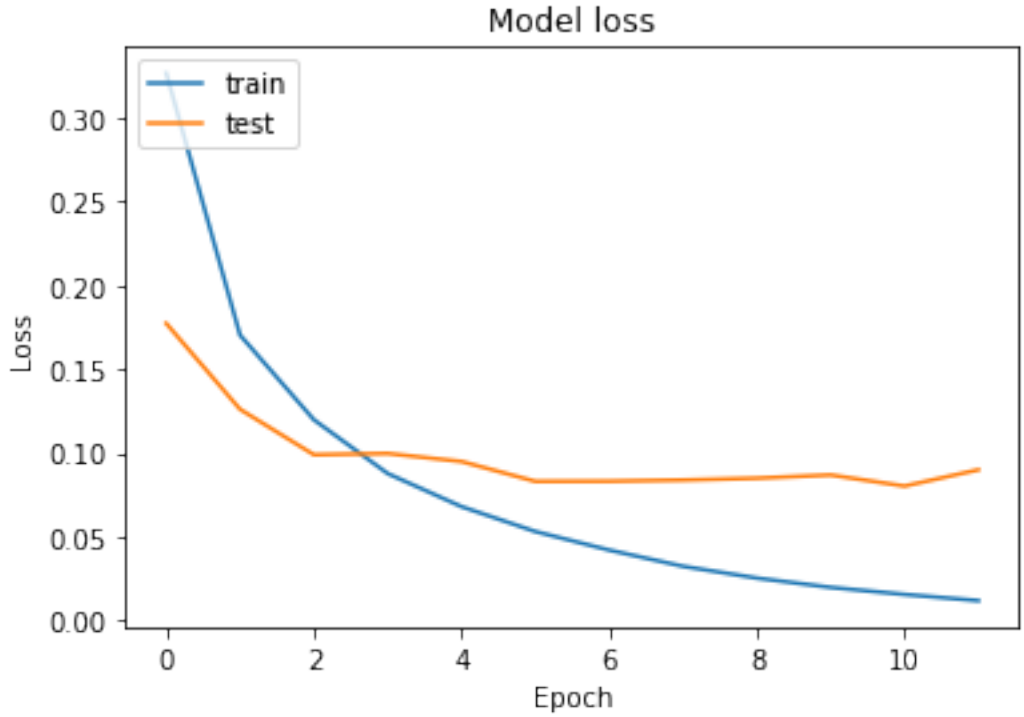
0.1.3 Tanh Activation

```
In [10]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12,validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 4s 69us/sample - loss: 0.3263 - acc: 0.9054 - va
Epoch 2/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.1702 - acc: 0.9484 - va
Epoch 3/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.1198 - acc: 0.9640 - va
Epoch 4/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.0878 - acc: 0.9734 - va
Epoch 5/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.0680 - acc: 0.9787 - va
Epoch 6/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.0532 - acc: 0.9831 - va
Epoch 7/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.0422 - acc: 0.9869 - va
Epoch 8/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.0325 - acc: 0.9901 - va
Epoch 9/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.0255 - acc: 0.9925 - va
Epoch 10/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.0199 - acc: 0.9940 - va
Epoch 11/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.0157 - acc: 0.9953 - va
Epoch 12/12
54000/54000 [==============================] - 3s 64us/sample - loss: 0.0120 - acc: 0.9966 - va
```

```
In [11]: draw_learning_curve(model_graph)
```


Model Accuracy
train
test
Accuracy
Epoch
1.00
0.98
0.96
0.94
0.92
0
2
4
6
8
10


Model loss
train
test
Loss
Epoch
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
2
4
6
8
10

```
In [12]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

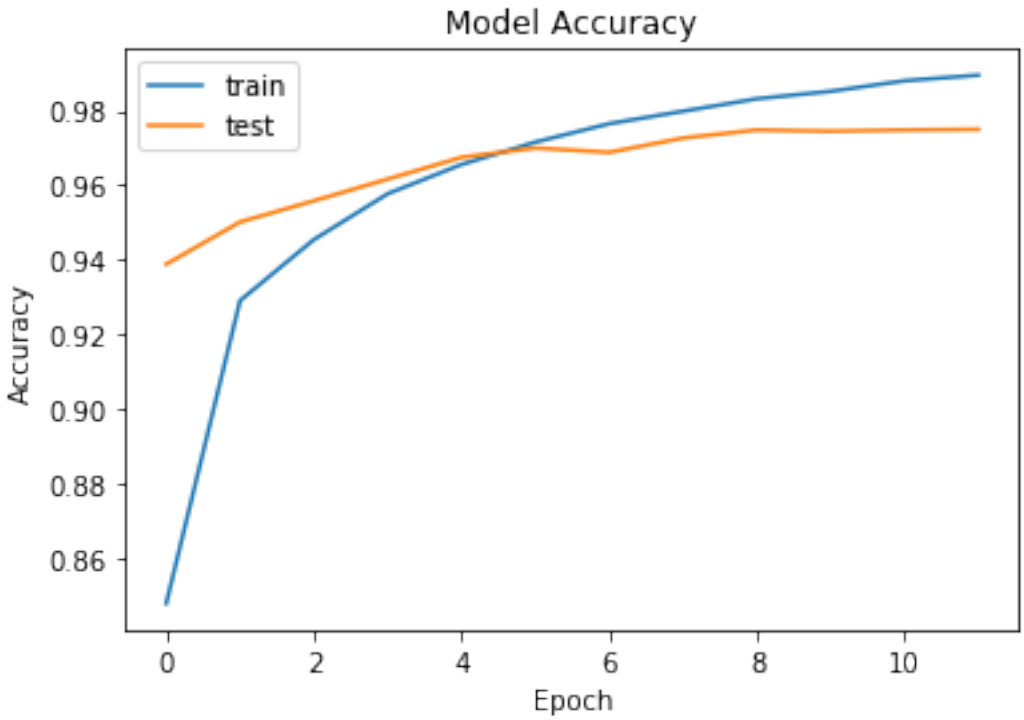
```
60000/60000 [==============================] - 2s 33us/sample - loss: 0.0195 - acc: 0.9942
Train accuracy: 0.9941667
10000/10000 [==============================] - 0s 33us/sample - loss: 0.1038 - acc: 0.9726
Test accuracy: 0.9726
```

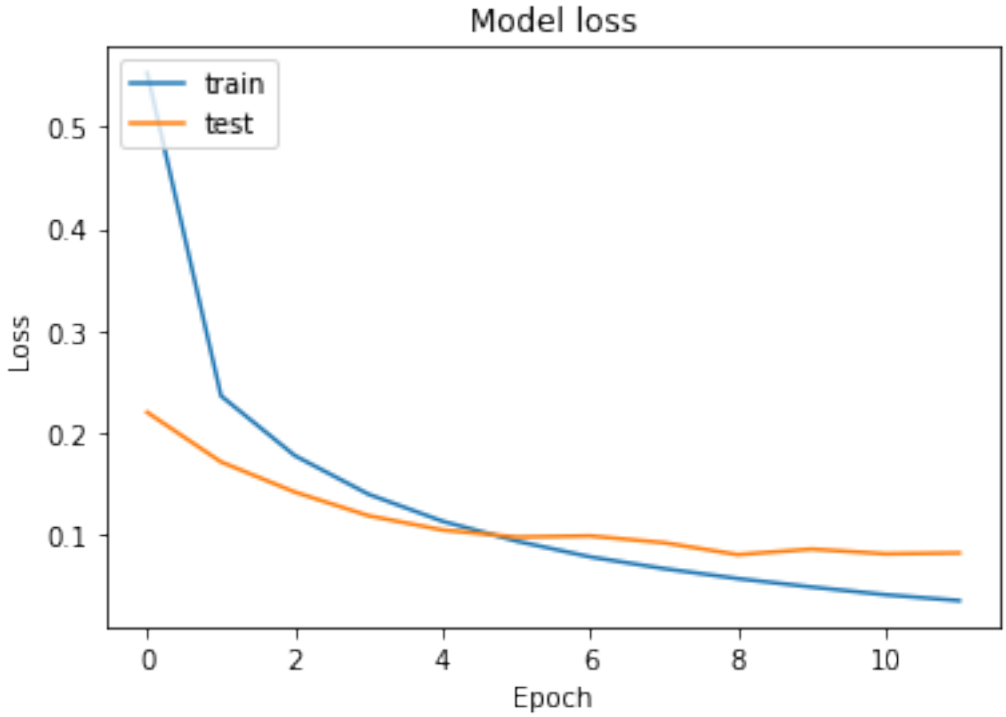
0.1.4 Sigmoid Activation

```
In [13]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 4s 71us/sample - loss: 0.5523 - acc: 0.8479 - va
Epoch 2/12
54000/54000 [==============================] - 4s 65us/sample - loss: 0.2362 - acc: 0.9291 - va
Epoch 3/12
54000/54000 [==============================] - 3s 65us/sample - loss: 0.1777 - acc: 0.9454 - va
Epoch 4/12
54000/54000 [==============================] - 3s 65us/sample - loss: 0.1398 - acc: 0.9577 - va
Epoch 5/12
54000/54000 [==============================] - 3s 65us/sample - loss: 0.1134 - acc: 0.9656 - va
Epoch 6/12
54000/54000 [==============================] - 3s 65us/sample - loss: 0.0940 - acc: 0.9716 - va
Epoch 7/12
54000/54000 [==============================] - 3s 65us/sample - loss: 0.0785 - acc: 0.9765 - va
Epoch 8/12
54000/54000 [==============================] - 3s 65us/sample - loss: 0.0670 - acc: 0.9799 - va
Epoch 9/12
54000/54000 [==============================] - 3s 65us/sample - loss: 0.0573 - acc: 0.9832 - va
Epoch 10/12
54000/54000 [==============================] - 4s 66us/sample - loss: 0.0490 - acc: 0.9852 - va
Epoch 11/12
54000/54000 [==============================] - 3s 65us/sample - loss: 0.0414 - acc: 0.9880 - va
Epoch 12/12
54000/54000 [==============================] - 3s 65us/sample - loss: 0.0357 - acc: 0.9895 - va
```

```
In [14]: draw_learning_curve(model_graph)
```


Model Accuracy
train
test
Accuracy
0.98
0.96
0.94
0.92
0.90
0.88
0.86
0
2
4
6
8
10
Epoch


Model loss
train
test
Loss
0.5
0.4
0.3
0.2
0.1
0
2
4
6
8
10
Epoch

```
In [15]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 2s 33us/sample - loss: 0.0334 - acc: 0.9910
Train accuracy: 0.9909667
10000/10000 [==============================] - 0s 34us/sample - loss: 0.0879 - acc: 0.9738
Test accuracy: 0.9738
```

0.2 Dropout

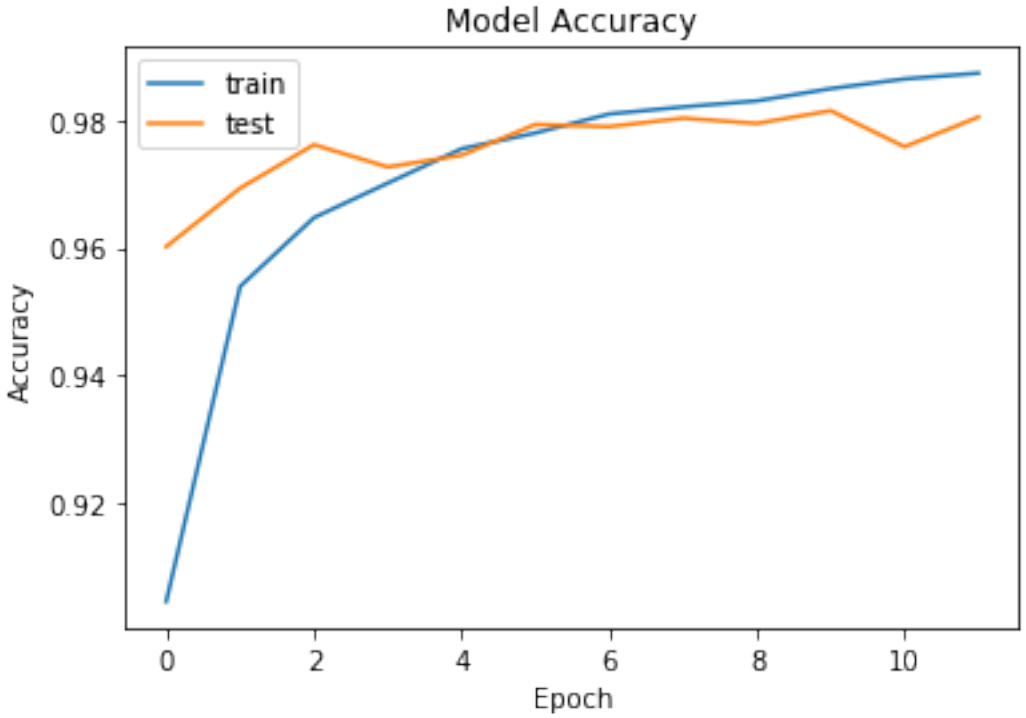
0.2.1 ReLu Activation

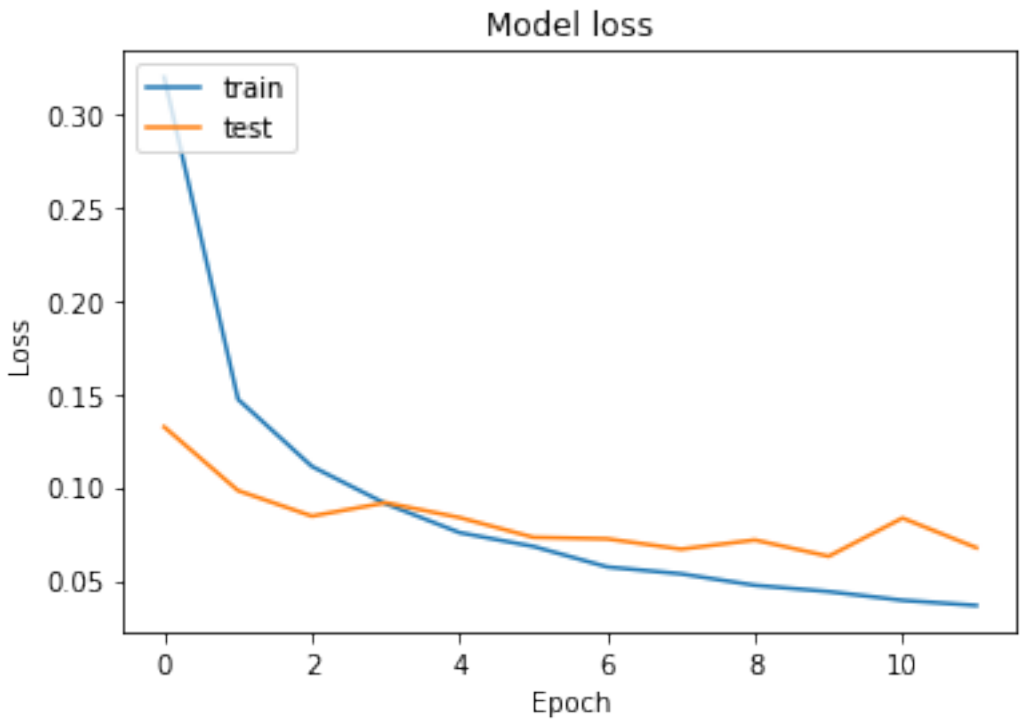
```
In [16]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dropout(0.2))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12,validation_split=0.1)
```

```
WARNING:tensorflow:From C:\Users\Miguel\Anaconda3\lib\site-packages\tensorflow\python\keras\lay
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.3197 - acc: 0.9045 - va
Epoch 2/12
54000/54000 [==============================] - 4s 67us/sample - loss: 0.1471 - acc: 0.9539 - va
Epoch 3/12
54000/54000 [==============================] - 4s 67us/sample - loss: 0.1111 - acc: 0.9647 - va
Epoch 4/12
54000/54000 [==============================] - 4s 67us/sample - loss: 0.0911 - acc: 0.9701 - va
Epoch 5/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0756 - acc: 0.9755 - va
Epoch 6/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0683 - acc: 0.9780 - va
Epoch 7/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0573 - acc: 0.9810 - va
Epoch 8/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0536 - acc: 0.9821 - va
Epoch 9/12
54000/54000 [==============================] - 4s 67us/sample - loss: 0.0474 - acc: 0.9830 - va
Epoch 10/12
54000/54000 [==============================] - 4s 67us/sample - loss: 0.0441 - acc: 0.9850 - va
Epoch 11/12
```

```
54000/54000 [==============================] - 4s 67us/sample - loss: 0.0394 - acc: 0.9865 - va
Epoch 12/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0366 - acc: 0.9874 - va
```

```
In [17]: draw_learning_curve(model_graph)
```






```
In [18]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 2s 37us/sample - loss: 0.0190 - acc: 0.9943
Train accuracy: 0.9943333
10000/10000 [==============================] - 0s 36us/sample - loss: 0.0729 - acc: 0.9787
Test accuracy: 0.9787
```

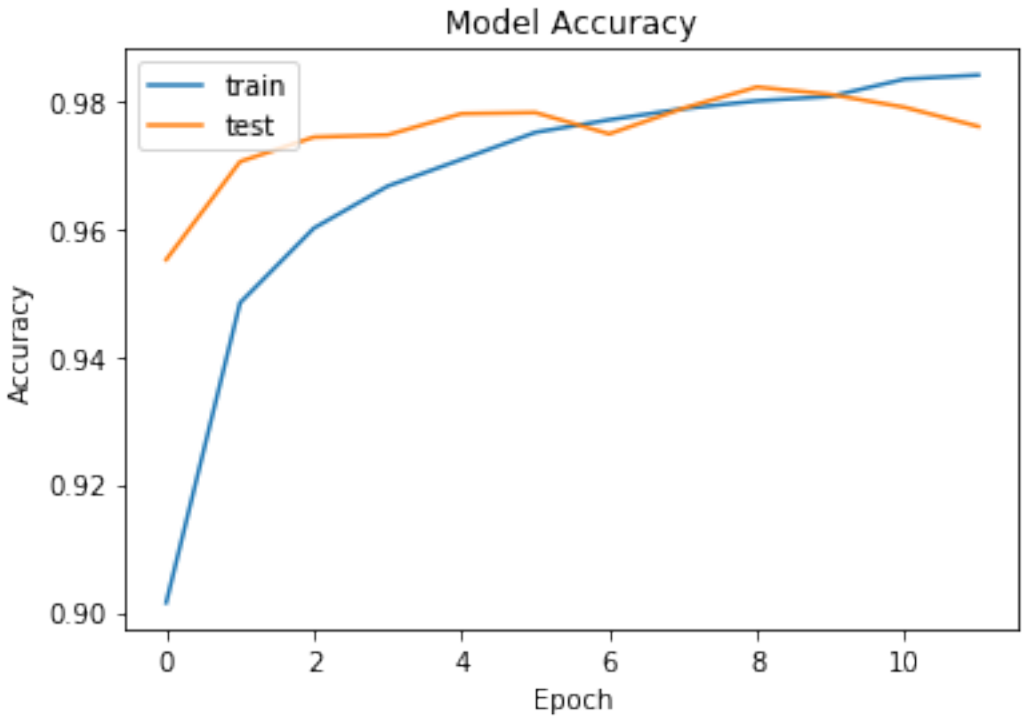
0.2.2 Leaky ReLu Activation

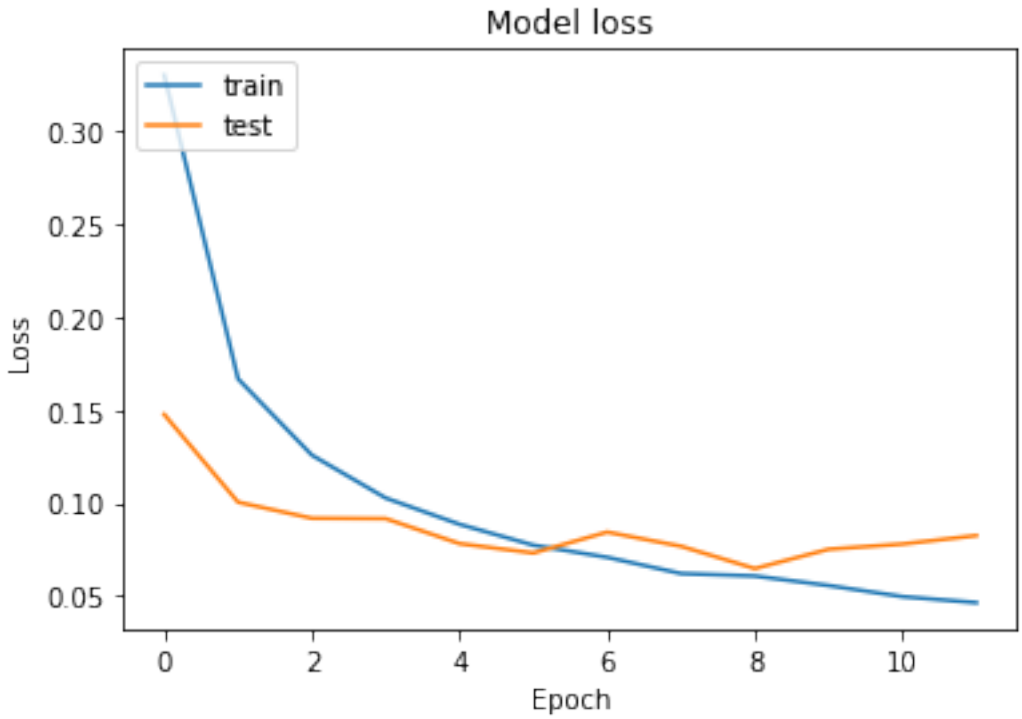
```
In [19]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dropout(0.2))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))

         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
```

```
54000/54000 [==============================] - 4s 75us/sample - loss: 0.3301 - acc: 0.9015 - va
Epoch 2/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.1667 - acc: 0.9484 - va
Epoch 3/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.1258 - acc: 0.9601 - va
Epoch 4/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.1027 - acc: 0.9667 - va
Epoch 5/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0886 - acc: 0.9709 - va
Epoch 6/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0773 - acc: 0.9751 - va
Epoch 7/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0708 - acc: 0.9770 - va
Epoch 8/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0621 - acc: 0.9788 - va
Epoch 9/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0608 - acc: 0.9800 - va
Epoch 10/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0556 - acc: 0.9807 - va
Epoch 11/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0497 - acc: 0.9834 - va
Epoch 12/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0464 - acc: 0.9840 - va
```

In [20]: `draw_learning_curve(model_graph)`






