```
In [0]:    import warnings
           warnings.filterwarnings('ignore')
```

```
In [0]:    # Credits: https://github.com/keras-team/keras/blob/master/examples/mnist_cnn.py


           from __future__ import print_function
           import keras
           from PIL import Image
           import numpy as np
           from keras.datasets import mnist
           from keras.models import Sequential
           from keras.layers import Dense, Dropout, Flatten
           from keras.layers import Conv2D, MaxPooling2D
           from keras import backend as K
           import matplotlib.pyplot as plt
```

```
In [4]:    from prettytable import PrettyTable
           table = PrettyTable()
           table.field_names= ["No of Conv Layers","Kernel Size","Accuracy", "Loss"]
           print(table)
```

```
+-------------------+-------------+----------+------+
| No of Conv Layers | Kernel Size | Accuracy | Loss |
+-------------------+-------------+----------+------+
+-------------------+-------------+----------+------+
```

```
In [22]:   batch_size = 128
           num_classes = 10
           epochs = 12

           # input image dimensions
           img_rows, img_cols = 28, 28

           # the data, split between train and test sets
           (x_train, y_train), (x_test, y_test) = mnist.load_data()


           if K.image_data_format() == 'channels_first':
               x_train = x_train.reshape(x_train.shape[0], 1, img_rows, img_cols)
               x_test = x_test.reshape(x_test.shape[0], 1, img_rows, img_cols)
               input_shape = (1, img_rows, img_cols)
           else:
               x_train = x_train.reshape(x_train.shape[0], img_rows, img_cols, 1)
               x_test = x_test.reshape(x_test.shape[0], img_rows, img_cols, 1)
               input_shape = (img_rows, img_cols, 1)

           x_train = x_train.astype('float32')
           x_test = x_test.astype('float32')
           x_train /= 255
           x_test /= 255
           print('x_train shape:', x_train.shape)
           print(x_train.shape[0], 'train samples')
           print(x_test.shape[0], 'test samples')
```

```
x_train shape: (60000, 28, 28, 1)
60000 train samples
10000 test samples
```

```python
# convert class vectors to binary class matrices
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)

model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3),
                 activation='relu',
                 input_shape=input_shape))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))

model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adadelta(),
              metrics=['accuracy'])
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:66: The name tf.get_default_graph is deprecated. Please use tf.compat.v
1.get_default_graph instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:541: The name tf.placeholder is deprecated. Please use tf.compat.v1.pla
ceholder instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:4432: The name tf.random_uniform is deprecated. Please use tf.random.un
iform instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:4267: The name tf.nn.max_pool is deprecated. Please use tf.nn.max_pool2
d instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:148: The name tf.placeholder_with_default is deprecated. Please use tf.
compat.v1.placeholder_with_default instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:3733: calling dropout (from tensorflow.python.ops.nn_ops) with keep_pro
b is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/optimizers.py:793: The name tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer
instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:3576: The name tf.log is deprecated. Please use tf.math.log instead.