```
In [21]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

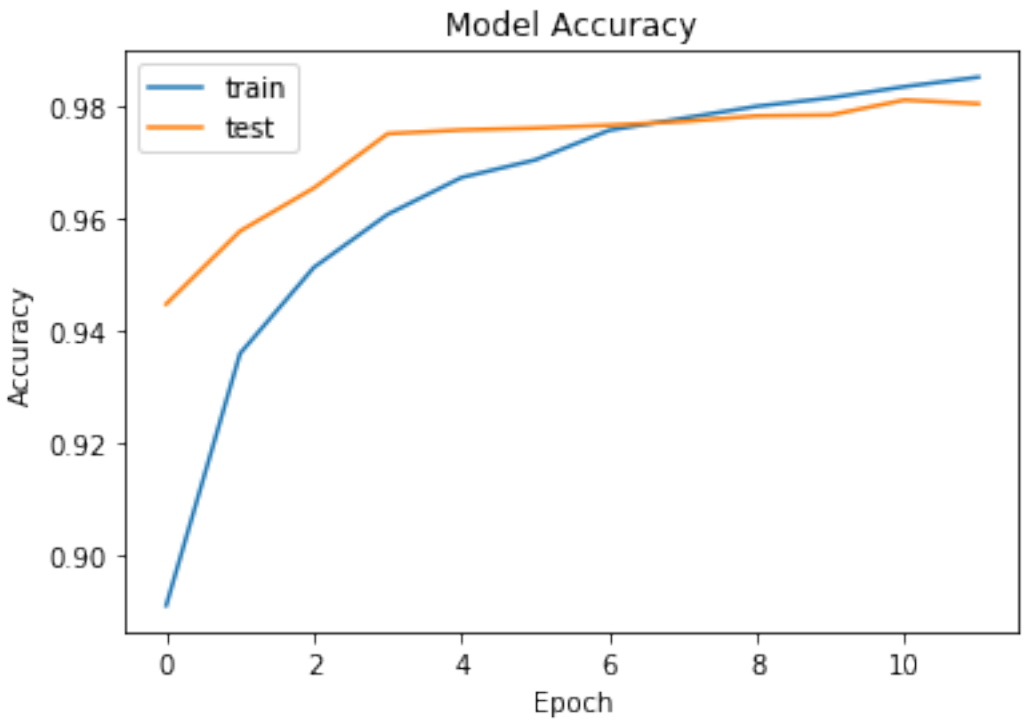
```
60000/60000 [==============================] - 2s 36us/sample - loss: 0.0327 - acc: 0.9895
Train accuracy: 0.9895
10000/10000 [==============================] - 0s 36us/sample - loss: 0.0917 - acc: 0.9745
Test accuracy: 0.9745
```

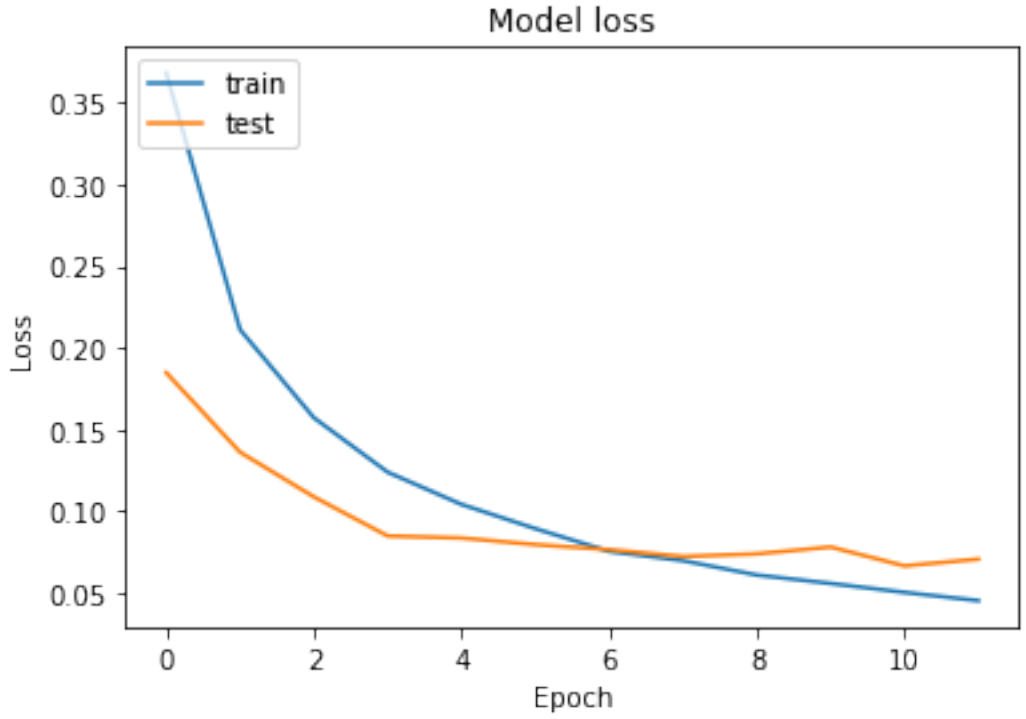
0.2.3 Tanh Activation

```
In [22]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dropout(0.2))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))

         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 4s 74us/sample - loss: 0.3680 - acc: 0.8910 - va
Epoch 2/12
54000/54000 [==============================] - 4s 70us/sample - loss: 0.2111 - acc: 0.9359 - va
Epoch 3/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.1572 - acc: 0.9512 - va
Epoch 4/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.1240 - acc: 0.9606 - va
Epoch 5/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.1043 - acc: 0.9672 - va
Epoch 6/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0895 - acc: 0.9703 - va
Epoch 7/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0755 - acc: 0.9756 - va
Epoch 8/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.0699 - acc: 0.9777 - va
Epoch 9/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.0611 - acc: 0.9799 - va
Epoch 10/12
54000/54000 [==============================] - 4s 71us/sample - loss: 0.0559 - acc: 0.9814 - va
Epoch 11/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0505 - acc: 0.9834 - va
Epoch 12/12
54000/54000 [==============================] - 4s 70us/sample - loss: 0.0454 - acc: 0.9851 - va
```

```
In [23]: draw_learning_curve(model_graph)
```


Model Accuracy
train
test
Accuracy
0.98
0.96
0.94
0.92
0.90
0
2
4
6
8
10
Epoch


Model loss
train
test
Loss
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0
2
4
6
8
10
Epoch

```
In [24]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

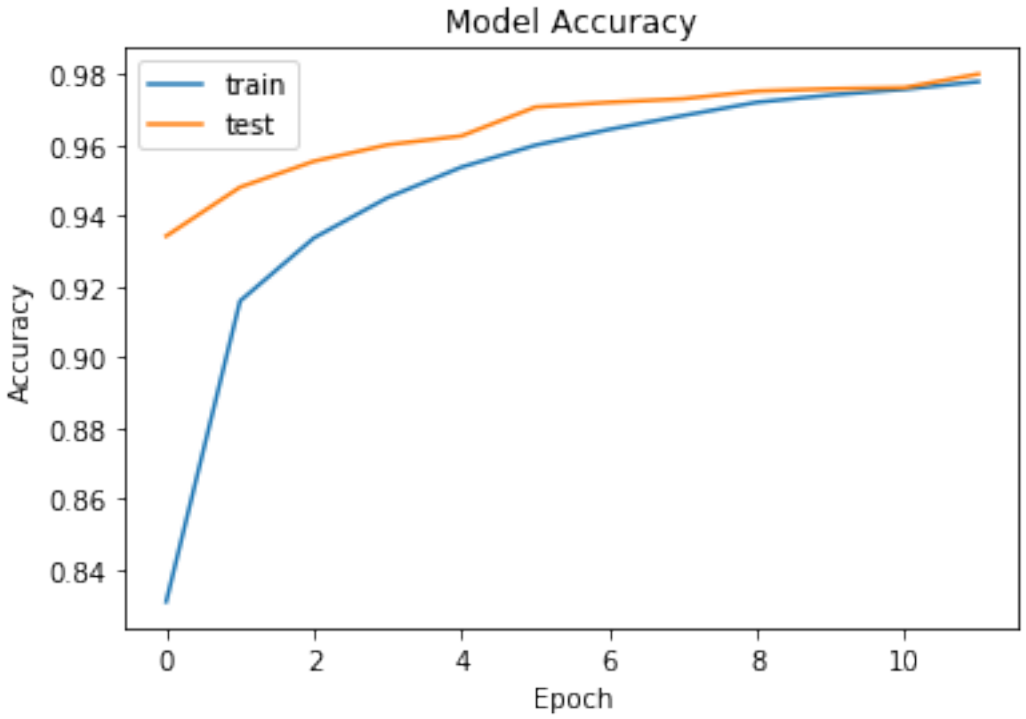
```
60000/60000 [==============================] - 2s 38us/sample - loss: 0.0257 - acc: 0.9929
Train accuracy: 0.9929
10000/10000 [==============================] - 0s 37us/sample - loss: 0.0764 - acc: 0.9780
Test accuracy: 0.978
```

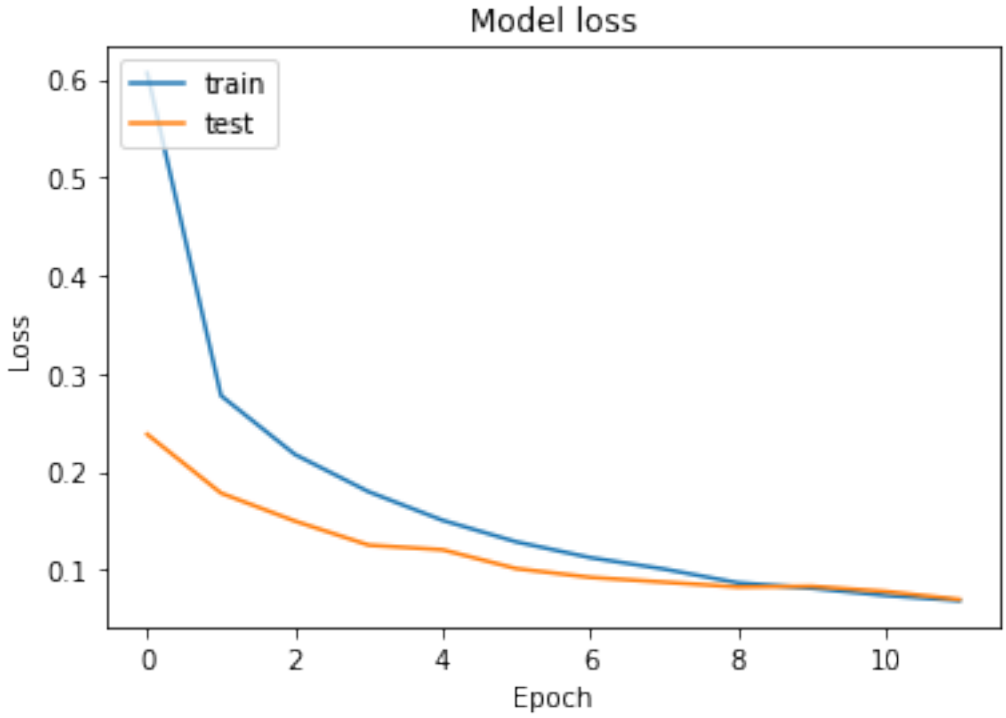
0.2.4 Sigmoid Activation

```
In [25]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dropout(0.2))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 4s 74us/sample - loss: 0.6062 - acc: 0.8309 - va
Epoch 2/12
54000/54000 [==============================] - 4s 69us/sample - loss: 0.2776 - acc: 0.9160 - va
Epoch 3/12
54000/54000 [==============================] - 4s 69us/sample - loss: 0.2178 - acc: 0.9338 - va
Epoch 4/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.1797 - acc: 0.9452 - va
Epoch 5/12
54000/54000 [==============================] - 4s 69us/sample - loss: 0.1506 - acc: 0.9539 - va
Epoch 6/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.1287 - acc: 0.9600 - va
Epoch 7/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.1126 - acc: 0.9645 - va
Epoch 8/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.1010 - acc: 0.9684 - va
Epoch 9/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0870 - acc: 0.9722 - va
Epoch 10/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0814 - acc: 0.9742 - va
Epoch 11/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0741 - acc: 0.9759 - va
Epoch 12/12
54000/54000 [==============================] - 4s 68us/sample - loss: 0.0688 - acc: 0.9780 - va
```

In [26]: draw_learning_curve(model_graph)





```
In [27]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 2s 37us/sample - loss: 0.0417 - acc: 0.9879
Train accuracy: 0.98793334
10000/10000 [==============================] - 0s 37us/sample - loss: 0.0827 - acc: 0.9745
Test accuracy: 0.9745
```

0.3 Batch Normalization

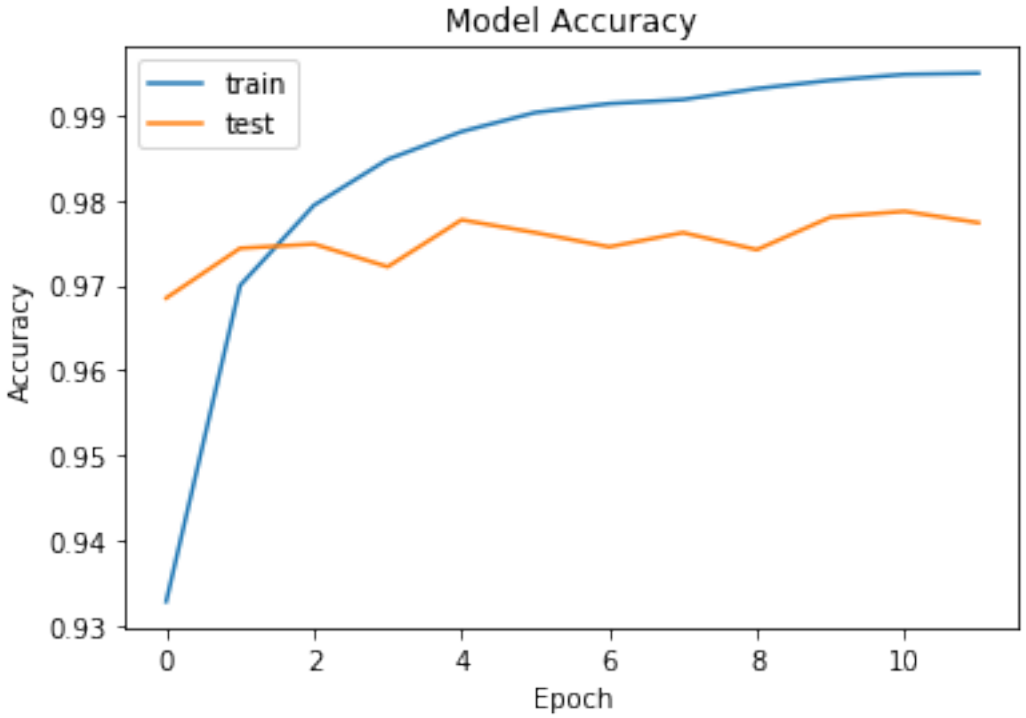
0.3.1 ReLu Activation

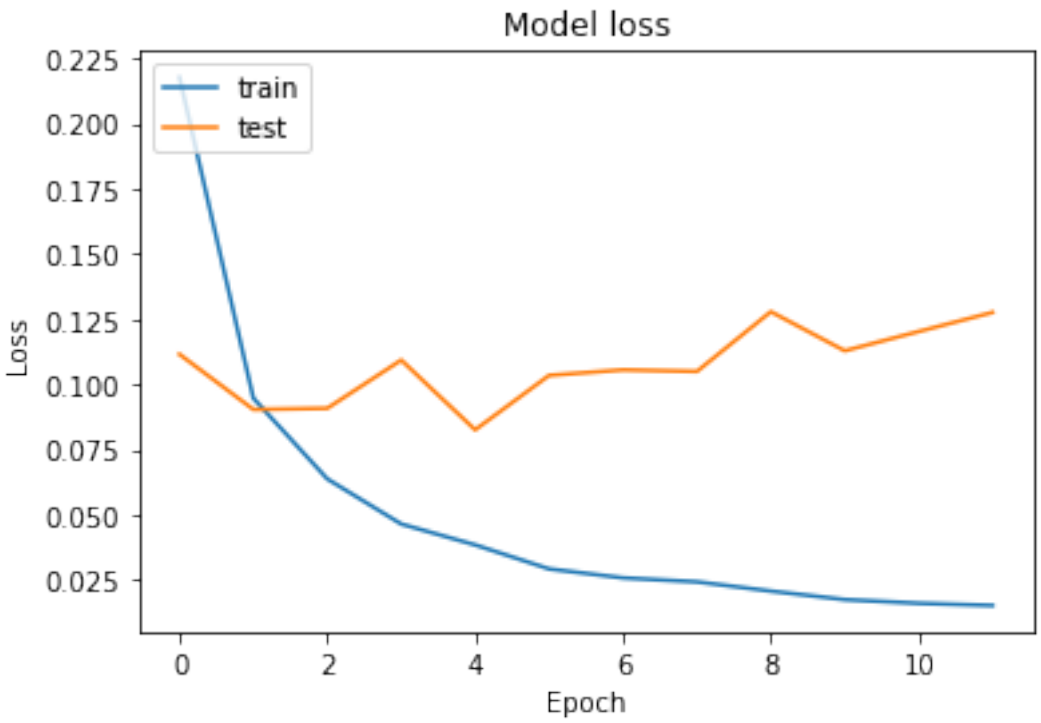
```
In [28]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph= model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 5s 87us/sample - loss: 0.2178 - acc: 0.9329 - va
Epoch 2/12
54000/54000 [==============================] - 4s 80us/sample - loss: 0.0946 - acc: 0.9700 - va
Epoch 3/12
54000/54000 [==============================] - 4s 80us/sample - loss: 0.0637 - acc: 0.9794 - va
Epoch 4/12
54000/54000 [==============================] - 4s 80us/sample - loss: 0.0464 - acc: 0.9848 - va
Epoch 5/12
54000/54000 [==============================] - 4s 80us/sample - loss: 0.0384 - acc: 0.9880 - va
Epoch 6/12
54000/54000 [==============================] - 4s 82us/sample - loss: 0.0292 - acc: 0.9903 - va
Epoch 7/12
54000/54000 [==============================] - 4s 80us/sample - loss: 0.0257 - acc: 0.9913 - va
Epoch 8/12
54000/54000 [==============================] - 4s 80us/sample - loss: 0.0242 - acc: 0.9918 - va
Epoch 9/12
54000/54000 [==============================] - 4s 80us/sample - loss: 0.0207 - acc: 0.9931 - va
Epoch 10/12
54000/54000 [==============================] - 4s 80us/sample - loss: 0.0174 - acc: 0.9940 - va
Epoch 11/12
```

```
54000/54000 [==============================] - 4s 80us/sample - loss: 0.0160 - acc: 0.9947 - va
Epoch 12/12
54000/54000 [==============================] - 4s 80us/sample - loss: 0.0152 - acc: 0.9949 - va
```

```
In [29]: draw_learning_curve(model_graph)
```






```
In [30]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 2s 40us/sample - loss: 0.0229 - acc: 0.9946
Train accuracy: 0.9945667
10000/10000 [==============================] - 0s 40us/sample - loss: 0.1253 - acc: 0.9746
Test accuracy: 0.9746
```

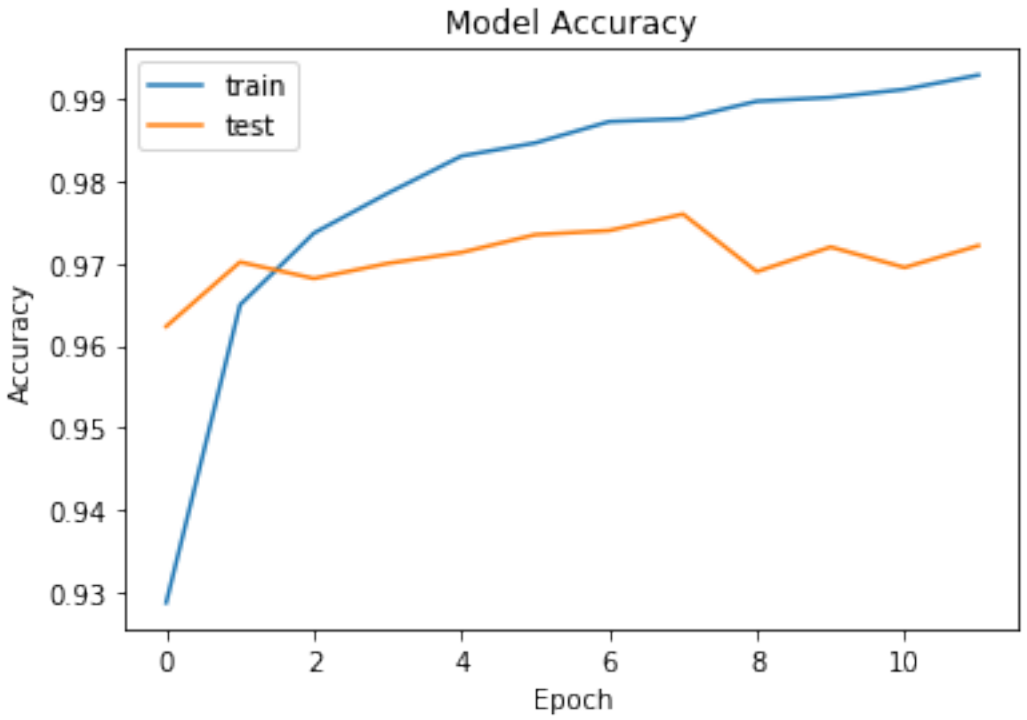
0.3.2 Leaky ReLu Activation

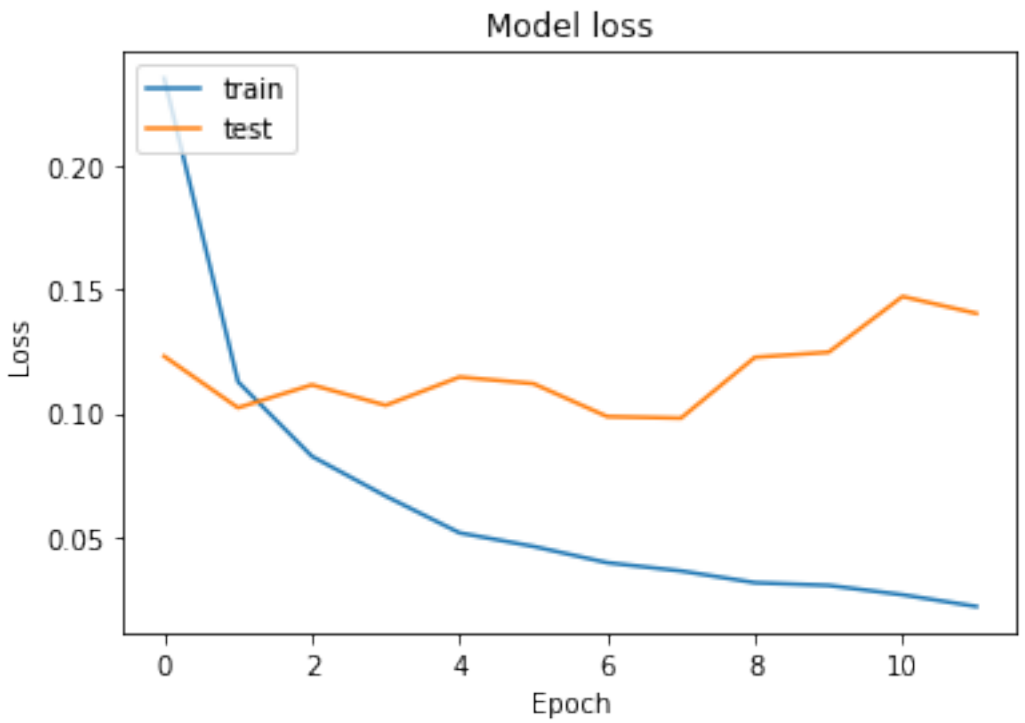
```
In [31]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
```

```
54000/54000 [==============================] - 5s 93us/sample - loss: 0.2357 - acc: 0.9287 - va
Epoch 2/12
54000/54000 [==============================] - 5s 85us/sample - loss: 0.1128 - acc: 0.9649 - va
Epoch 3/12
54000/54000 [==============================] - 4s 81us/sample - loss: 0.0826 - acc: 0.9737 - va
Epoch 4/12
54000/54000 [==============================] - 4s 82us/sample - loss: 0.0665 - acc: 0.9785 - va
Epoch 5/12
54000/54000 [==============================] - 4s 82us/sample - loss: 0.0516 - acc: 0.9831 - va
Epoch 6/12
54000/54000 [==============================] - 4s 81us/sample - loss: 0.0461 - acc: 0.9847 - va
Epoch 7/12
54000/54000 [==============================] - 4s 82us/sample - loss: 0.0395 - acc: 0.9873 - va
Epoch 8/12
54000/54000 [==============================] - 4s 82us/sample - loss: 0.0361 - acc: 0.9876 - va
Epoch 9/12
54000/54000 [==============================] - 4s 82us/sample - loss: 0.0314 - acc: 0.9897 - va
Epoch 10/12
54000/54000 [==============================] - 4s 82us/sample - loss: 0.0303 - acc: 0.9902 - va
Epoch 11/12
54000/54000 [==============================] - 4s 81us/sample - loss: 0.0265 - acc: 0.9912 - va
Epoch 12/12
54000/54000 [==============================] - 4s 82us/sample - loss: 0.0218 - acc: 0.9929 - va
```

In [32]: `draw_learning_curve(model_graph)`