```

```
In [9]:    model.summary()
```

```
Model: "sequential_1"
_____
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 26, 26, 32)        320
_____
conv2d_2 (Conv2D)            (None, 24, 24, 64)        18496
_____
max_pooling2d_1 (MaxPooling2 (None, 12, 12, 64)        0
_____
dropout_1 (Dropout)          (None, 12, 12, 64)        0
_____
flatten_1 (Flatten)          (None, 9216)              0
_____
dense_1 (Dense)              (None, 128)               1179776
_____
dropout_2 (Dropout)          (None, 128)               0
_____
dense_2 (Dense)              (None, 10)                1290
=================================================================
Total params: 1,199,882
Trainable params: 1,199,882
Non-trainable params: 0
_____
```

```
In [10]:   history = model.fit(x_train, y_train,
                     batch_size=batch_size,
                     epochs=epochs,
                     verbose=0,
                     validation_data=(x_test, y_test))
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_core/python/ops/math_grad.py:1424: where (from tensorflow.python.ops.array_ops) is deprecated an
d will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:1033: The name tf.assign_add is deprecated. Please use tf.compat.v1.ass
ign_add instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:1020: The name tf.assign is deprecated. Please use tf.compat.v1.assign
instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:3005: The name tf.Session is deprecated. Please use tf.compat.v1.Sessio
n instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:190: The name tf.get_default_session is deprecated. Please use tf.compa
t.v1.get_default_session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:197: The name tf.ConfigProto is deprecated. Please use tf.compat.v1.Con
figProto instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:207: The name tf.global_variables is deprecated. Please use tf.compat.v
1.global_variables instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:216: The name tf.is_variable_initialized is deprecated. Please use tf.c
ompat.v1.is_variable_initialized instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:223: The name tf.variables_initializer is deprecated. Please use tf.com
pat.v1.variables_initializer instead.
```
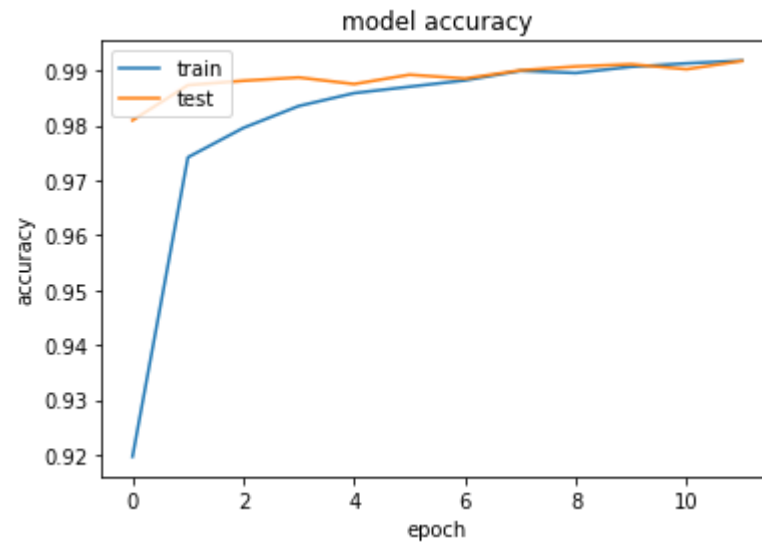
```
In [11]: score = model.evaluate(x_test, y_test, verbose=0)
         print('Test loss:', score[0])
         print('Test accuracy:', score[1])
```

```
Test loss: 0.02629823839784285
Test accuracy: 0.9917
```

```
In [20]: # summarize history for accuracy