```
In [33]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

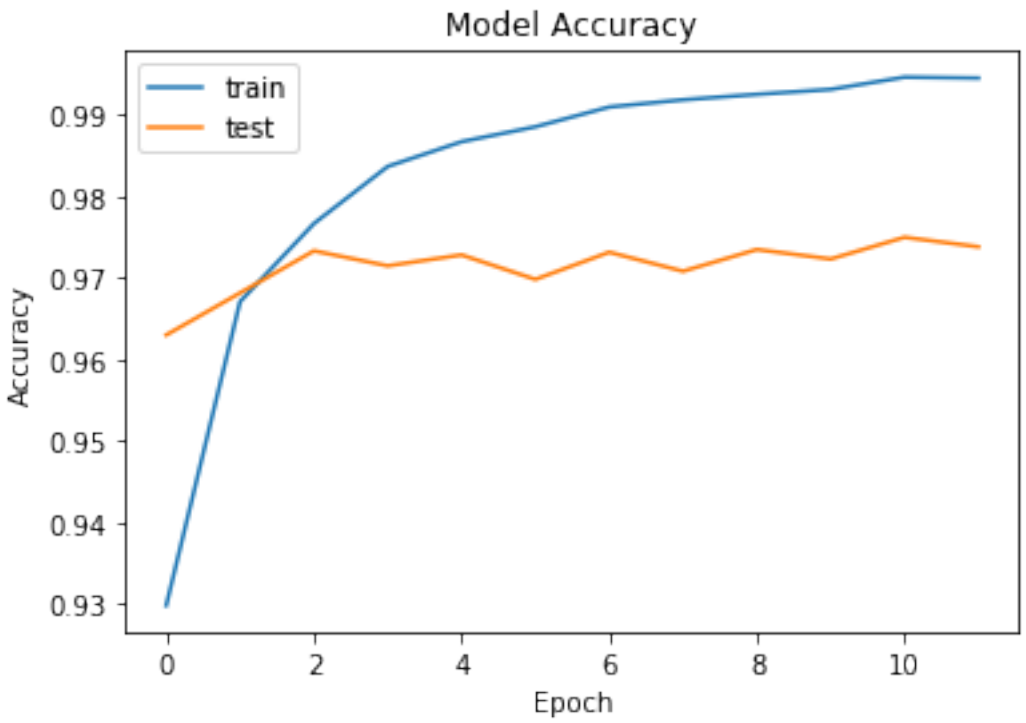
```
60000/60000 [==============================] - 2s 40us/sample - loss: 0.0427 - acc: 0.9881
Train accuracy: 0.9881167
10000/10000 [==============================] - 0s 40us/sample - loss: 0.1645 - acc: 0.9663
Test accuracy: 0.9663
```

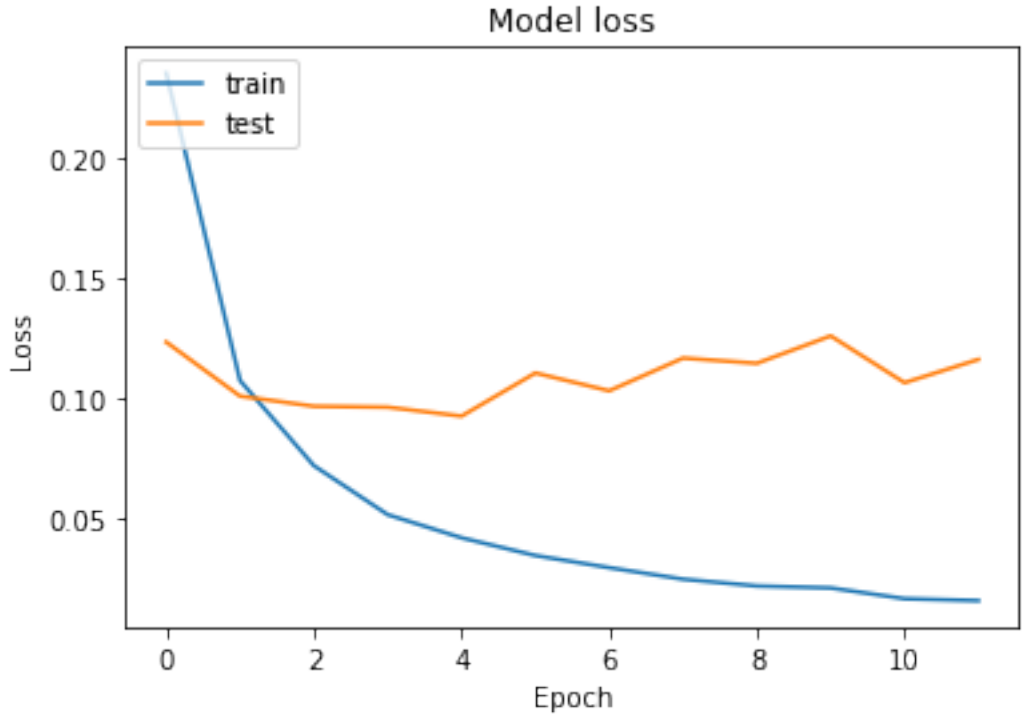
0.3.3 Tanh Activation

```
In [34]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 5s 89us/sample - loss: 0.2354 - acc: 0.9298 - va
Epoch 2/12
54000/54000 [==============================] - 5s 83us/sample - loss: 0.1071 - acc: 0.9671 - va
Epoch 3/12
54000/54000 [==============================] - 5s 83us/sample - loss: 0.0715 - acc: 0.9767 - va
Epoch 4/12
54000/54000 [==============================] - 4s 83us/sample - loss: 0.0511 - acc: 0.9837 - va
Epoch 5/12
54000/54000 [==============================] - 5s 83us/sample - loss: 0.0414 - acc: 0.9867 - va
Epoch 6/12
54000/54000 [==============================] - 5s 84us/sample - loss: 0.0340 - acc: 0.9886 - va
Epoch 7/12
54000/54000 [==============================] - 4s 83us/sample - loss: 0.0290 - acc: 0.9910 - va
Epoch 8/12
54000/54000 [==============================] - 4s 83us/sample - loss: 0.0241 - acc: 0.9919 - va
Epoch 9/12
54000/54000 [==============================] - 4s 83us/sample - loss: 0.0213 - acc: 0.9926 - va
Epoch 10/12
54000/54000 [==============================] - 4s 83us/sample - loss: 0.0205 - acc: 0.9932 - va
Epoch 11/12
54000/54000 [==============================] - 4s 82us/sample - loss: 0.0160 - acc: 0.9947 - va
Epoch 12/12
54000/54000 [==============================] - 4s 82us/sample - loss: 0.0152 - acc: 0.9946 - va
```

```
In [35]: draw_learning_curve(model_graph)
```


Model Accuracy
train
test
Accuracy
0.99
0.98
0.97
0.96
0.95
0.94
0.93
0
2
4
6
8
10
Epoch


Model loss
train
test
Loss
0.20
0.15
0.10
0.05
0
2
4
6
8
10
Epoch

```
In [36]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

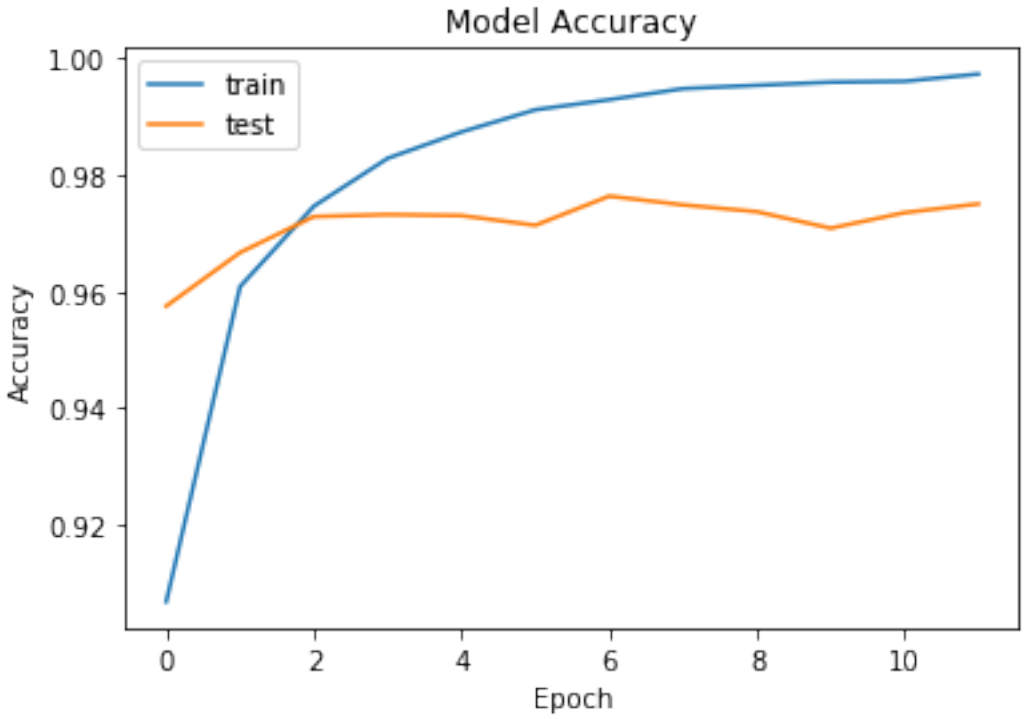
```
60000/60000 [==============================] - 2s 41us/sample - loss: 0.0211 - acc: 0.9944
Train accuracy: 0.99435
10000/10000 [==============================] - 0s 41us/sample - loss: 0.1312 - acc: 0.9694
Test accuracy: 0.9694
```

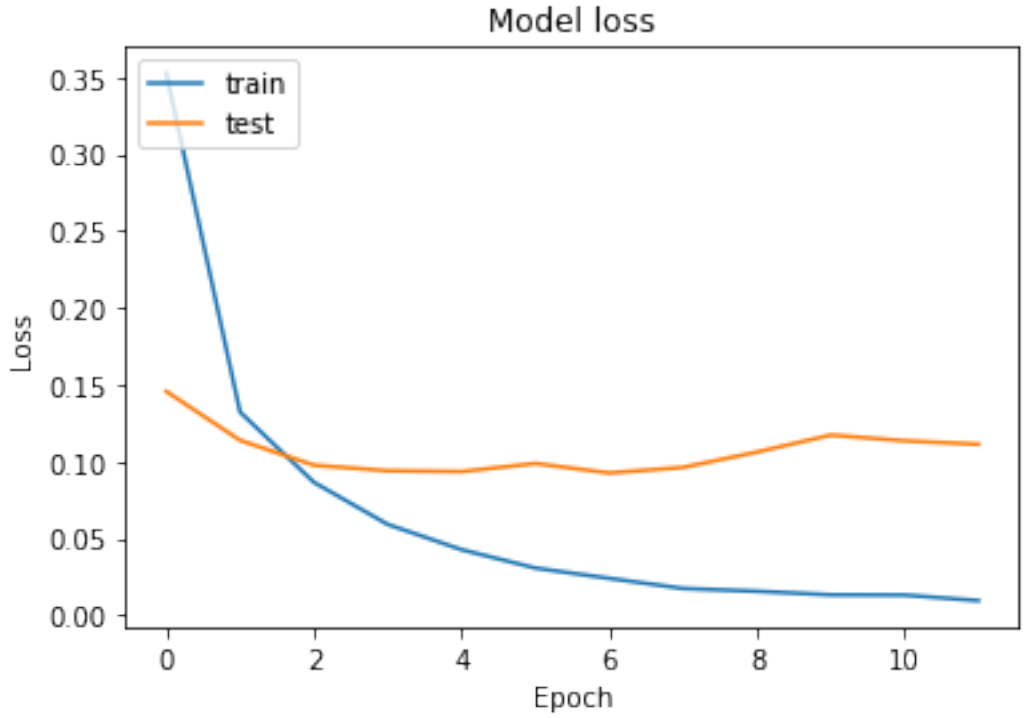
0.3.4 Sigmoid Activation

```
In [37]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 5s 92us/sample - loss: 0.3526 - acc: 0.9069 - va
Epoch 2/12
54000/54000 [==============================] - 5s 84us/sample - loss: 0.1322 - acc: 0.9608 - va
Epoch 3/12
54000/54000 [==============================] - 5s 85us/sample - loss: 0.0866 - acc: 0.9746 - va
Epoch 4/12
54000/54000 [==============================] - 5s 84us/sample - loss: 0.0592 - acc: 0.9828 - va
Epoch 5/12
54000/54000 [==============================] - 4s 83us/sample - loss: 0.0428 - acc: 0.9873 - va
Epoch 6/12
54000/54000 [==============================] - 5s 85us/sample - loss: 0.0308 - acc: 0.9911 - va
Epoch 7/12
54000/54000 [==============================] - 5s 84us/sample - loss: 0.0241 - acc: 0.9929 - va
Epoch 8/12
54000/54000 [==============================] - 5s 84us/sample - loss: 0.0175 - acc: 0.9947 - va
Epoch 9/12
54000/54000 [==============================] - 5s 84us/sample - loss: 0.0157 - acc: 0.9953 - va
Epoch 10/12
54000/54000 [==============================] - 5s 84us/sample - loss: 0.0134 - acc: 0.9959 - va
Epoch 11/12
54000/54000 [==============================] - 5s 84us/sample - loss: 0.0132 - acc: 0.9960 - va
Epoch 12/12
54000/54000 [==============================] - 5s 84us/sample - loss: 0.0096 - acc: 0.9972 - va
```

In [38]: draw_learning_curve(model_graph)





```
In [39]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 3s 42us/sample - loss: 0.0172 - acc: 0.9955
Train accuracy: 0.99555
10000/10000 [==============================] - 0s 42us/sample - loss: 0.1249 - acc: 0.9714
Test accuracy: 0.9714
```

0.4 L1 Regularization

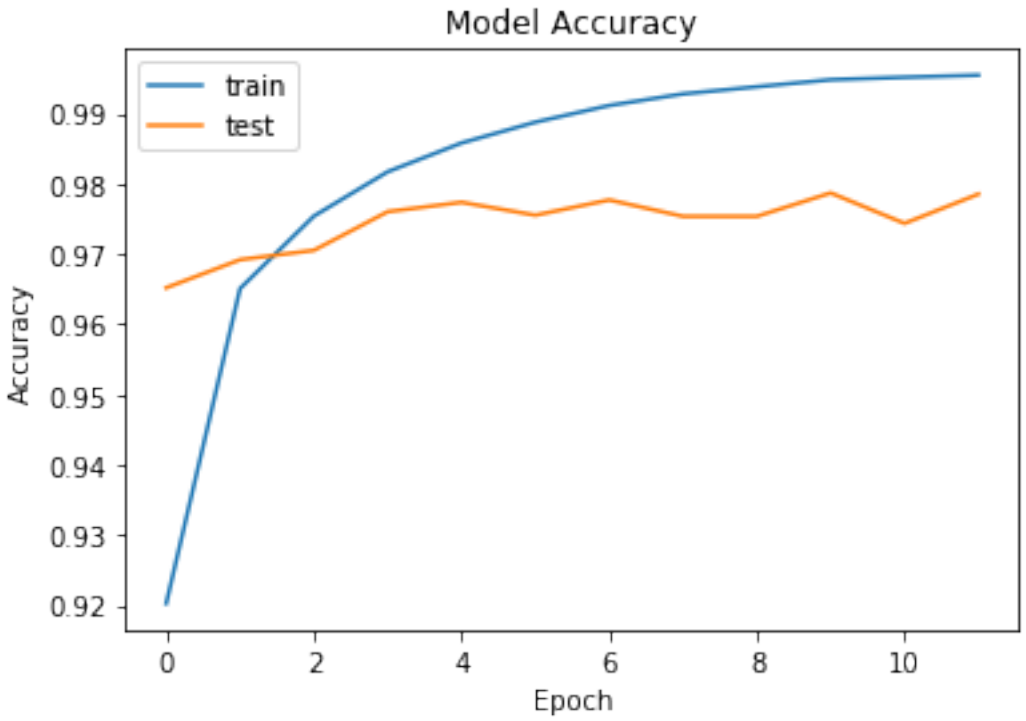
0.4.1 ReLu Activation

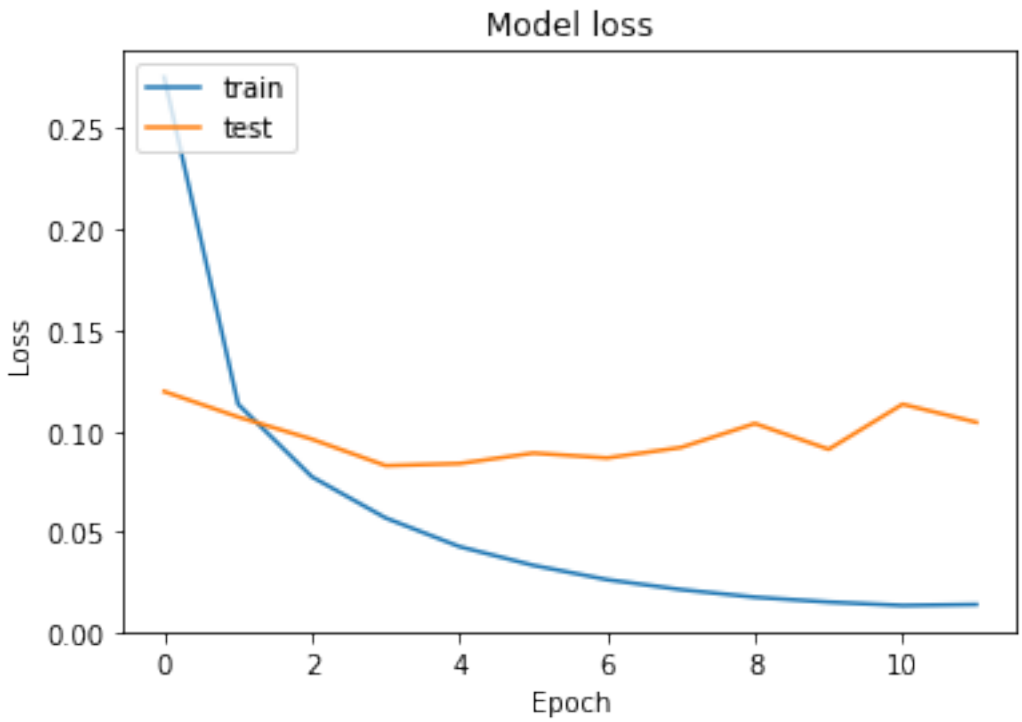
```
In [40]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l1=0.01))
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))

         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 4s 82us/sample - loss: 0.2754 - acc: 0.9202 - va
Epoch 2/12
54000/54000 [==============================] - 4s 74us/sample - loss: 0.1134 - acc: 0.9651 - va
Epoch 3/12
54000/54000 [==============================] - 4s 74us/sample - loss: 0.0774 - acc: 0.9754 - va
Epoch 4/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.0570 - acc: 0.9817 - va
Epoch 5/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.0428 - acc: 0.9858 - va
Epoch 6/12
54000/54000 [==============================] - 4s 72us/sample - loss: 0.0335 - acc: 0.9887 - va
Epoch 7/12
54000/54000 [==============================] - 4s 72us/sample - loss: 0.0265 - acc: 0.9911 - va
Epoch 8/12
54000/54000 [==============================] - 4s 72us/sample - loss: 0.0215 - acc: 0.9928 - va
Epoch 9/12
54000/54000 [==============================] - 4s 72us/sample - loss: 0.0178 - acc: 0.9938 - va
Epoch 10/12
54000/54000 [==============================] - 4s 72us/sample - loss: 0.0154 - acc: 0.9948 - va
```

```
Epoch 11/12
54000/54000 [==============================] - 4s 72us/sample - loss: 0.0137 - acc: 0.9951 - va
Epoch 12/12
54000/54000 [==============================] - 4s 72us/sample - loss: 0.0142 - acc: 0.9954 - va
```

```
In [41]: draw_learning_curve(model_graph)
```






```
In [42]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 2s 38us/sample - loss: 0.0196 - acc: 0.9945
Train accuracy: 0.9945167
10000/10000 [==============================] - 0s 39us/sample - loss: 0.1229 - acc: 0.9721
Test accuracy: 0.9721
```

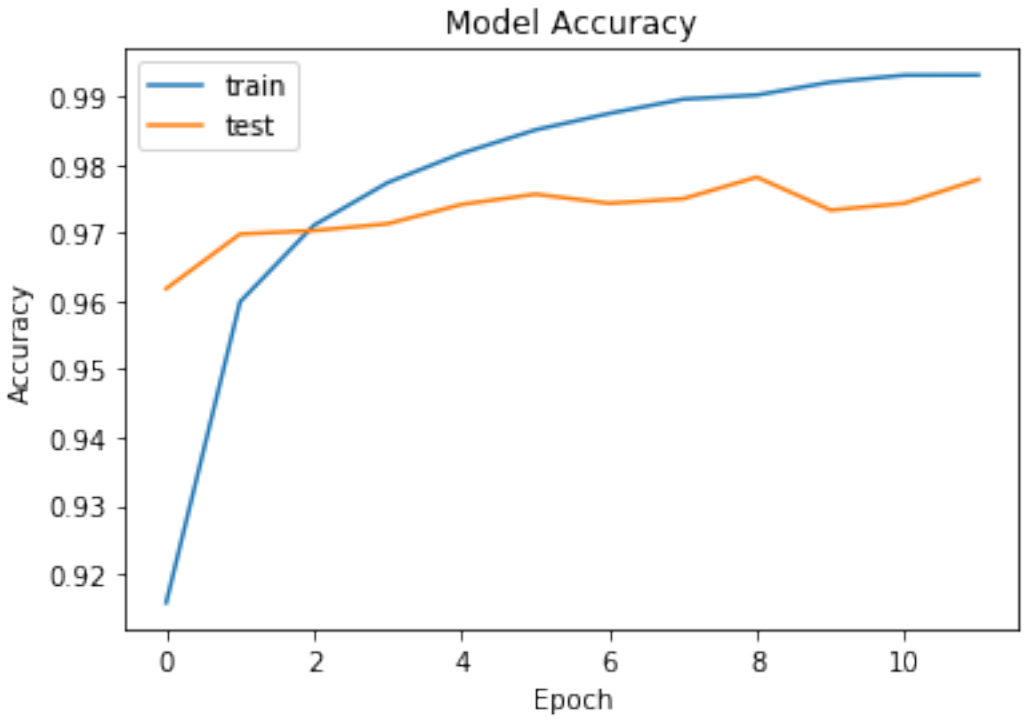
0.4.2 Leaky ReLu Activation

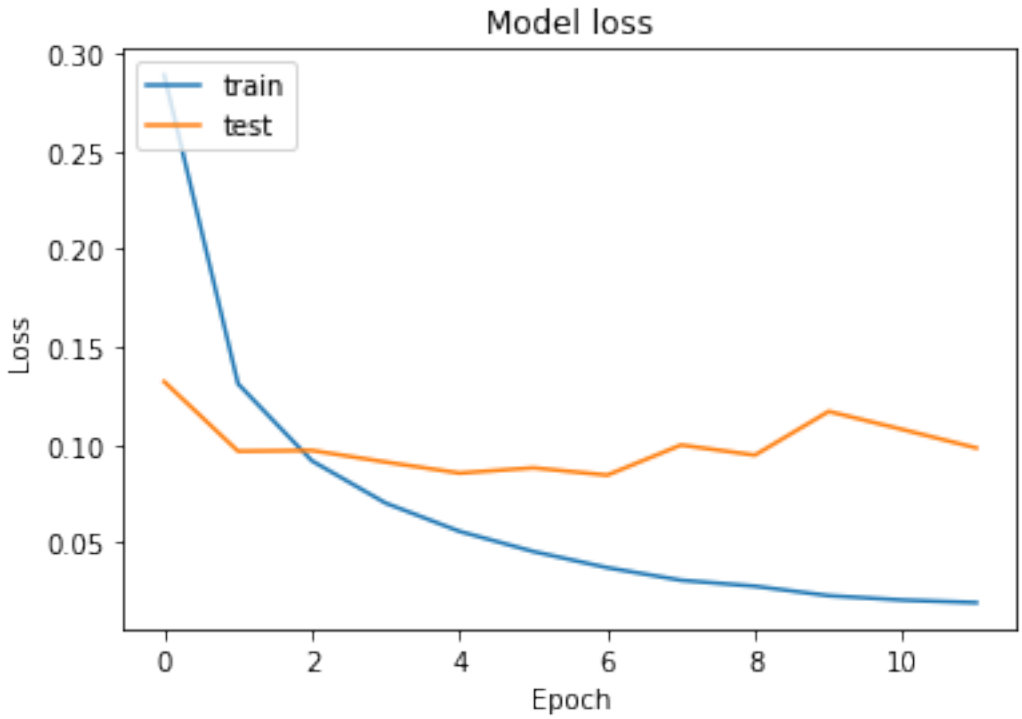
```
In [43]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l1=0.01))
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))

         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
```

```
54000/54000 [==============================] - 4s 81us/sample - loss: 0.2887 - acc: 0.9158 - va
Epoch 2/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.1311 - acc: 0.9599 - va
Epoch 3/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.0919 - acc: 0.9711 - va
Epoch 4/12
54000/54000 [==============================] - 4s 72us/sample - loss: 0.0703 - acc: 0.9774 - va
Epoch 5/12
54000/54000 [==============================] - 4s 72us/sample - loss: 0.0558 - acc: 0.9817 - va
Epoch 6/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.0456 - acc: 0.9851 - va
Epoch 7/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.0373 - acc: 0.9875 - va
Epoch 8/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.0309 - acc: 0.9896 - va
Epoch 9/12
54000/54000 [==============================] - 4s 72us/sample - loss: 0.0279 - acc: 0.9902 - va
Epoch 10/12
54000/54000 [==============================] - 4s 72us/sample - loss: 0.0230 - acc: 0.9921 - va
Epoch 11/12
54000/54000 [==============================] - 4s 72us/sample - loss: 0.0208 - acc: 0.9931 - va
Epoch 12/12
54000/54000 [==============================] - 4s 72us/sample - loss: 0.0195 - acc: 0.9931 - va
```

In [44]: `draw_learning_curve(model_graph)`






```
In [45]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

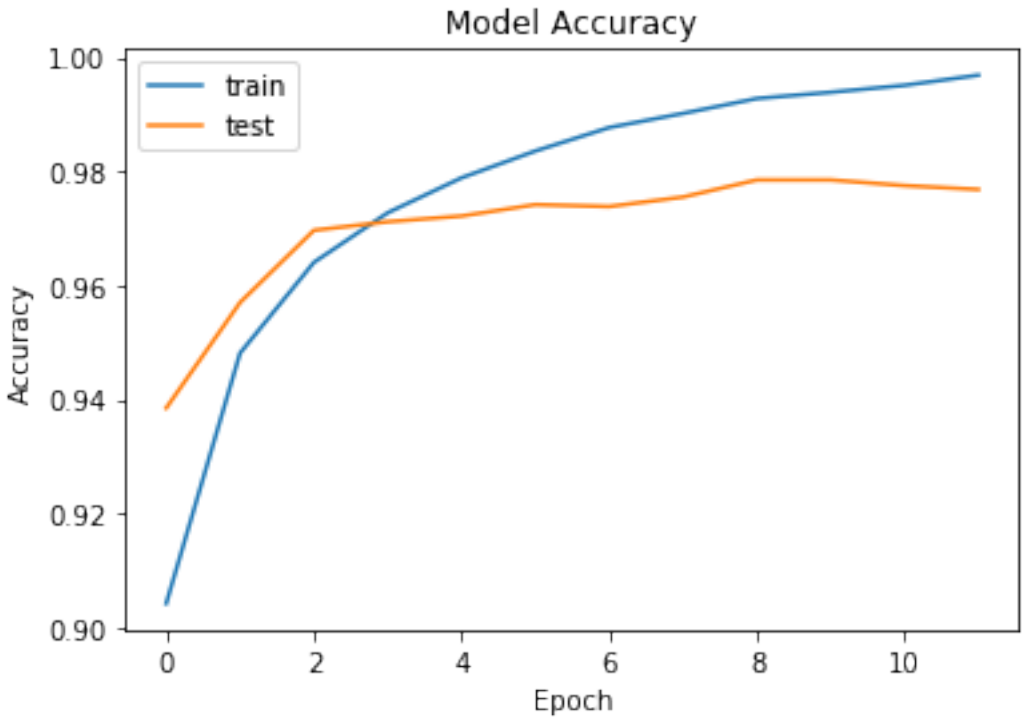
```
60000/60000 [==============================] - 2s 39us/sample - loss: 0.0201 - acc: 0.9946
Train accuracy: 0.9946333
10000/10000 [==============================] - 0s 40us/sample - loss: 0.1121 - acc: 0.9732
Test accuracy: 0.9732
```

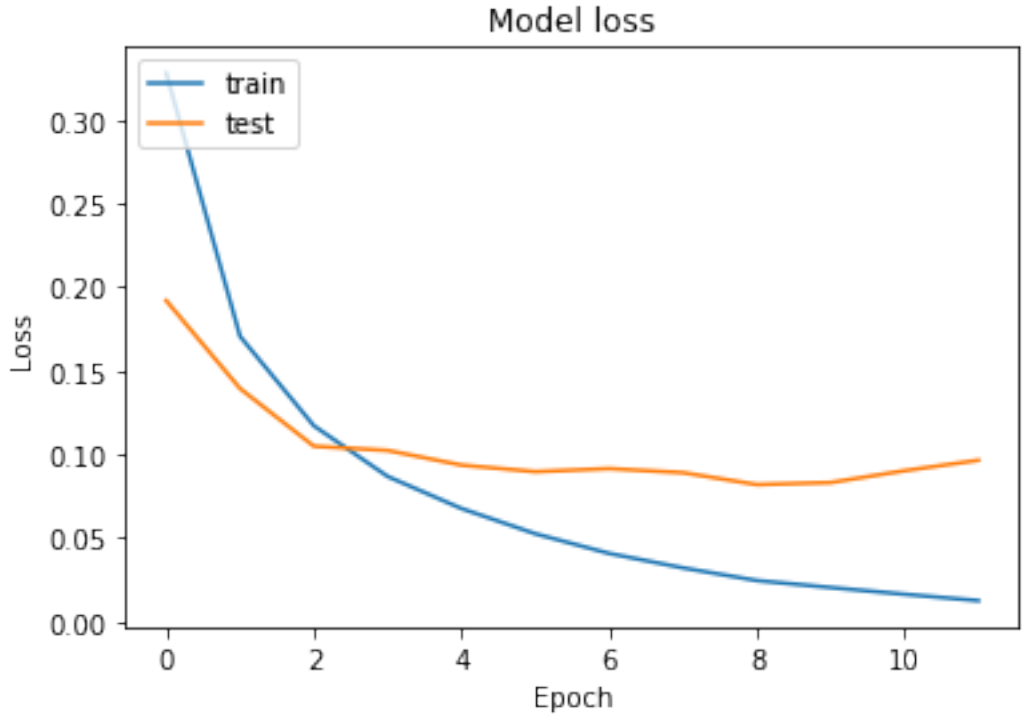
0.4.3 Tanh Activation

```
In [46]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l1=0.01))
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))

         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 4s 81us/sample - loss: 0.3279 - acc: 0.9042 - va
Epoch 2/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.1705 - acc: 0.9481 - va
Epoch 3/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.1170 - acc: 0.9640 - va
Epoch 4/12
54000/54000 [==============================] - 4s 74us/sample - loss: 0.0867 - acc: 0.9727 - va
Epoch 5/12
54000/54000 [==============================] - 4s 74us/sample - loss: 0.0675 - acc: 0.9789 - va
Epoch 6/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.0524 - acc: 0.9836 - va
Epoch 7/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.0406 - acc: 0.9877 - va
Epoch 8/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.0320 - acc: 0.9902 - va
Epoch 9/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.0244 - acc: 0.9928 - va
Epoch 10/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.0202 - acc: 0.9939 - va
Epoch 11/12
54000/54000 [==============================] - 4s 73us/sample - loss: 0.0162 - acc: 0.9951 - va
Epoch 12/12
54000/54000 [==============================] - 4s 74us/sample - loss: 0.0124 - acc: 0.9969 - va
```

```
In [47]: draw_learning_curve(model_graph)
```


Model Accuracy
train
test
Accuracy
Epoch
1.00
0.98
0.96
0.94
0.92
0.90
0
2
4
6
8
10


Model loss
train
test
Loss
Epoch
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
2
4
6
8
10

```
In [48]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

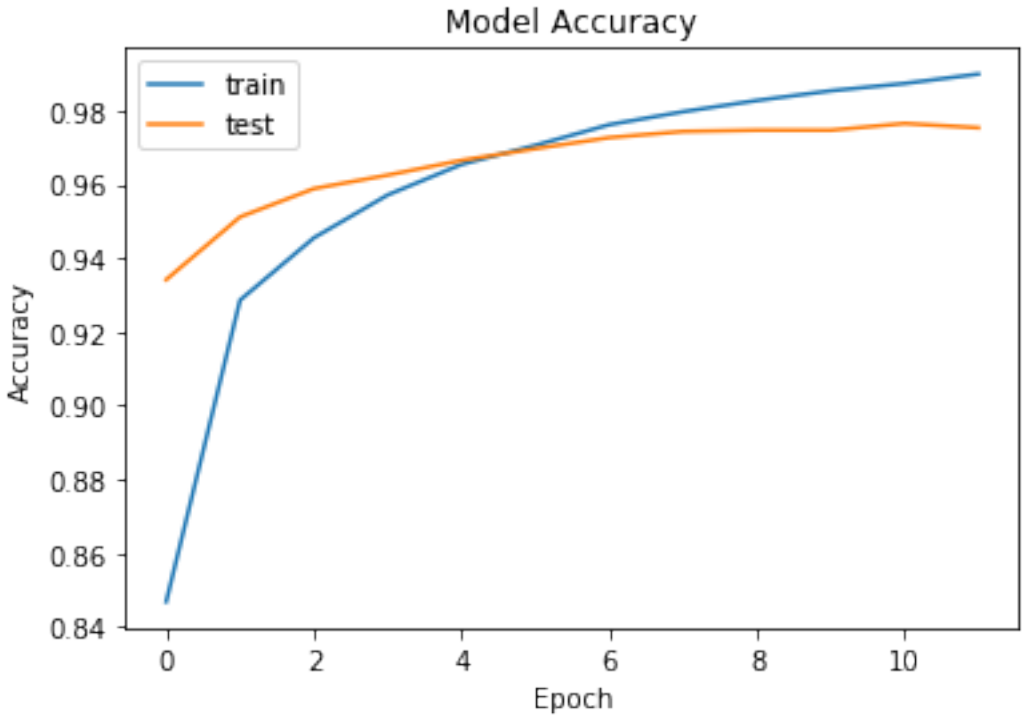
```
60000/60000 [==============================] - 2s 40us/sample - loss: 0.0205 - acc: 0.9945
Train accuracy: 0.99455
10000/10000 [==============================] - 0s 40us/sample - loss: 0.1030 - acc: 0.9718
Test accuracy: 0.9718
```

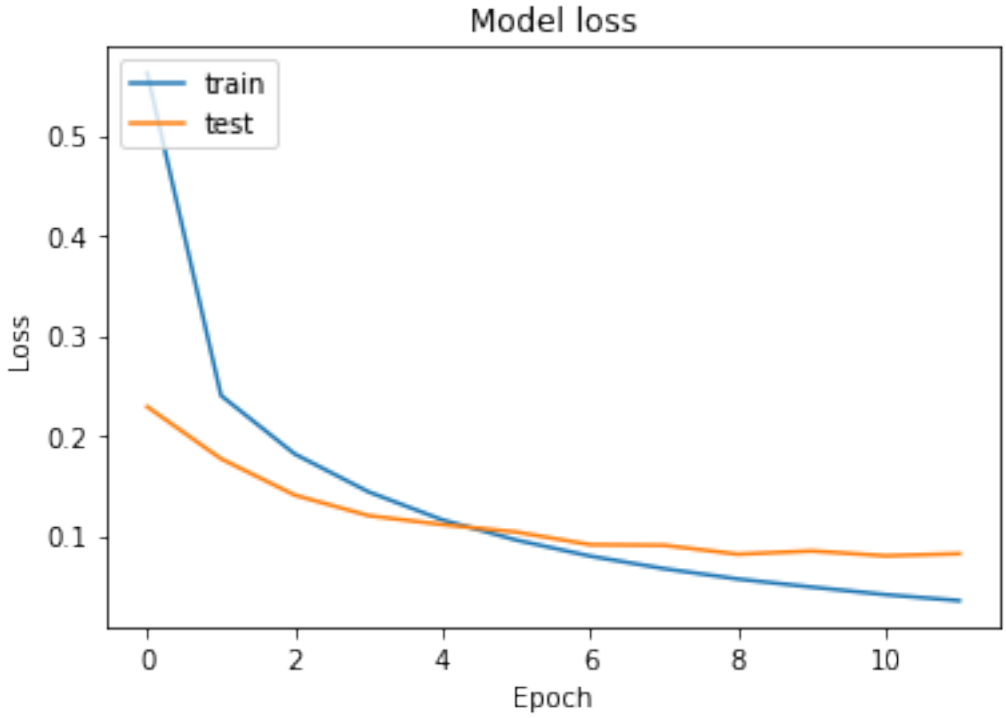
0.4.4 Sigmoid Activation

```
In [49]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l1=0.01))
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))

         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 4s 83us/sample - loss: 0.5622 - acc: 0.8469 - va
Epoch 2/12
54000/54000 [==============================] - 4s 76us/sample - loss: 0.2399 - acc: 0.9286 - va
Epoch 3/12
54000/54000 [==============================] - 4s 76us/sample - loss: 0.1814 - acc: 0.9455 - va
Epoch 4/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.1436 - acc: 0.9571 - va
Epoch 5/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.1156 - acc: 0.9654 - va
Epoch 6/12
54000/54000 [==============================] - 4s 74us/sample - loss: 0.0954 - acc: 0.9706 - va
Epoch 7/12
54000/54000 [==============================] - 4s 74us/sample - loss: 0.0794 - acc: 0.9762 - va
Epoch 8/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.0669 - acc: 0.9797 - va
Epoch 9/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.0565 - acc: 0.9827 - va
Epoch 10/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.0485 - acc: 0.9854 - va
Epoch 11/12
54000/54000 [==============================] - 4s 74us/sample - loss: 0.0409 - acc: 0.9873 - va
Epoch 12/12
54000/54000 [==============================] - 4s 74us/sample - loss: 0.0349 - acc: 0.9899 - va
```

```
In [50]: draw_learning_curve(model_graph)
```





```
In [51]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 2s 41us/sample - loss: 0.0329 - acc: 0.9906
Train accuracy: 0.99065
10000/10000 [==============================] - 0s 41us/sample - loss: 0.0892 - acc: 0.9725
Test accuracy: 0.9725
```

0.5 L2 Regularization

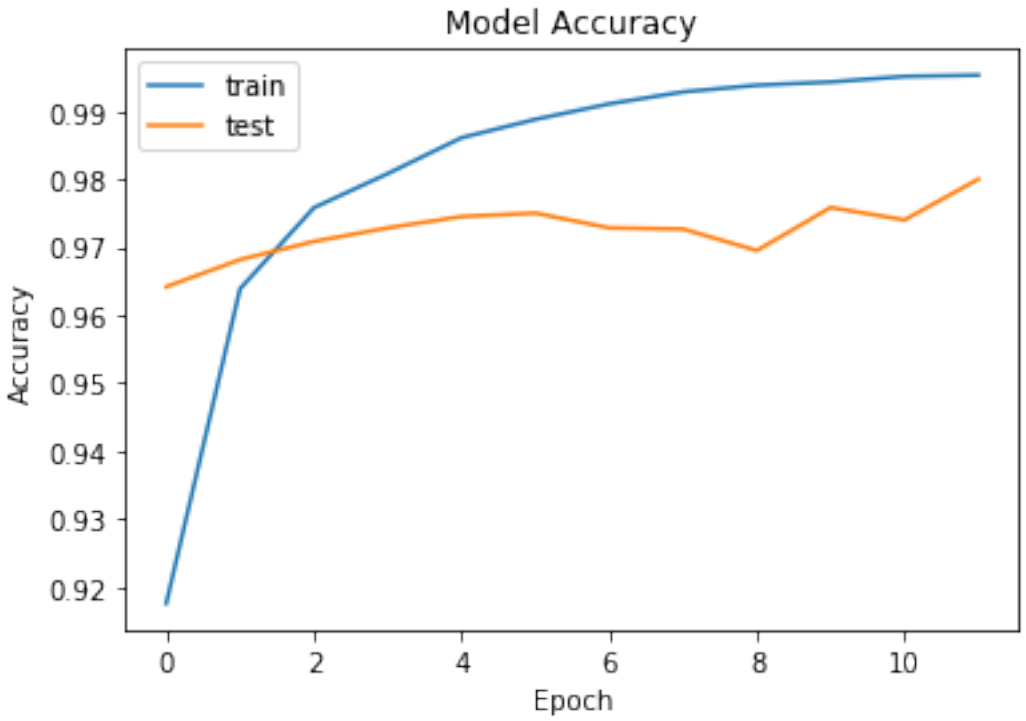
0.5.1 ReLu Activation

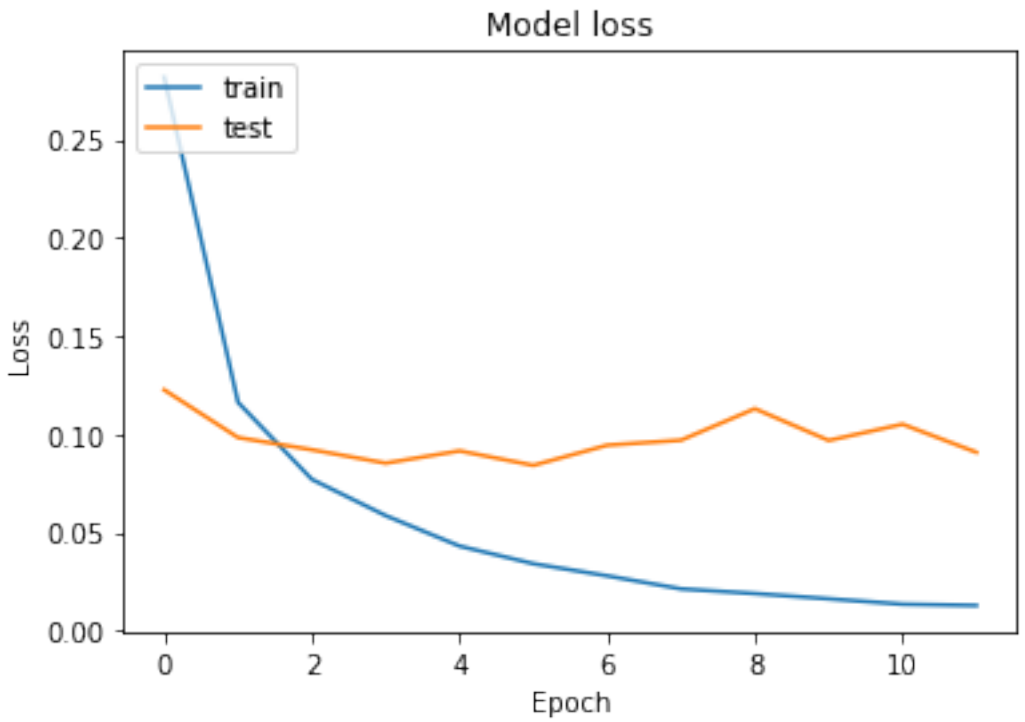
```
In [52]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l2=0.01))
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))

         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 5s 86us/sample - loss: 0.2818 - acc: 0.9176 - va
Epoch 2/12
54000/54000 [==============================] - 4s 77us/sample - loss: 0.1163 - acc: 0.9639 - va
Epoch 3/12
54000/54000 [==============================] - 4s 76us/sample - loss: 0.0771 - acc: 0.9758 - va
Epoch 4/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.0586 - acc: 0.9808 - va
Epoch 5/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.0431 - acc: 0.9861 - va
Epoch 6/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.0341 - acc: 0.9888 - va
Epoch 7/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.0280 - acc: 0.9911 - va
Epoch 8/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.0213 - acc: 0.9928 - va
Epoch 9/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.0189 - acc: 0.9938 - va
Epoch 10/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.0163 - acc: 0.9943 - va
```

```
Epoch 11/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.0135 - acc: 0.9951 - va
Epoch 12/12
54000/54000 [==============================] - 4s 75us/sample - loss: 0.0129 - acc: 0.9953 - va
```