         plt.plot(history.history['acc'])
         plt.plot(history.history['val_acc'])
         plt.title('model accuracy')
         plt.ylabel('accuracy')
         plt.xlabel('epoch')
         plt.legend(['train', 'test'], loc='upper left')
         plt.show()
```



```
In [21]: # summarize history for loss
         plt.plot(history.history['loss'])
         plt.plot(history.history['val_loss'])
         plt.title('model loss')
         plt.ylabel('loss')
         plt.xlabel('epoch')
         plt.legend(['train', 'test'], loc='upper left')
         plt.show()
```

```
In [11]: table.add_row([2,'(3,3)','0.99','0.02'])
         print(table)

         +-------------------+-------------+----------+------+
         | No of Conv Layers | Kernel Size | Accuracy | Loss |
         +-------------------+-------------+----------+------+
         |         2         |    (3,3)    |   0.99   | 0.02 |
         +-------------------+-------------+----------+------+
```

```
In [0]: # Model 2 kernel size = (5,5)
```

```
In [0]: # convert class vectors to binary class matrices
        y_train = keras.utils.to_categorical(y_train, num_classes)
        y_test = keras.utils.to_categorical(y_test, num_classes)

        model = Sequential()
        model.add(Conv2D(32, kernel_size=(5, 5),
                         activation='relu',
                         input_shape=input_shape))
        model.add(Conv2D(64, (5, 5), activation='relu'))
        model.add(MaxPooling2D(pool_size=(2, 2)))
        model.add(Dropout(0.25))
        model.add(Flatten())
        model.add(Dense(128, activation='relu'))
        model.add(Dropout(0.5))
        model.add(Dense(num_classes, activation='softmax'))

        model.compile(loss=keras.losses.categorical_crossentropy,
                      optimizer=keras.optimizers.Adadelta(),
                      metrics=['accuracy'])
```

```
In [30]: model.summary()

         Model: "sequential_3"

         _____
         Layer (type)                Output Shape              Param #
         ===============================================================
         conv2d_5 (Conv2D)           (None, 24, 24, 32)        832
         _____
         conv2d_6 (Conv2D)           (None, 20, 20, 64)        51264
         _____
         max_pooling2d_3 (MaxPooling2 (None, 10, 10, 64)       0
         _____
         dropout_5 (Dropout)         (None, 10, 10, 64)        0
         _____
         flatten_3 (Flatten)         (None, 6400)              0
         _____
         dense_5 (Dense)             (None, 128)               819328
         _____