```
In [53]: draw_learning_curve(model_graph)
```






```
In [54]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 3s 42us/sample - loss: 0.0168 - acc: 0.9955
Train accuracy: 0.99546665
10000/10000 [==============================] - 0s 42us/sample - loss: 0.1142 - acc: 0.9742
Test accuracy: 0.9742
```

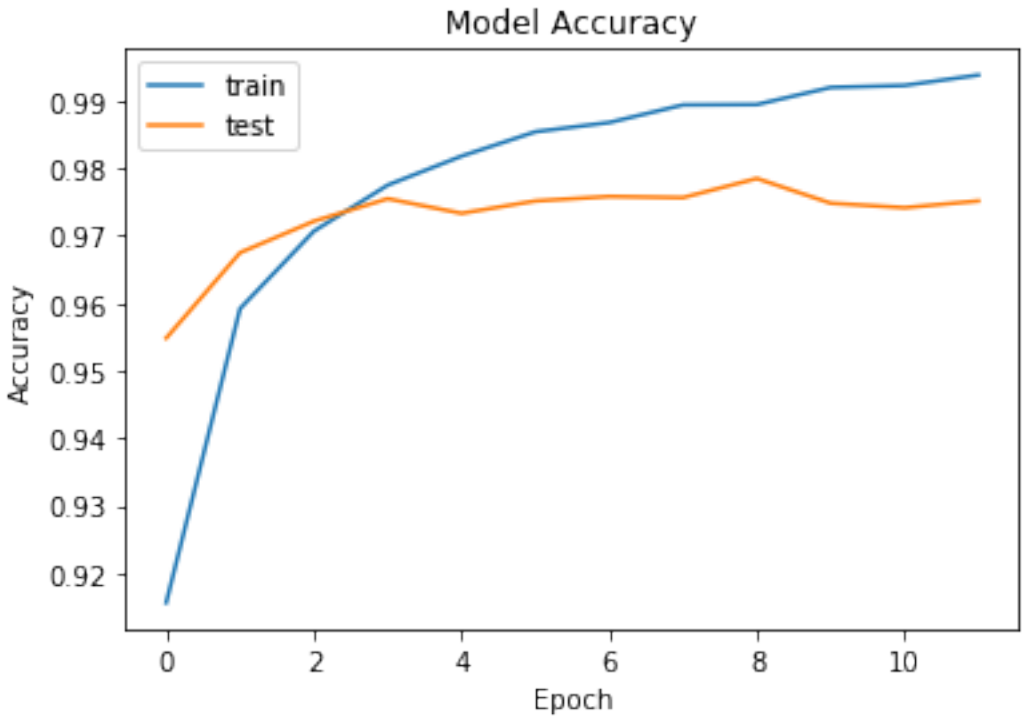
0.5.2 Leaky ReLu Activation

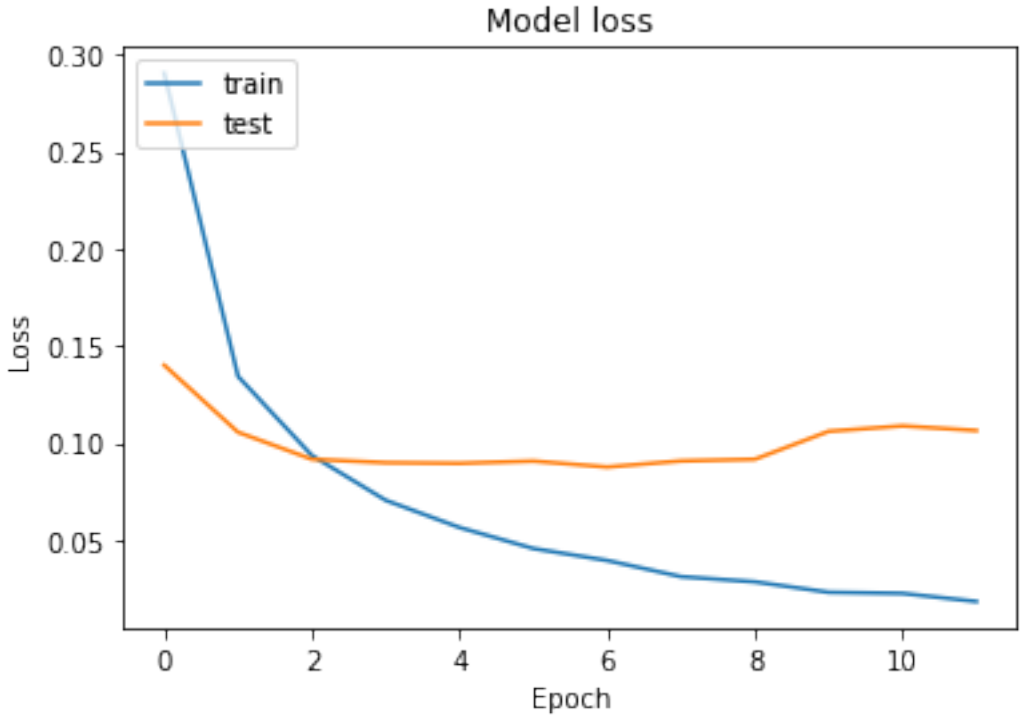
```
In [55]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l2=0.01))
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))

         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
```

```
54000/54000 [==============================] - 5s 85us/sample - loss: 0.2903 - acc: 0.9156 - va
Epoch 2/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.1342 - acc: 0.9592 - va
Epoch 3/12
54000/54000 [==============================] - 4s 77us/sample - loss: 0.0935 - acc: 0.9707 - va
Epoch 4/12
54000/54000 [==============================] - 4s 77us/sample - loss: 0.0704 - acc: 0.9775 - va
Epoch 5/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.0564 - acc: 0.9818 - va
Epoch 6/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.0456 - acc: 0.9854 - va
Epoch 7/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.0394 - acc: 0.9868 - va
Epoch 8/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.0311 - acc: 0.9894 - va
Epoch 9/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.0284 - acc: 0.9895 - va
Epoch 10/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.0231 - acc: 0.9920 - va
Epoch 11/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.0225 - acc: 0.9923 - va
Epoch 12/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.0183 - acc: 0.9938 - va
```

In [56]: `draw_learning_curve(model_graph)`






```
In [57]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

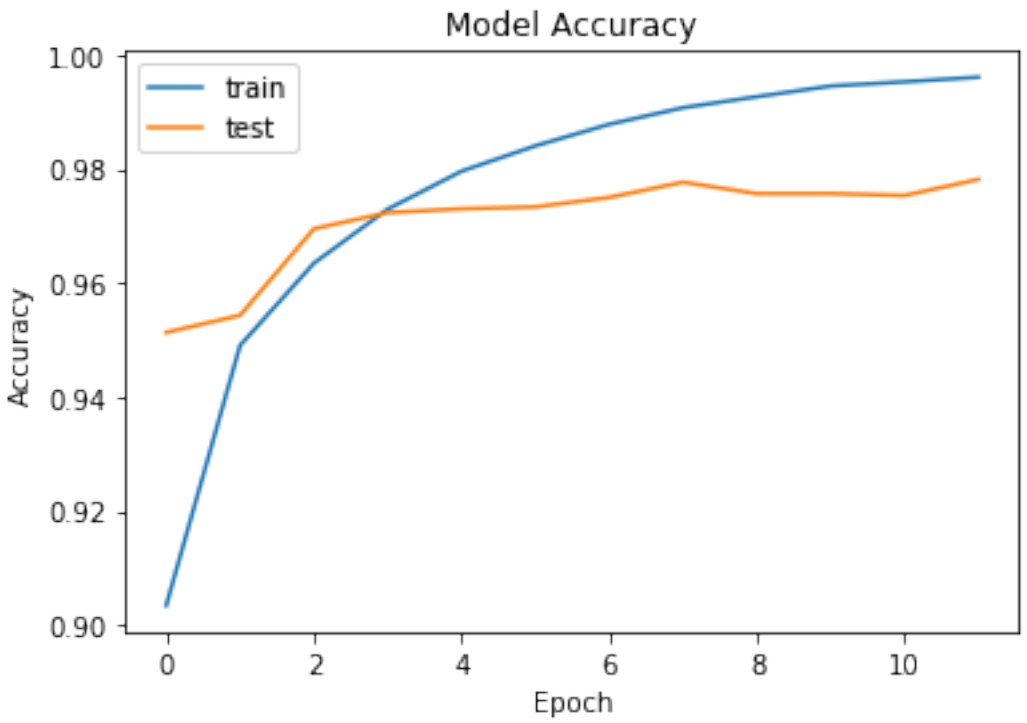
```
60000/60000 [==============================] - 3s 43us/sample - loss: 0.0248 - acc: 0.9926
Train accuracy: 0.99263334
10000/10000 [==============================] - 0s 42us/sample - loss: 0.1224 - acc: 0.9713
Test accuracy: 0.9713
```

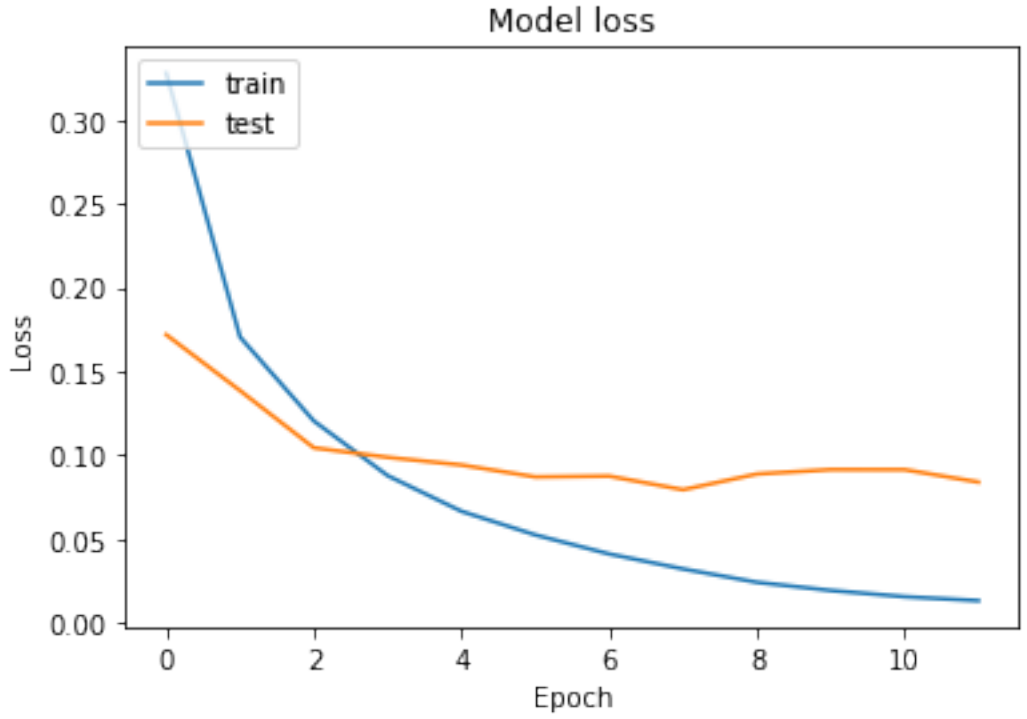
0.5.3 Tanh Activation

```
In [58]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l2=0.01))
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 5s 86us/sample - loss: 0.3277 - acc: 0.9035 - va
Epoch 2/12
54000/54000 [==============================] - 4s 79us/sample - loss: 0.1704 - acc: 0.9490 - va
Epoch 3/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.1203 - acc: 0.9635 - va
Epoch 4/12
54000/54000 [==============================] - 4s 77us/sample - loss: 0.0877 - acc: 0.9729 - va
Epoch 5/12
54000/54000 [==============================] - 4s 77us/sample - loss: 0.0664 - acc: 0.9796 - va
Epoch 6/12
54000/54000 [==============================] - 4s 77us/sample - loss: 0.0523 - acc: 0.9840 - va
Epoch 7/12
54000/54000 [==============================] - 4s 77us/sample - loss: 0.0410 - acc: 0.9878 - va
Epoch 8/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.0321 - acc: 0.9907 - va
Epoch 9/12
54000/54000 [==============================] - 4s 79us/sample - loss: 0.0241 - acc: 0.9926 - va
Epoch 10/12
54000/54000 [==============================] - 4s 79us/sample - loss: 0.0192 - acc: 0.9945 - va
Epoch 11/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.0155 - acc: 0.9953 - va
Epoch 12/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.0131 - acc: 0.9961 - va
```

```
In [59]: draw_learning_curve(model_graph)
```


Model Accuracy
train
test
Accuracy
Epoch
1.00
0.98
0.96
0.94
0.92
0.90
0
2
4
6
8
10


Model loss
train
test
Loss
Epoch
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
2
4
6
8
10

```
In [60]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

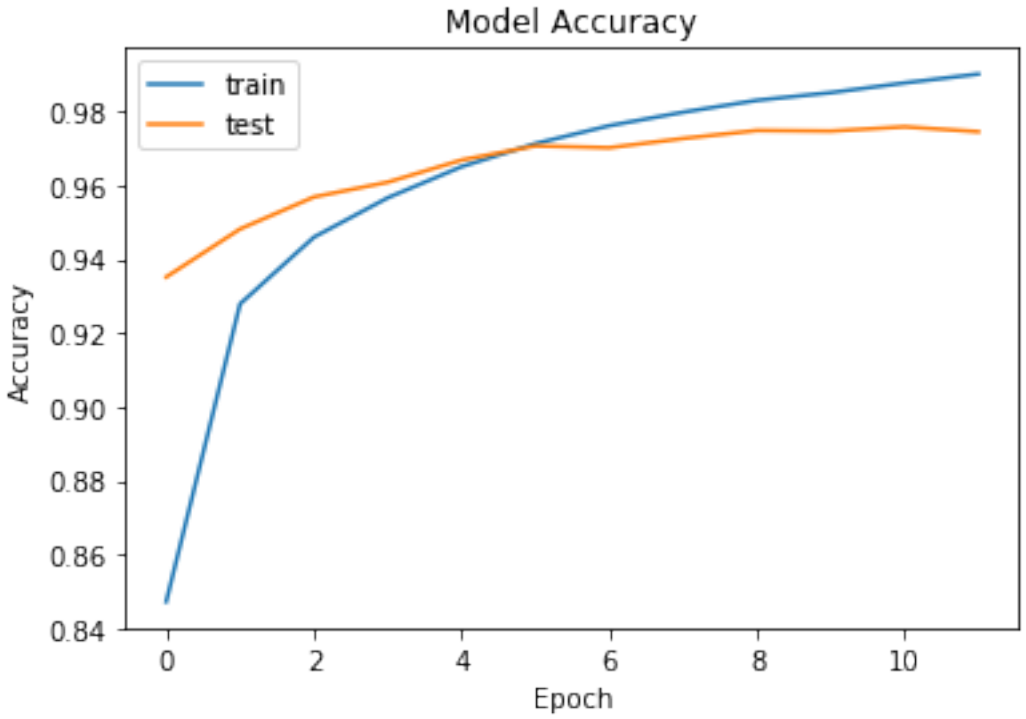
```
60000/60000 [==============================] - 3s 44us/sample - loss: 0.0151 - acc: 0.9963
Train accuracy: 0.9963167
10000/10000 [==============================] - 0s 43us/sample - loss: 0.0987 - acc: 0.9754
Test accuracy: 0.9754
```

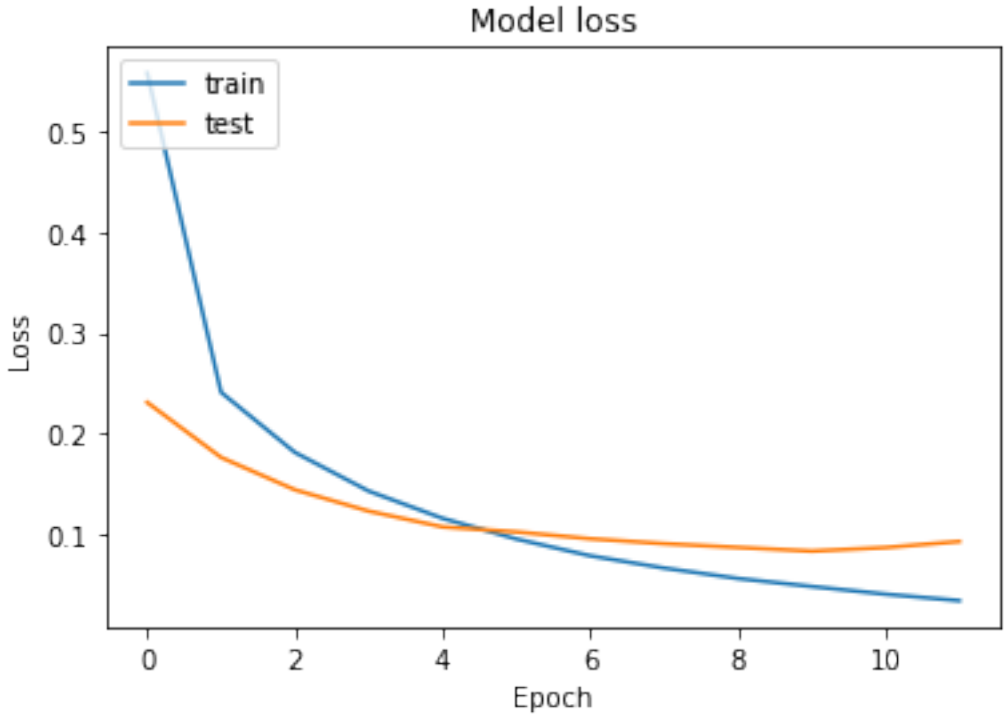
0.5.4 Sigmoid Activation

```
In [61]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l2=0.01))
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))

         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 5s 86us/sample - loss: 0.5579 - acc: 0.8473 - va
Epoch 2/12
54000/54000 [==============================] - 4s 79us/sample - loss: 0.2411 - acc: 0.9279 - va
Epoch 3/12
54000/54000 [==============================] - 4s 77us/sample - loss: 0.1814 - acc: 0.9459 - va
Epoch 4/12
54000/54000 [==============================] - 4s 77us/sample - loss: 0.1430 - acc: 0.9566 - va
Epoch 5/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.1160 - acc: 0.9650 - va
Epoch 6/12
54000/54000 [==============================] - 4s 77us/sample - loss: 0.0955 - acc: 0.9713 - va
Epoch 7/12
54000/54000 [==============================] - 4s 77us/sample - loss: 0.0788 - acc: 0.9761 - va
Epoch 8/12
54000/54000 [==============================] - 4s 79us/sample - loss: 0.0666 - acc: 0.9798 - va
Epoch 9/12
54000/54000 [==============================] - 4s 79us/sample - loss: 0.0563 - acc: 0.9830 - va
Epoch 10/12
54000/54000 [==============================] - 4s 79us/sample - loss: 0.0484 - acc: 0.9851 - va
Epoch 11/12
54000/54000 [==============================] - 4s 79us/sample - loss: 0.0407 - acc: 0.9877 - va
Epoch 12/12
54000/54000 [==============================] - 4s 78us/sample - loss: 0.0343 - acc: 0.9901 - va
```

In [62]: draw_learning_curve(model_graph)





```
In [63]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 3s 43us/sample - loss: 0.0373 - acc: 0.9891
Train accuracy: 0.98913336
10000/10000 [==============================] - 0s 43us/sample - loss: 0.1032 - acc: 0.9712
Test accuracy: 0.9712
```

0.6 Dropout + Batch Normalization

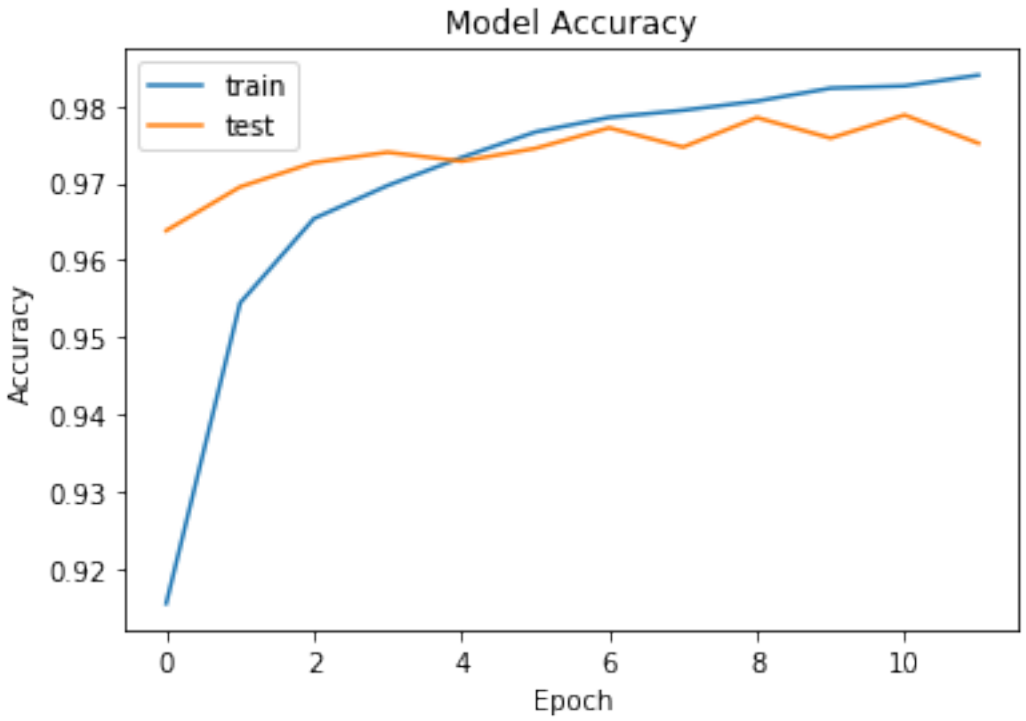
0.6.1 ReLu Activation

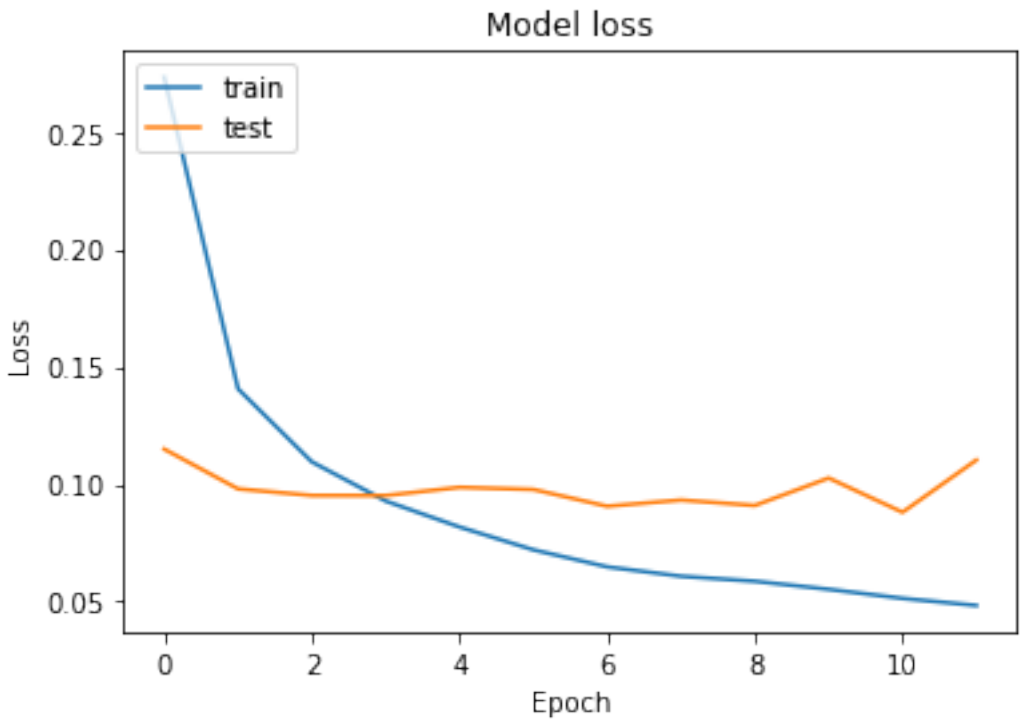
```
In [64]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Dropout(0.2))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 6s 103us/sample - loss: 0.2732 - acc: 0.9154 - v
Epoch 2/12
54000/54000 [==============================] - 5s 95us/sample - loss: 0.1406 - acc: 0.9544 - va
Epoch 3/12
54000/54000 [==============================] - 5s 94us/sample - loss: 0.1095 - acc: 0.9654 - va
Epoch 4/12
54000/54000 [==============================] - 5s 93us/sample - loss: 0.0928 - acc: 0.9697 - va
Epoch 5/12
54000/54000 [==============================] - 5s 93us/sample - loss: 0.0818 - acc: 0.9733 - va
Epoch 6/12
54000/54000 [==============================] - 5s 94us/sample - loss: 0.0721 - acc: 0.9766 - va
Epoch 7/12
54000/54000 [==============================] - 5s 94us/sample - loss: 0.0648 - acc: 0.9785 - va
Epoch 8/12
54000/54000 [==============================] - 5s 93us/sample - loss: 0.0608 - acc: 0.9794 - va
Epoch 9/12
54000/54000 [==============================] - 5s 93us/sample - loss: 0.0587 - acc: 0.9806 - va
Epoch 10/12
54000/54000 [==============================] - 5s 94us/sample - loss: 0.0552 - acc: 0.9823 - va
```

```
Epoch 11/12
54000/54000 [==============================] - 5s 94us/sample - loss: 0.0514 - acc: 0.9826 - va
Epoch 12/12
54000/54000 [==============================] - 5s 93us/sample - loss: 0.0484 - acc: 0.9840 - va
```

```
In [65]: draw_learning_curve(model_graph)
```






```
In [66]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 3s 48us/sample - loss: 0.0310 - acc: 0.9913
Train accuracy: 0.9913167
10000/10000 [==============================] - 0s 48us/sample - loss: 0.1047 - acc: 0.9750
Test accuracy: 0.975
```

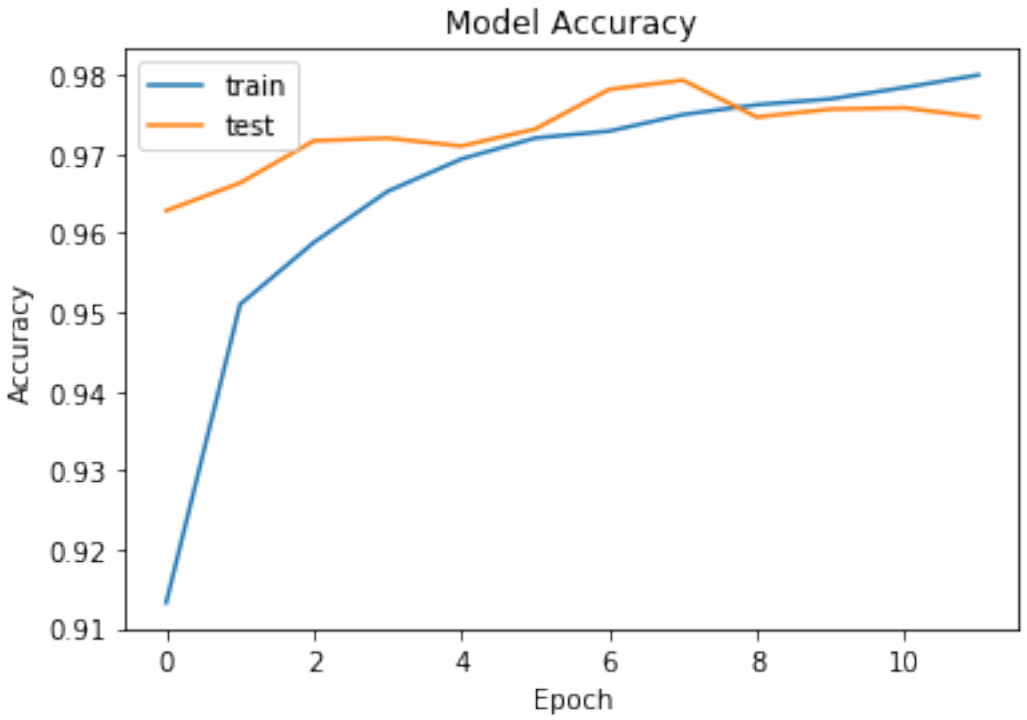
0.6.2 Leaky ReLu Activation

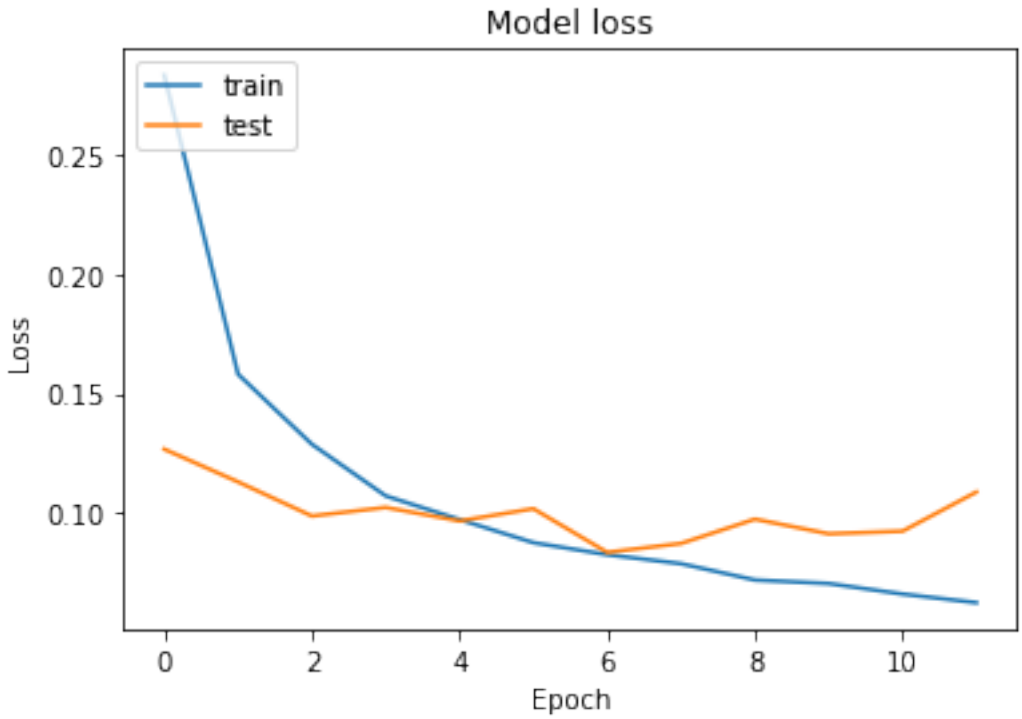
```
In [67]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Dropout(0.2))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
```

```
54000/54000 [==============================] - 6s 104us/sample - loss: 0.2837 - acc: 0.9132 - v
Epoch 2/12
54000/54000 [==============================] - 5s 94us/sample - loss: 0.1582 - acc: 0.9510 - va
Epoch 3/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.1288 - acc: 0.9588 - va
Epoch 4/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.1071 - acc: 0.9653 - va
Epoch 5/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0972 - acc: 0.9694 - va
Epoch 6/12
54000/54000 [==============================] - 5s 94us/sample - loss: 0.0875 - acc: 0.9720 - va
Epoch 7/12
54000/54000 [==============================] - 5s 95us/sample - loss: 0.0825 - acc: 0.9729 - va
Epoch 8/12
54000/54000 [==============================] - 5s 94us/sample - loss: 0.0787 - acc: 0.9750 - va
Epoch 9/12
54000/54000 [==============================] - 5s 95us/sample - loss: 0.0718 - acc: 0.9762 - va
Epoch 10/12
54000/54000 [==============================] - 5s 95us/sample - loss: 0.0703 - acc: 0.9770 - va
Epoch 11/12
54000/54000 [==============================] - 5s 95us/sample - loss: 0.0659 - acc: 0.9784 - va
Epoch 12/12
54000/54000 [==============================] - 5s 94us/sample - loss: 0.0623 - acc: 0.9800 - va
```

In [68]: `draw_learning_curve(model_graph)`






```
In [69]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

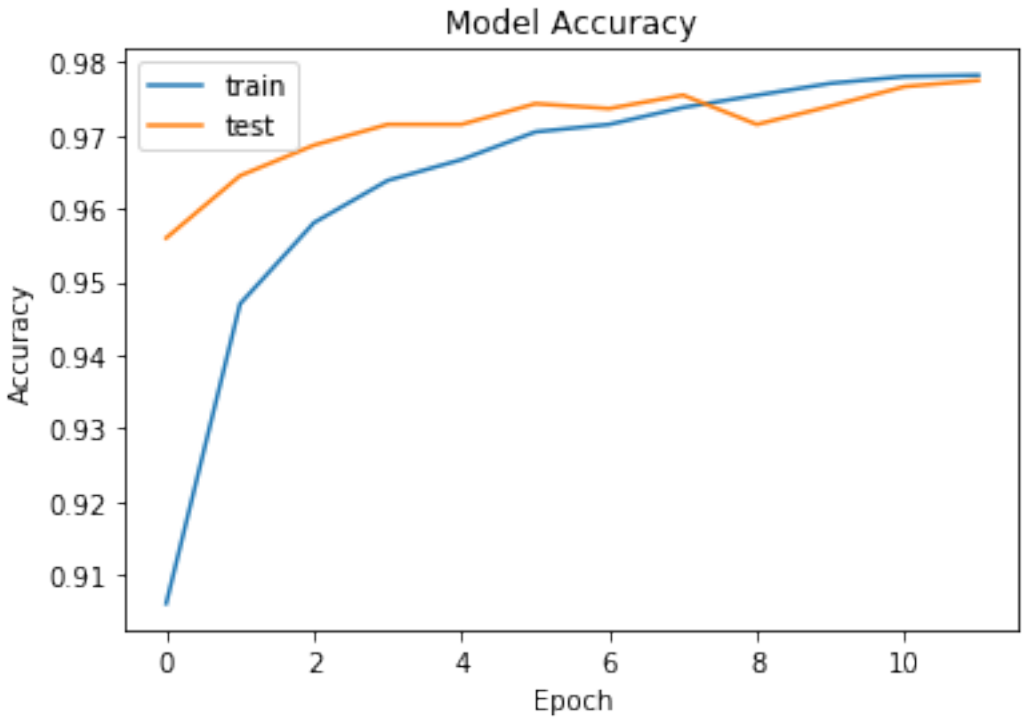
```
60000/60000 [==============================] - 3s 49us/sample - loss: 0.0341 - acc: 0.9905
Train accuracy: 0.99045
10000/10000 [==============================] - 1s 51us/sample - loss: 0.1075 - acc: 0.9731
Test accuracy: 0.9731
```

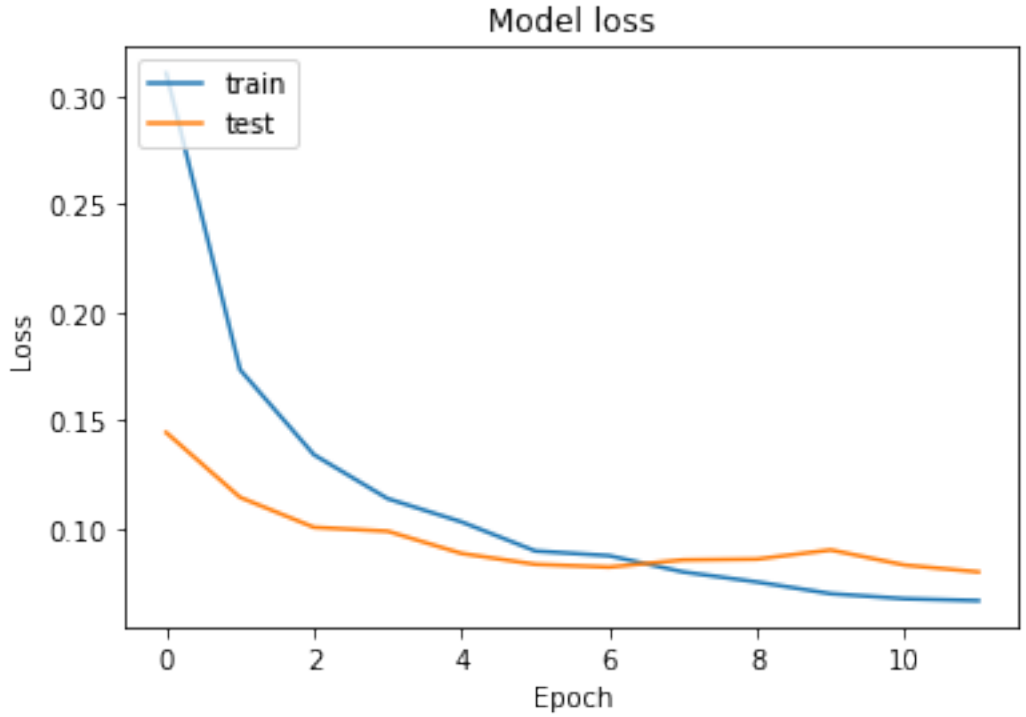
0.6.3 Tanh Activation

```
In [70]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Dropout(0.2))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 6s 105us/sample - loss: 0.3105 - acc: 0.9061 - v
Epoch 2/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.1733 - acc: 0.9470 - va
Epoch 3/12
54000/54000 [==============================] - 5s 95us/sample - loss: 0.1340 - acc: 0.9581 - va
Epoch 4/12
54000/54000 [==============================] - 5s 95us/sample - loss: 0.1136 - acc: 0.9639 - va
Epoch 5/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.1029 - acc: 0.9667 - va
Epoch 6/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0894 - acc: 0.9705 - va
Epoch 7/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0872 - acc: 0.9715 - va
Epoch 8/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0797 - acc: 0.9738 - va
Epoch 9/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0751 - acc: 0.9755 - va
Epoch 10/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0696 - acc: 0.9771 - va
Epoch 11/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0673 - acc: 0.9780 - va
Epoch 12/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0664 - acc: 0.9782 - va
```

```
In [71]: draw_learning_curve(model_graph)
```


Model Accuracy
train
test
Accuracy
Epoch
0.98
0.97
0.96
0.95
0.94
0.93
0.92
0.91
0
2
4
6
8
10


Model loss
train
test
Loss
Epoch
0.30
0.25
0.20
0.15
0.10
0
2
4
6
8
10

```
In [72]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

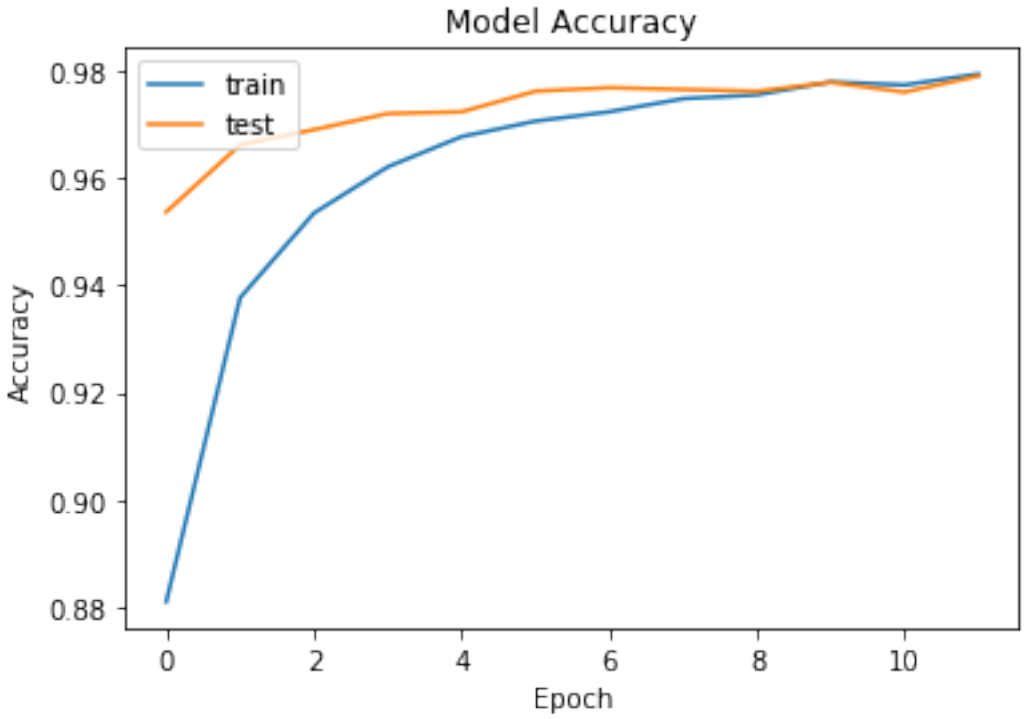
```
60000/60000 [==============================] - 3s 49us/sample - loss: 0.0294 - acc: 0.9913
Train accuracy: 0.99135
10000/10000 [==============================] - 0s 49us/sample - loss: 0.0961 - acc: 0.9731
Test accuracy: 0.9731
```

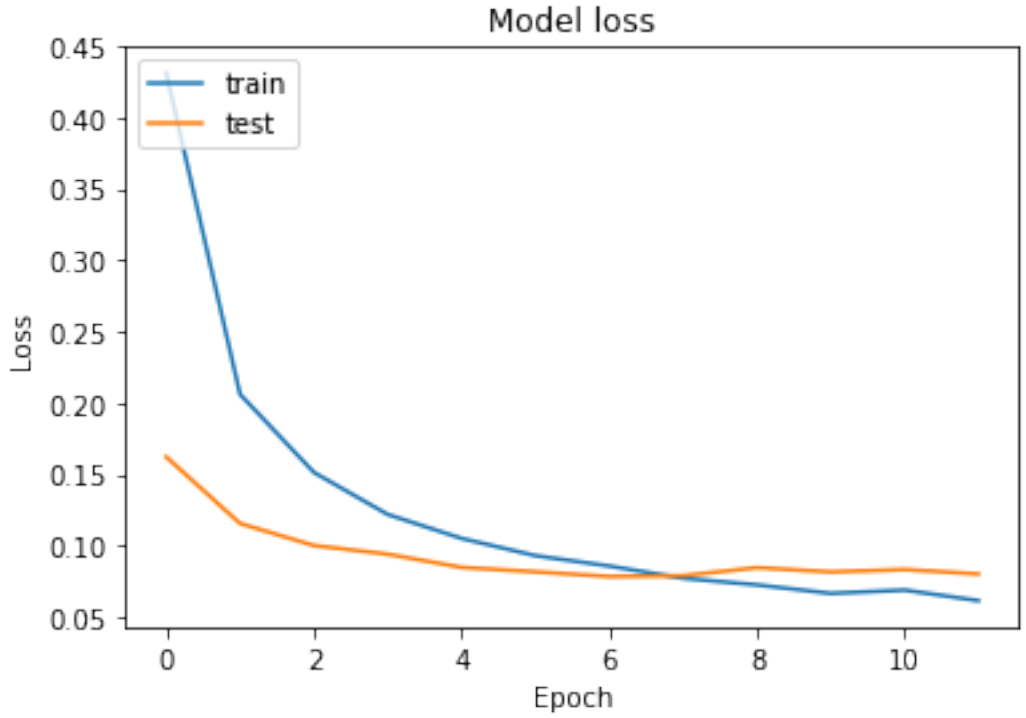
0.6.4 Sigmoid Activation

```
In [73]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Dropout(0.2))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 6s 106us/sample - loss: 0.4310 - acc: 0.8810 - v
Epoch 2/12
54000/54000 [==============================] - 5s 95us/sample - loss: 0.2061 - acc: 0.9377 - va
Epoch 3/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.1514 - acc: 0.9534 - va
Epoch 4/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.1220 - acc: 0.9621 - va
Epoch 5/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.1052 - acc: 0.9677 - va
Epoch 6/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0932 - acc: 0.9706 - va
Epoch 7/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0859 - acc: 0.9723 - va
Epoch 8/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0773 - acc: 0.9748 - va
Epoch 9/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0727 - acc: 0.9755 - va
Epoch 10/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0667 - acc: 0.9779 - va
Epoch 11/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0691 - acc: 0.9773 - va
Epoch 12/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0617 - acc: 0.9794 - va
```

```
In [74]: draw_learning_curve(model_graph)
```





```
In [75]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 3s 49us/sample - loss: 0.0284 - acc: 0.9916
Train accuracy: 0.99158335
10000/10000 [==============================] - 0s 49us/sample - loss: 0.0858 - acc: 0.9751
Test accuracy: 0.9751
```

0.7 Batch Normalization + L2 Regularization

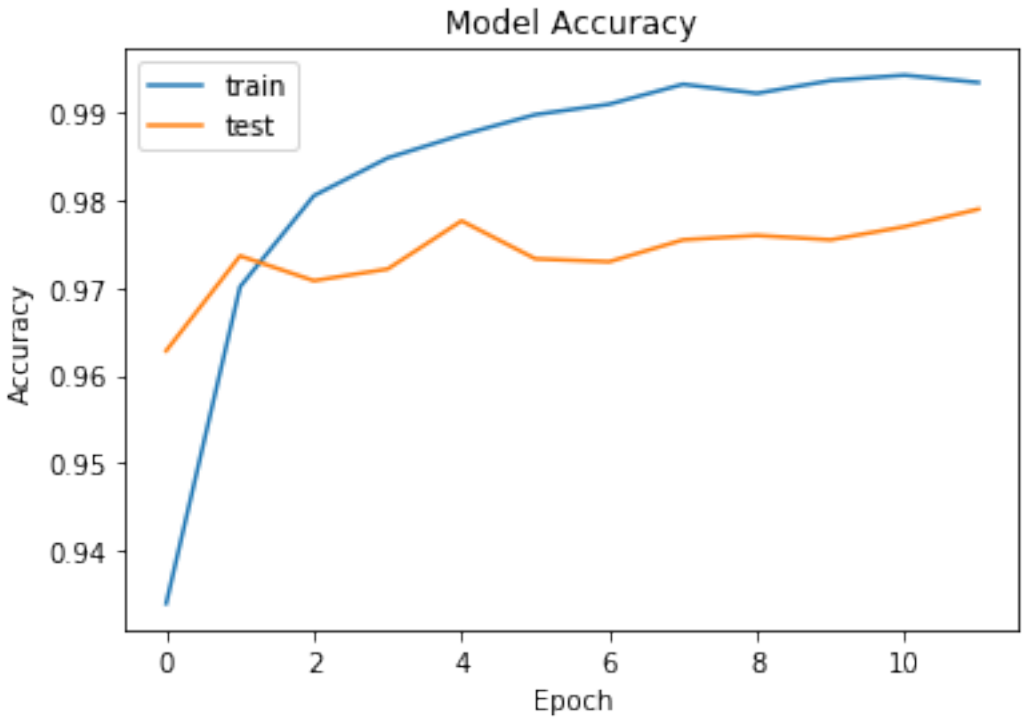
0.7.1 ReLu Activation

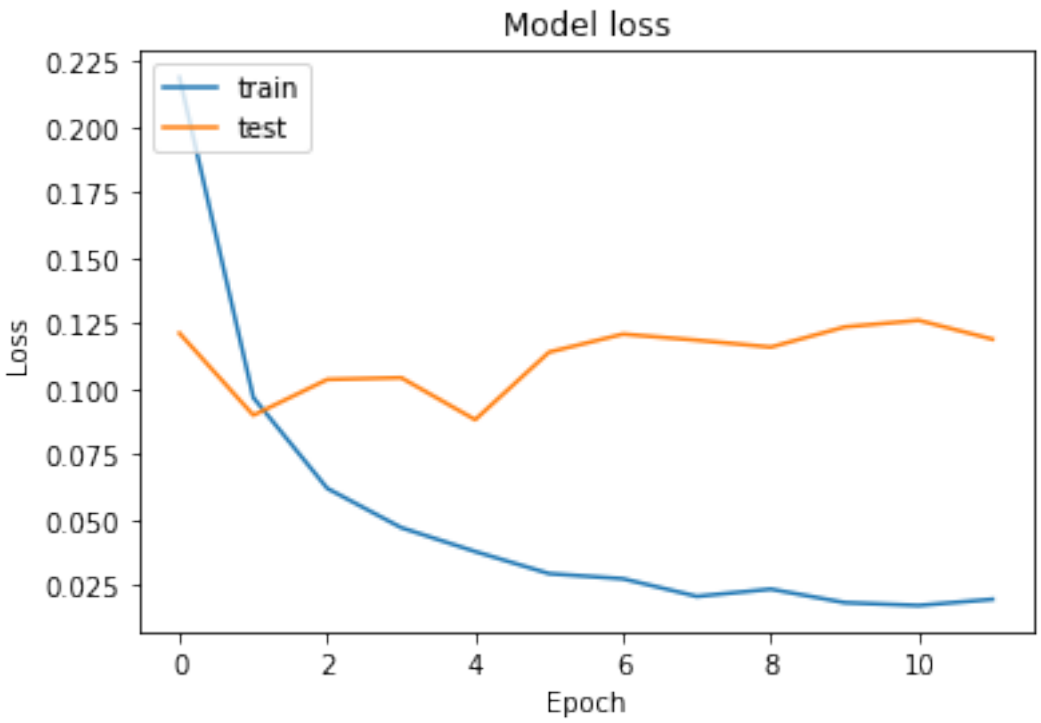
```
In [76]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l2=0.01))
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 6s 106us/sample - loss: 0.2191 - acc: 0.9340 - v
Epoch 2/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0966 - acc: 0.9701 - va
Epoch 3/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0620 - acc: 0.9805 - va
Epoch 4/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0470 - acc: 0.9849 - va
Epoch 5/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0378 - acc: 0.9875 - va
Epoch 6/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0294 - acc: 0.9898 - va
Epoch 7/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0274 - acc: 0.9910 - va
Epoch 8/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0207 - acc: 0.9933 - va
Epoch 9/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0235 - acc: 0.9922 - va
Epoch 10/12
54000/54000 [==============================] - 5s 95us/sample - loss: 0.0183 - acc: 0.9937 - va
```

```
Epoch 11/12
54000/54000 [==============================] - 5s 95us/sample - loss: 0.0172 - acc: 0.9943 - va
Epoch 12/12
54000/54000 [==============================] - 5s 95us/sample - loss: 0.0196 - acc: 0.9935 - va
```

```
In [77]: draw_learning_curve(model_graph)
```






```
In [78]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 3s 51us/sample - loss: 0.0217 - acc: 0.9947
Train accuracy: 0.9947
10000/10000 [==============================] - 0s 50us/sample - loss: 0.1266 - acc: 0.9757
Test accuracy: 0.9757
```

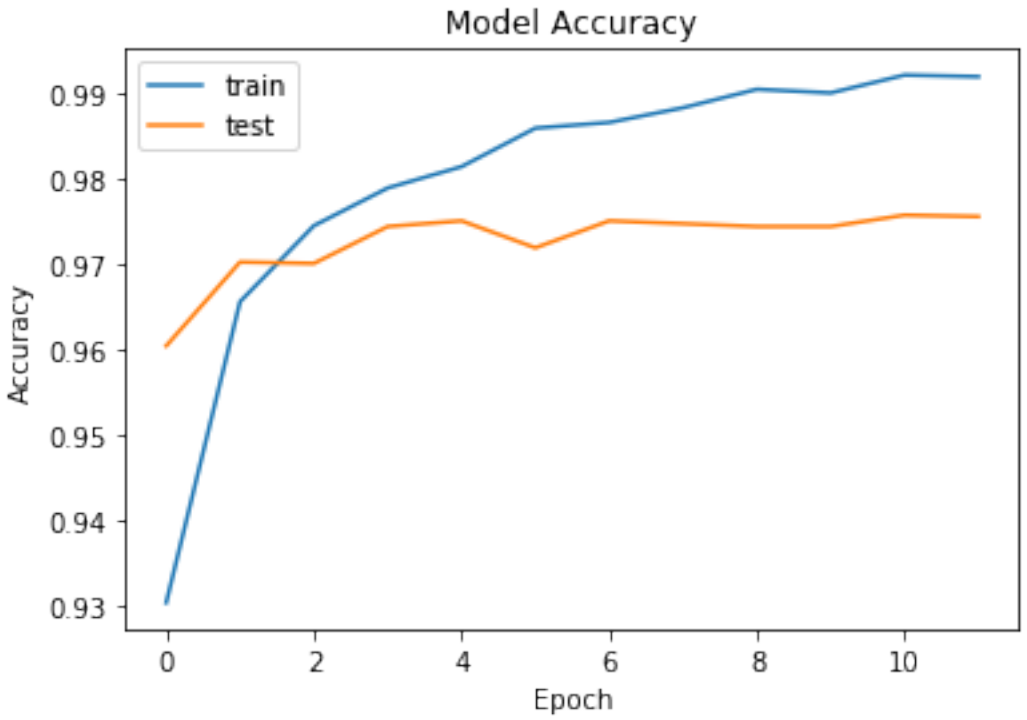
0.7.2 Leaky ReLu Activation

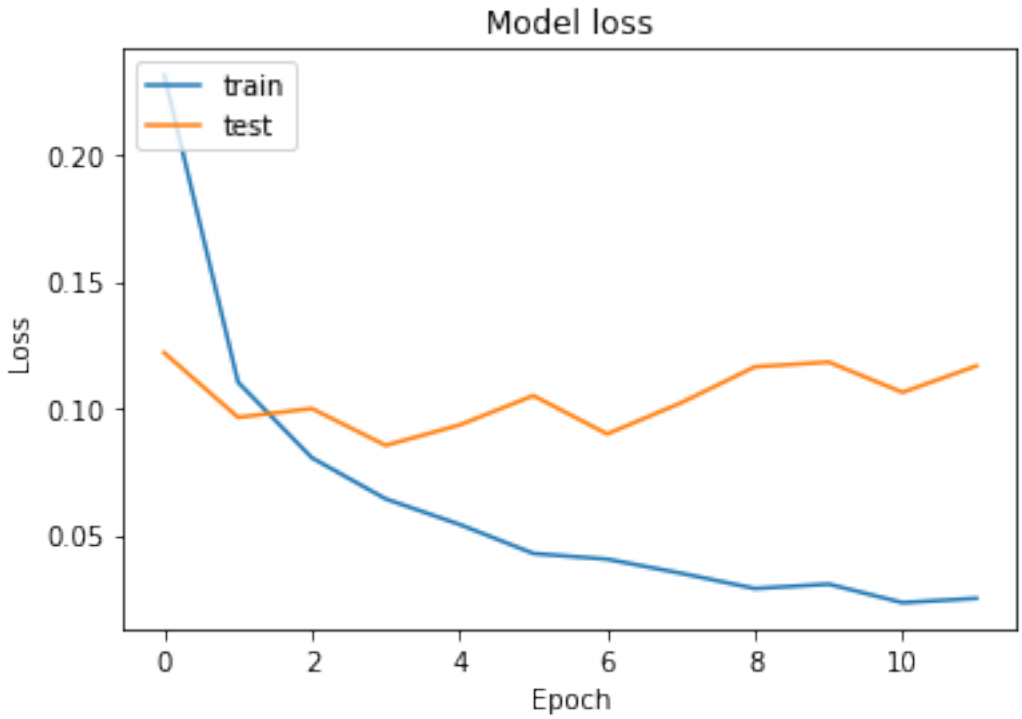
```
In [79]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l2=0.01))
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
```

```
54000/54000 [==============================] - 6s 107us/sample - loss: 0.2314 - acc: 0.9304 - v
Epoch 2/12
54000/54000 [==============================] - 5s 98us/sample - loss: 0.1103 - acc: 0.9657 - va
Epoch 3/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0805 - acc: 0.9746 - va
Epoch 4/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0643 - acc: 0.9790 - va
Epoch 5/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0543 - acc: 0.9815 - va
Epoch 6/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0428 - acc: 0.9860 - va
Epoch 7/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0406 - acc: 0.9867 - va
Epoch 8/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0350 - acc: 0.9884 - va
Epoch 9/12
54000/54000 [==============================] - 5s 96us/sample - loss: 0.0290 - acc: 0.9906 - va
Epoch 10/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0307 - acc: 0.9902 - va
Epoch 11/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0234 - acc: 0.9923 - va
Epoch 12/12
54000/54000 [==============================] - 5s 98us/sample - loss: 0.0251 - acc: 0.9921 - va
```

In [80]: `draw_learning_curve(model_graph)`






```
In [81]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

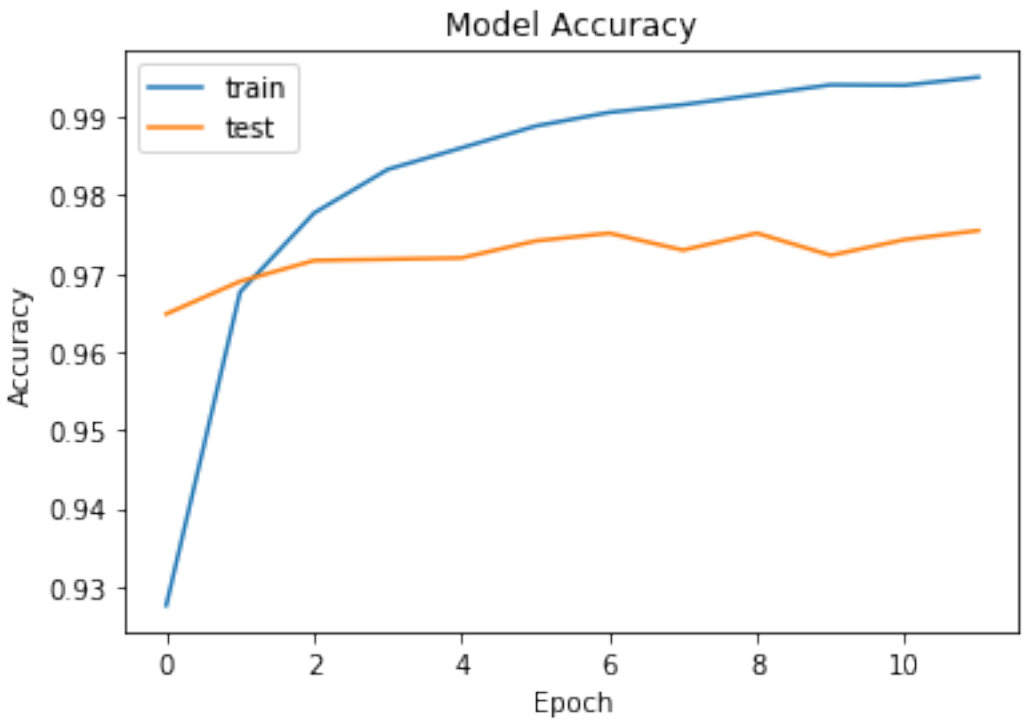
```
60000/60000 [==============================] - 3s 53us/sample - loss: 0.0282 - acc: 0.9918
Train accuracy: 0.9917667
10000/10000 [==============================] - 1s 52us/sample - loss: 0.1563 - acc: 0.9699
Test accuracy: 0.9699
```

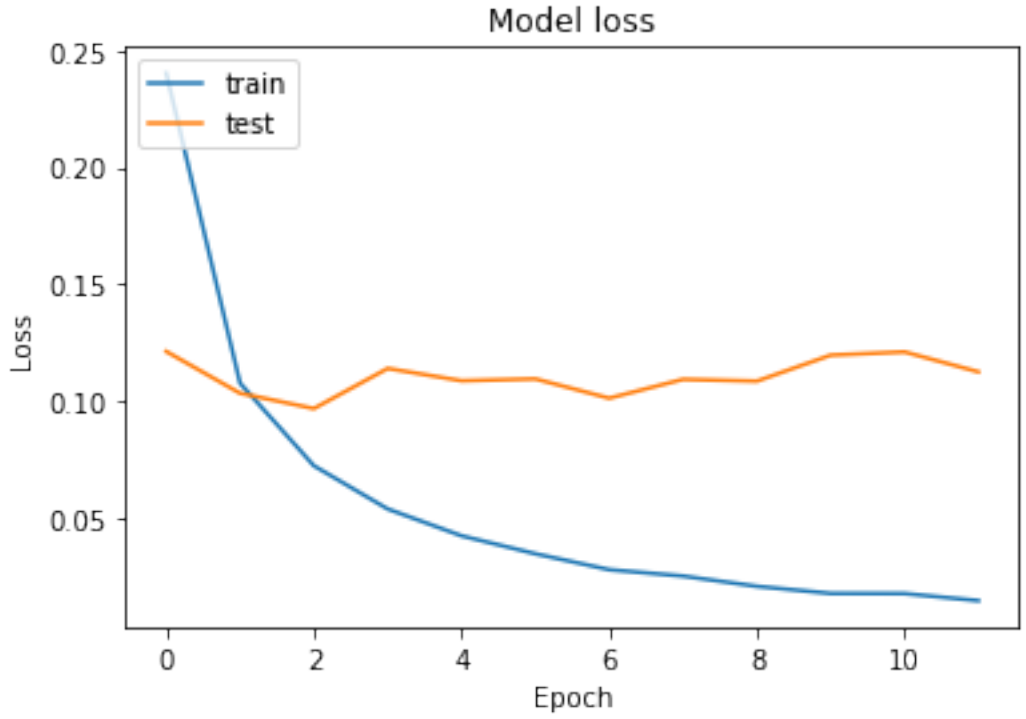
0.7.3 Tanh Activation

```
In [82]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l2=0.01))
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 6s 108us/sample - loss: 0.2405 - acc: 0.9276 - v
Epoch 2/12
54000/54000 [==============================] - 5s 99us/sample - loss: 0.1076 - acc: 0.9676 - va
Epoch 3/12
54000/54000 [==============================] - 5s 99us/sample - loss: 0.0722 - acc: 0.9777 - va
Epoch 4/12
54000/54000 [==============================] - 5s 98us/sample - loss: 0.0536 - acc: 0.9833 - va
Epoch 5/12
54000/54000 [==============================] - 5s 98us/sample - loss: 0.0421 - acc: 0.9861 - va
Epoch 6/12
54000/54000 [==============================] - 5s 98us/sample - loss: 0.0344 - acc: 0.9889 - va
Epoch 7/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0276 - acc: 0.9906 - va
Epoch 8/12
54000/54000 [==============================] - 5s 99us/sample - loss: 0.0249 - acc: 0.9916 - va
Epoch 9/12
54000/54000 [==============================] - 5s 98us/sample - loss: 0.0205 - acc: 0.9929 - va
Epoch 10/12
54000/54000 [==============================] - 5s 98us/sample - loss: 0.0174 - acc: 0.9941 - va
Epoch 11/12
54000/54000 [==============================] - 5s 97us/sample - loss: 0.0174 - acc: 0.9941 - va
Epoch 12/12
54000/54000 [==============================] - 5s 98us/sample - loss: 0.0143 - acc: 0.9951 - va
```

In [83]: `draw_learning_curve(model_graph)`


Model Accuracy
train
test
Accuracy
0.99
0.98
0.97
0.96
0.95
0.94
0.93
0
2
4
6
8
10
Epoch


Model loss
train
test
Loss
0.25
0.20
0.15
0.10
0.05
0
2
4
6
8
10
Epoch

```
In [84]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

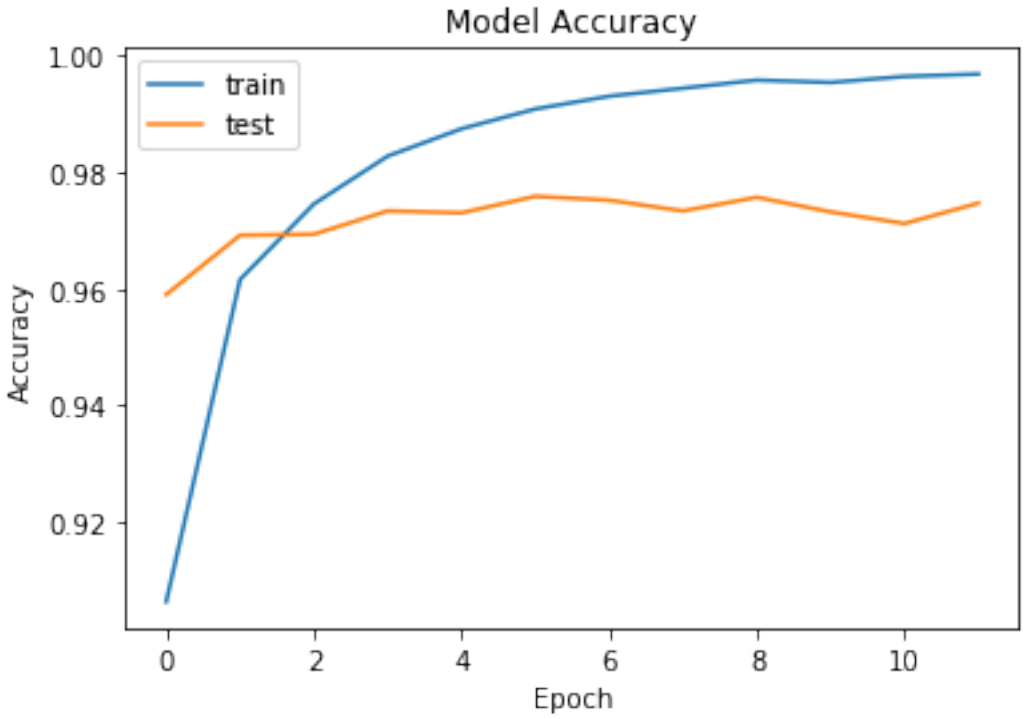
```
60000/60000 [==============================] - 3s 53us/sample - loss: 0.0210 - acc: 0.9944
Train accuracy: 0.99445
10000/10000 [==============================] - 1s 53us/sample - loss: 0.1327 - acc: 0.9715
Test accuracy: 0.9715
```

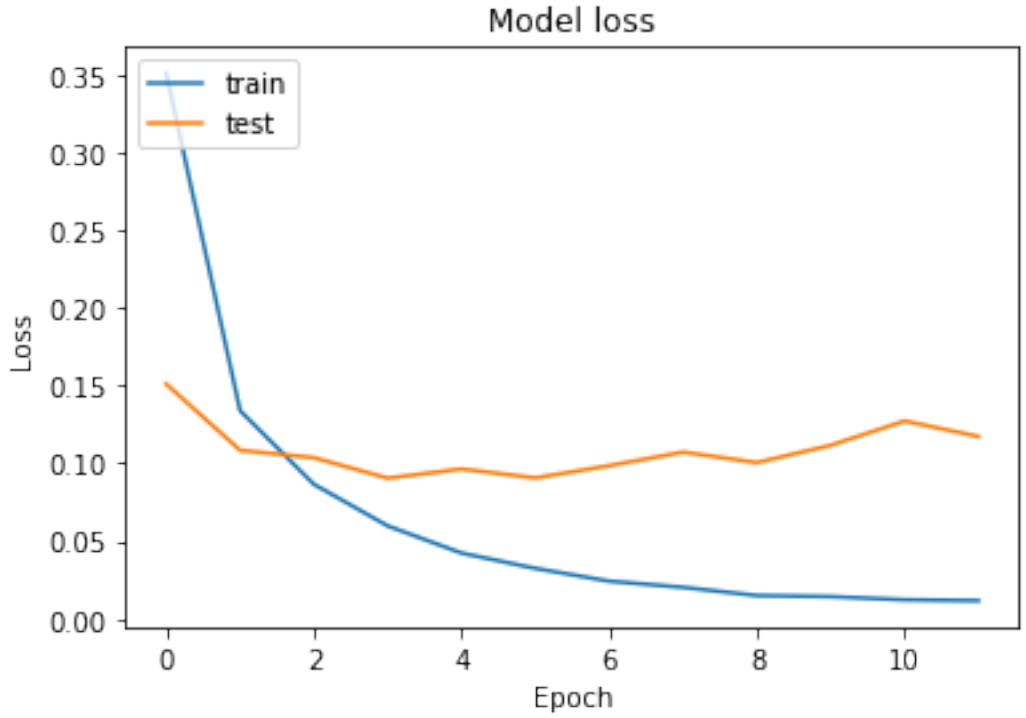
0.7.4 Sigmoid Activation

```
In [85]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l2=0.01))
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 6s 110us/sample - loss: 0.3511 - acc: 0.9064 - v
Epoch 2/12
54000/54000 [==============================] - 5s 100us/sample - loss: 0.1339 - acc: 0.9616 - v
Epoch 3/12
54000/54000 [==============================] - 5s 100us/sample - loss: 0.0865 - acc: 0.9745 - v
Epoch 4/12
54000/54000 [==============================] - 5s 99us/sample - loss: 0.0597 - acc: 0.9827 - va
Epoch 5/12
54000/54000 [==============================] - 5s 98us/sample - loss: 0.0422 - acc: 0.9874 - va
Epoch 6/12
54000/54000 [==============================] - 5s 98us/sample - loss: 0.0324 - acc: 0.9908 - va
Epoch 7/12
54000/54000 [==============================] - 5s 98us/sample - loss: 0.0241 - acc: 0.9930 - va
Epoch 8/12
54000/54000 [==============================] - 5s 99us/sample - loss: 0.0202 - acc: 0.9944 - va
Epoch 9/12
54000/54000 [==============================] - 5s 100us/sample - loss: 0.0149 - acc: 0.9957 - v
Epoch 10/12
54000/54000 [==============================] - 5s 99us/sample - loss: 0.0142 - acc: 0.9953 - va
Epoch 11/12
54000/54000 [==============================] - 5s 99us/sample - loss: 0.0122 - acc: 0.9964 - va
Epoch 12/12
54000/54000 [==============================] - 5s 100us/sample - loss: 0.0116 - acc: 0.9968 - v
```

```
In [86]: draw_learning_curve(model_graph)
```





```
In [87]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 3s 53us/sample - loss: 0.0187 - acc: 0.9956
Train accuracy: 0.9956167
10000/10000 [==============================] - 1s 54us/sample - loss: 0.1175 - acc: 0.9702
Test accuracy: 0.9702
```

0.8 Batch Normalization + L1 Regularization

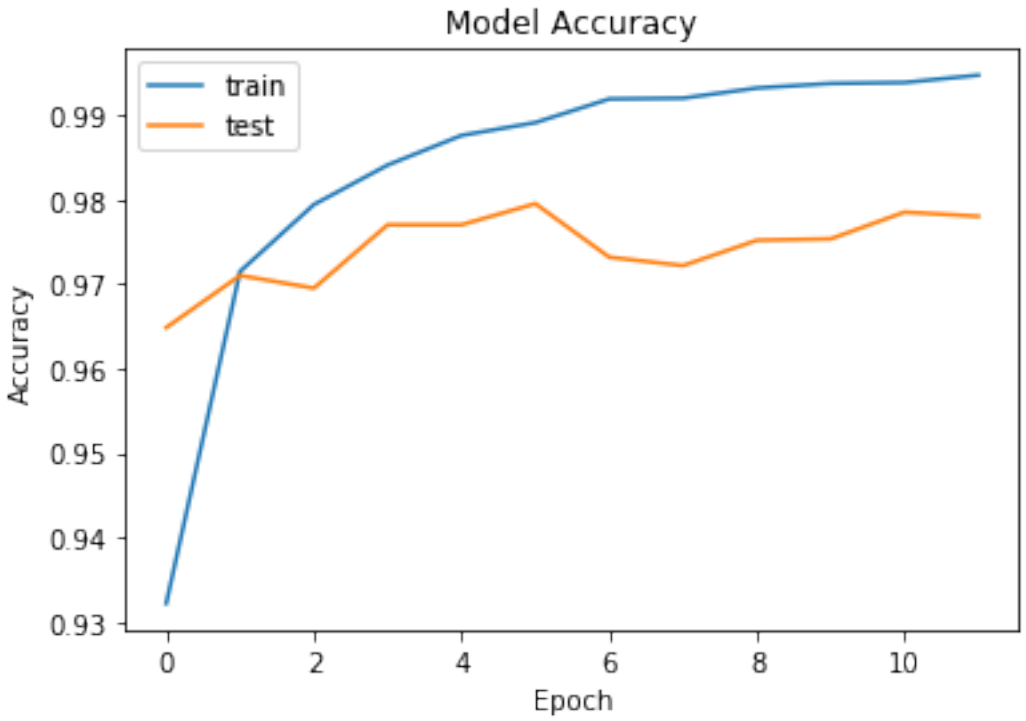
0.8.1 ReLu Activation

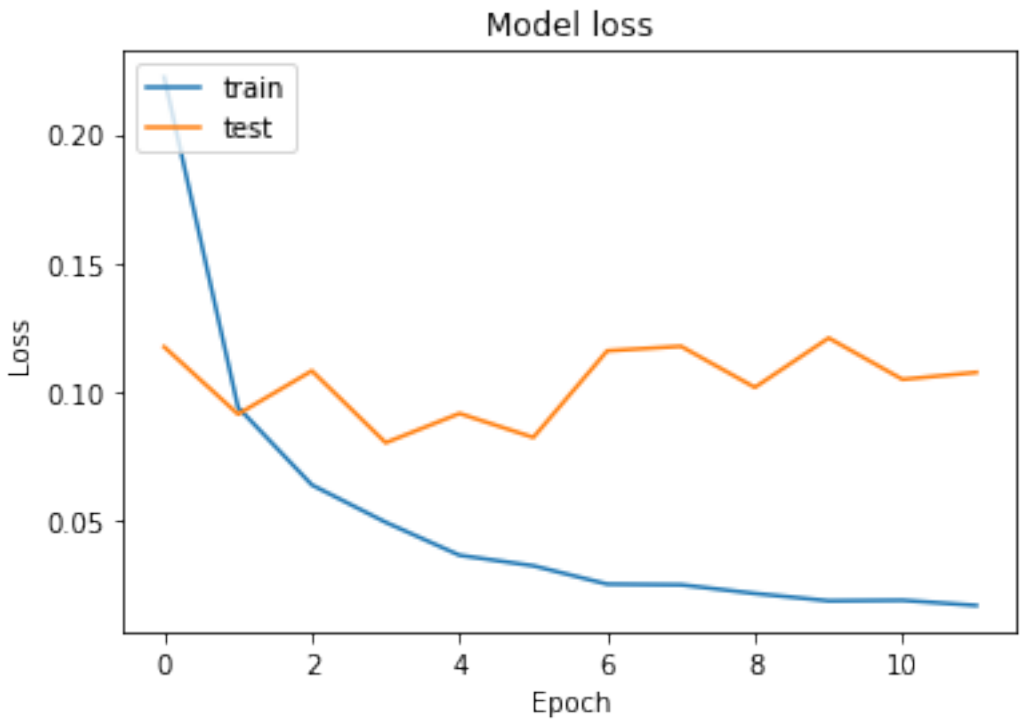
```
In [88]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l1=0.01))
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 6s 112us/sample - loss: 0.2222 - acc: 0.9322 - v
Epoch 2/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.0937 - acc: 0.9715 - v
Epoch 3/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.0636 - acc: 0.9794 - v
Epoch 4/12
54000/54000 [==============================] - 5s 100us/sample - loss: 0.0490 - acc: 0.9841 - v
Epoch 5/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.0362 - acc: 0.9876 - v
Epoch 6/12
54000/54000 [==============================] - 5s 102us/sample - loss: 0.0321 - acc: 0.9891 - v
Epoch 7/12
54000/54000 [==============================] - 6s 102us/sample - loss: 0.0250 - acc: 0.9919 - v
Epoch 8/12
54000/54000 [==============================] - 5s 102us/sample - loss: 0.0247 - acc: 0.9920 - v
Epoch 9/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.0213 - acc: 0.9932 - v
Epoch 10/12
54000/54000 [==============================] - 5s 102us/sample - loss: 0.0185 - acc: 0.9937 - v
```

```
Epoch 11/12
54000/54000 [==============================] - 5s 100us/sample - loss: 0.0187 - acc: 0.9938 - v
Epoch 12/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.0166 - acc: 0.9947 - v
```

```
In [89]: draw_learning_curve(model_graph)
```






```
In [90]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 3s 57us/sample - loss: 0.0221 - acc: 0.9943
Train accuracy: 0.9942833
10000/10000 [==============================] - 1s 54us/sample - loss: 0.1484 - acc: 0.9727
Test accuracy: 0.9727
```

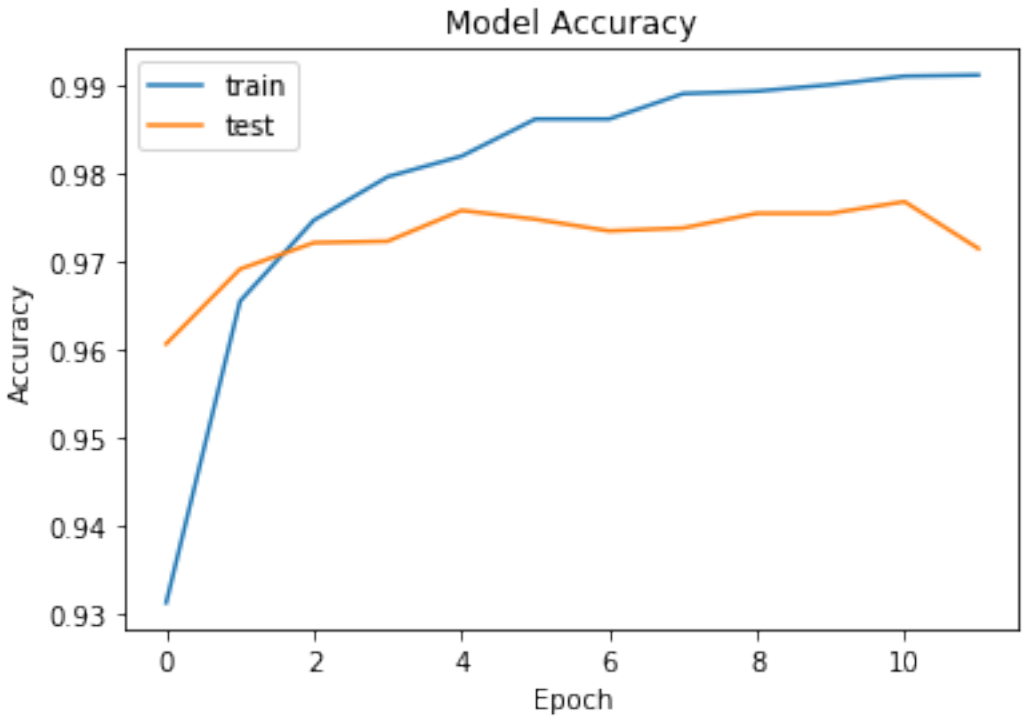
0.8.2 Leaky ReLu Activation

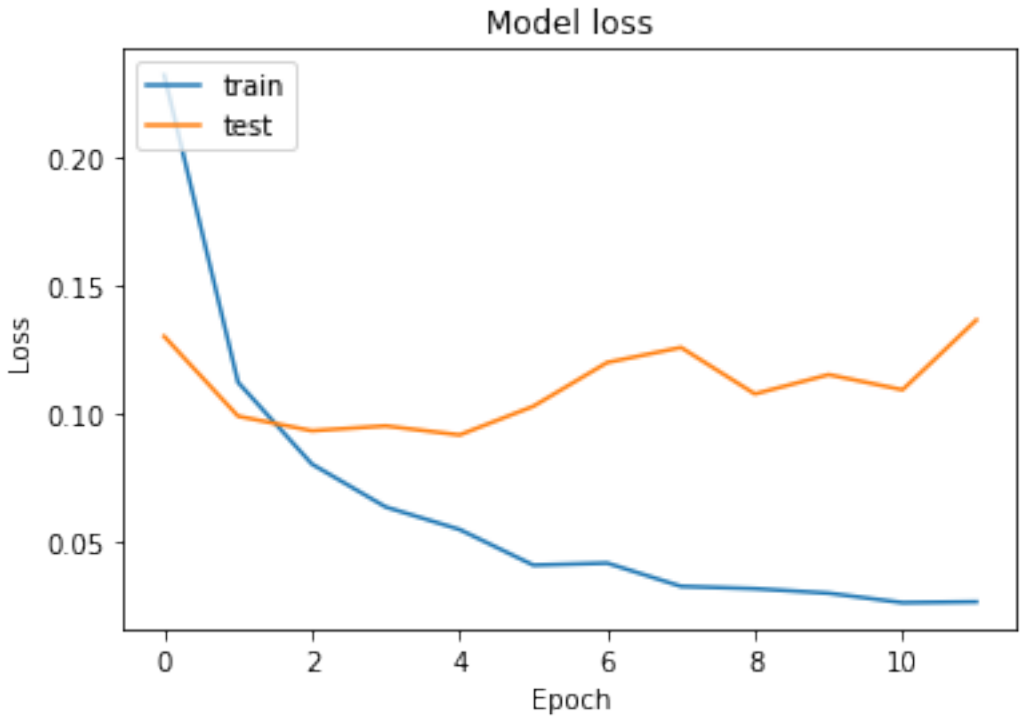
```
In [91]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l1=0.01))
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.leaky_relu))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
```

```
54000/54000 [==============================] - 6s 112us/sample - loss: 0.2318 - acc: 0.9312 - v
Epoch 2/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.1121 - acc: 0.9655 - v
Epoch 3/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.0803 - acc: 0.9747 - v
Epoch 4/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.0635 - acc: 0.9797 - v
Epoch 5/12
54000/54000 [==============================] - 5s 102us/sample - loss: 0.0548 - acc: 0.9820 - v
Epoch 6/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.0409 - acc: 0.9862 - v
Epoch 7/12
54000/54000 [==============================] - 5s 102us/sample - loss: 0.0417 - acc: 0.9862 - v
Epoch 8/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.0326 - acc: 0.9891 - v
Epoch 9/12
54000/54000 [==============================] - 5s 102us/sample - loss: 0.0317 - acc: 0.9894 - v
Epoch 10/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.0299 - acc: 0.9901 - v
Epoch 11/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.0263 - acc: 0.9911 - v
Epoch 12/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.0266 - acc: 0.9912 - v
```

```
In [92]: draw_learning_curve(model_graph)
```






```
In [93]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

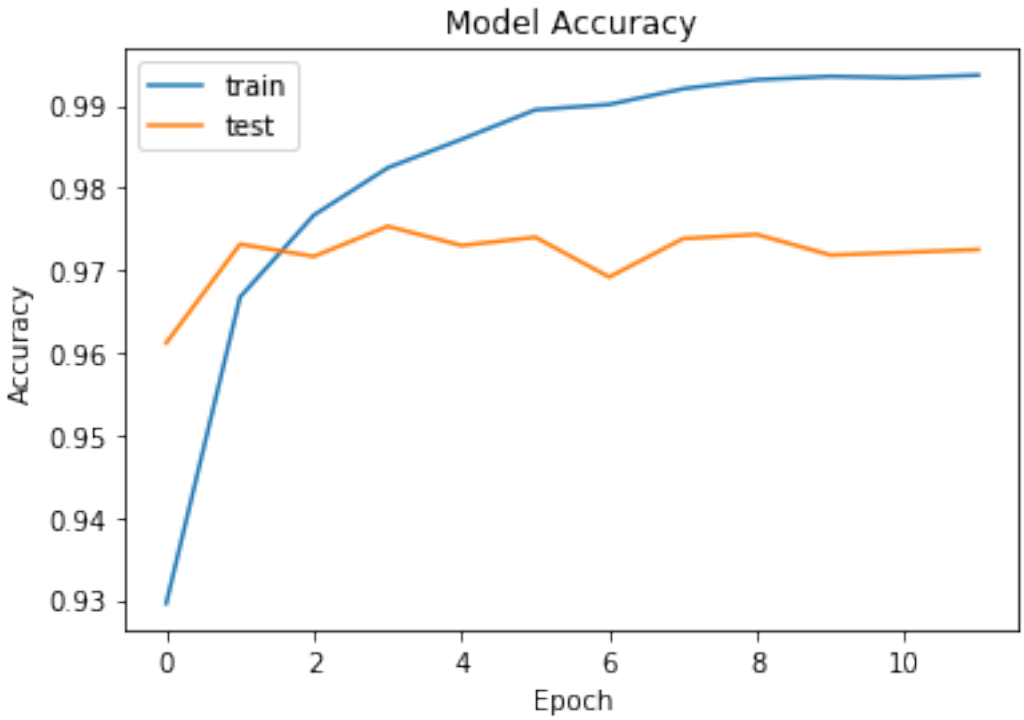
```
60000/60000 [==============================] - 3s 56us/sample - loss: 0.0379 - acc: 0.9894
Train accuracy: 0.98936665
10000/10000 [==============================] - ETA: 0s - loss: 0.1327 - acc: 0.970 - 1s 54us/sa
Test accuracy: 0.97
```

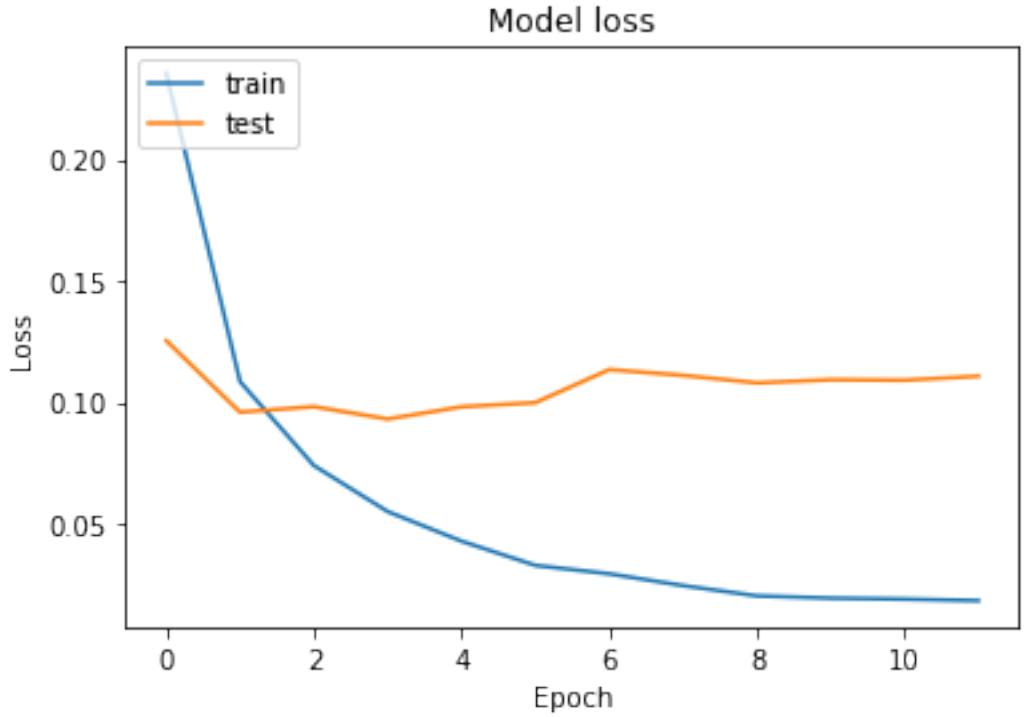
0.8.3 Tanh Activation

```
In [94]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l1=0.01))
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.tanh))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 6s 113us/sample - loss: 0.2354 - acc: 0.9296 - v
Epoch 2/12
54000/54000 [==============================] - 6s 102us/sample - loss: 0.1085 - acc: 0.9667 - v
Epoch 3/12
54000/54000 [==============================] - 5s 102us/sample - loss: 0.0739 - acc: 0.9767 - v
Epoch 4/12
54000/54000 [==============================] - 5s 102us/sample - loss: 0.0550 - acc: 0.9824 - v
Epoch 5/12
54000/54000 [==============================] - 6s 102us/sample - loss: 0.0428 - acc: 0.9859 - v
Epoch 6/12
54000/54000 [==============================] - 6s 102us/sample - loss: 0.0328 - acc: 0.9895 - v
Epoch 7/12
54000/54000 [==============================] - 5s 102us/sample - loss: 0.0294 - acc: 0.9901 - v
Epoch 8/12
54000/54000 [==============================] - 6s 102us/sample - loss: 0.0245 - acc: 0.9920 - v
Epoch 9/12
54000/54000 [==============================] - 6s 102us/sample - loss: 0.0203 - acc: 0.9931 - v
Epoch 10/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.0194 - acc: 0.9935 - v
Epoch 11/12
54000/54000 [==============================] - 6s 103us/sample - loss: 0.0190 - acc: 0.9934 - v
Epoch 12/12
54000/54000 [==============================] - 5s 101us/sample - loss: 0.0183 - acc: 0.9937 - v
```

```
In [95]: draw_learning_curve(model_graph)
```


Model Accuracy
train
test
Accuracy
0.99
0.98
0.97
0.96
0.95
0.94
0.93
0
2
4
6
8
10
Epoch


Model loss
train
test
Loss
0.20
0.15
0.10
0.05
0
2
4
6
8
10
Epoch

```
In [96]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

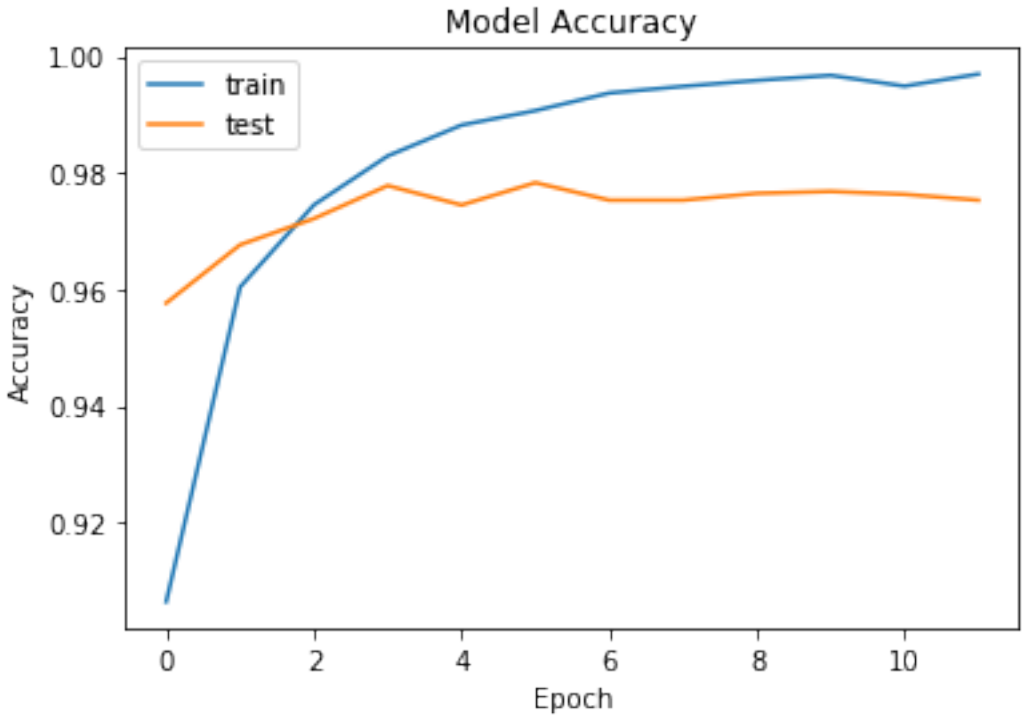
```
60000/60000 [==============================] - 3s 57us/sample - loss: 0.0223 - acc: 0.9937
Train accuracy: 0.99368334
10000/10000 [==============================] - 1s 57us/sample - loss: 0.1371 - acc: 0.9678
Test accuracy: 0.9678
```

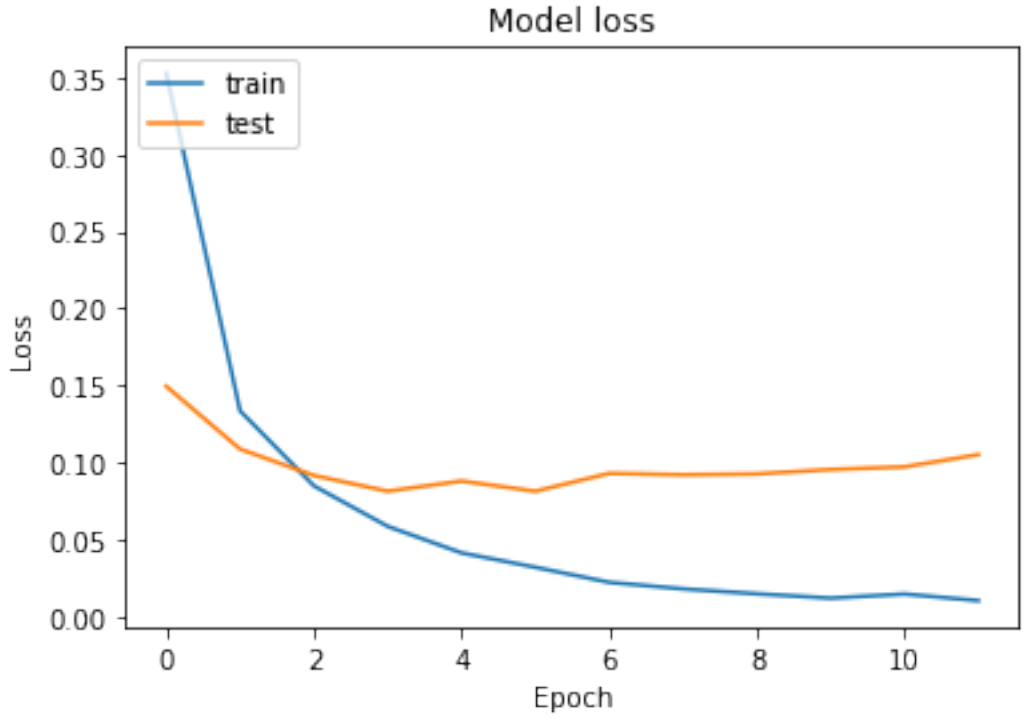
0.8.4 Sigmoid Activation

```
In [97]: model = tf.keras.models.Sequential()
         model.add(tf.keras.layers.ActivityRegularization(l1=0.01))
         model.add(tf.keras.layers.BatchNormalization())
         model.add(tf.keras.layers.Flatten())
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(128, activation = tf.nn.sigmoid))
         model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))
         model.compile(optimizer = 'adam',loss = 'sparse_categorical_crossentropy', metrics =
         model_graph = model.fit(x_train,y_train, epochs = 12, validation_split=0.1)
```

```
Train on 54000 samples, validate on 6000 samples
Epoch 1/12
54000/54000 [==============================] - 6s 114us/sample - loss: 0.3529 - acc: 0.9064 - v
Epoch 2/12
54000/54000 [==============================] - 6s 103us/sample - loss: 0.1334 - acc: 0.9604 - v
Epoch 3/12
54000/54000 [==============================] - 6s 103us/sample - loss: 0.0847 - acc: 0.9745 - v
Epoch 4/12
54000/54000 [==============================] - 6s 103us/sample - loss: 0.0583 - acc: 0.9829 - v
Epoch 5/12
54000/54000 [==============================] - 6s 103us/sample - loss: 0.0411 - acc: 0.9882 - v
Epoch 6/12
54000/54000 [==============================] - 6s 103us/sample - loss: 0.0317 - acc: 0.9907 - v
Epoch 7/12
54000/54000 [==============================] - 6s 103us/sample - loss: 0.0219 - acc: 0.9937 - v
Epoch 8/12
54000/54000 [==============================] - 6s 103us/sample - loss: 0.0177 - acc: 0.9949 - v
Epoch 9/12
54000/54000 [==============================] - 6s 103us/sample - loss: 0.0145 - acc: 0.9959 - v
Epoch 10/12
54000/54000 [==============================] - 6s 104us/sample - loss: 0.0117 - acc: 0.9968 - v
Epoch 11/12
54000/54000 [==============================] - 6s 106us/sample - loss: 0.0143 - acc: 0.9949 - v
Epoch 12/12
54000/54000 [==============================] - 6s 106us/sample - loss: 0.0100 - acc: 0.9970 - v
```

```
In [98]: draw_learning_curve(model_graph)
```





```
In [99]: val_loss, val_acc = model.evaluate(x_train, y_train)
         print("Train accuracy:", val_acc)
         val_loss, val_acc = model.evaluate(x_test, y_test)
         print("Test accuracy:", val_acc)
```

```
60000/60000 [==============================] - 4s 60us/sample - loss: 0.0168 - acc: 0.9959
Train accuracy: 0.99588335
10000/10000 [==============================] - 1s 61us/sample - loss: 0.1238 - acc: 0.9703
Test accuracy: 0.9703
```