         dropout_6 (Dropout)         (None, 128)               0
         _____
         dense_6 (Dense)             (None, 10)                1290
         ===============================================================
         Total params: 872,714
         Trainable params: 872,714
         Non-trainable params: 0
         _____
```

```
In [0]: history = model.fit(x_train, y_train,
                  batch_size=batch_size,
                  epochs=epochs,
                  verbose=0,
                  validation_data=(x_test, y_test))
```
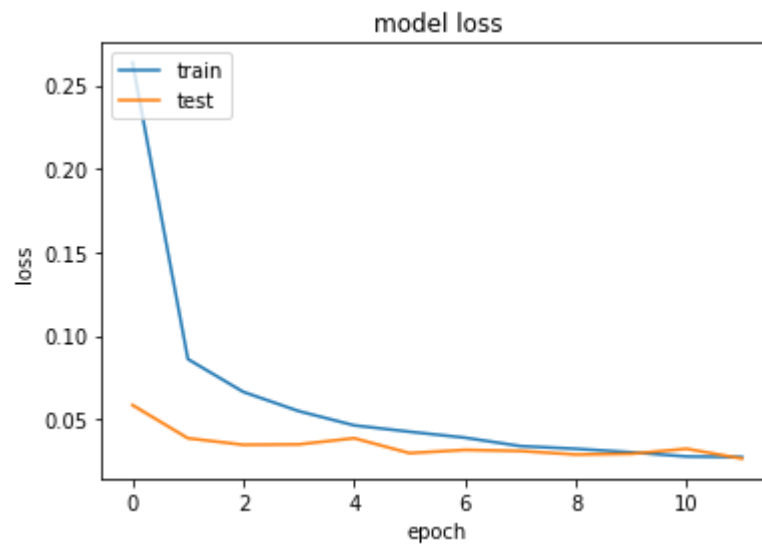
```
In [32]: score = model.evaluate(x_test, y_test, verbose=0)
         print('Test loss:', score[0])
         print('Test accuracy:', score[1])
```

Test loss: 0.021633764794276793
Test accuracy: 0.9937

```
In [33]: # summarize history for accuracy
         plt.plot(history.history['acc'])
         plt.plot(history.history['val_acc'])
         plt.title('model accuracy')
         plt.ylabel('accuracy')
         plt.xlabel('epoch')
         plt.legend(['train', 'test'], loc='upper left')
         plt.show()
         # summarize history for loss
         plt.plot(history.history['loss'])
         plt.plot(history.history['val_loss'])
         plt.title('model loss')
         plt.ylabel('loss')
         plt.xlabel('epoch')
         plt.legend(['train', 'test'], loc='upper left')
         plt.show()
```
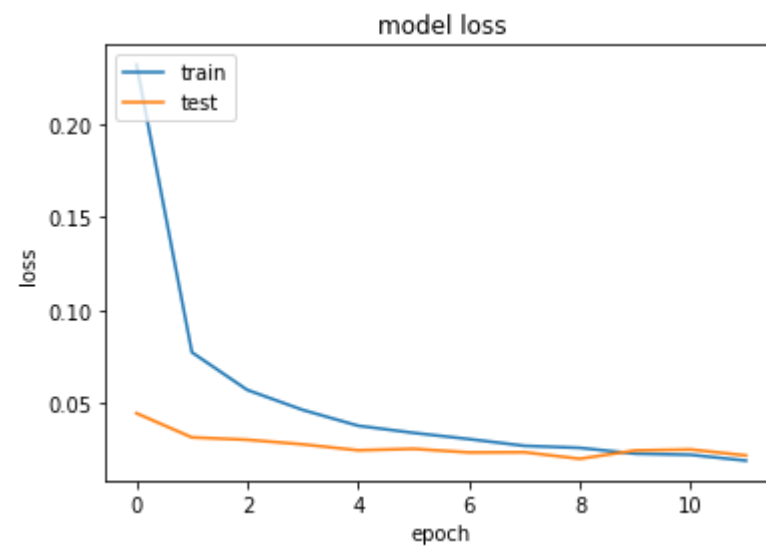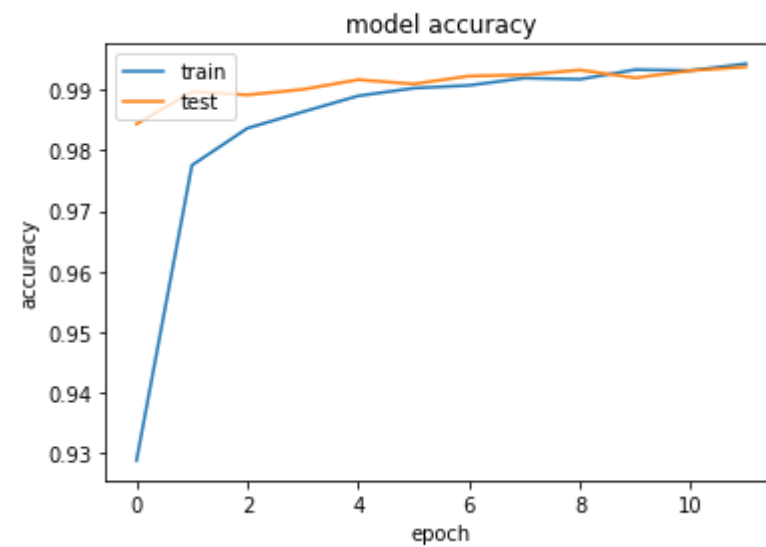
```
In [12]:  table.add_row([2,'(5,5)','0.99','0.02'])
          print(table)
```

```
+--------------------+-------------+----------+------+
| No of Conv Layers  | Kernel Size | Accuracy | Loss |
+--------------------+-------------+----------+------+
|          2         |    (3,3)    |   0.99   | 0.02 |
|          2         |    (5,5)    |   0.99   | 0.02 |
+--------------------+-------------+----------+------+
```

```python
In [6]:  # convert class vectors to binary class matrices
         y_train = keras.utils.to_categorical(y_train, num_classes)
         y_test = keras.utils.to_categorical(y_test, num_classes)

         model = Sequential()
         model.add(Conv2D(32, kernel_size=(7, 7),
                          activation='relu',
                          input_shape=input_shape))
         model.add(Conv2D(64, (7, 7), activation='relu'))
         model.add(MaxPooling2D(pool_size=(2, 2)))
         model.add(Dropout(0.25))
         model.add(Flatten())
         model.add(Dense(128, activation='relu'))
         model.add(Dropout(0.5))
         model.add(Dense(num_classes, activation='softmax'))

         model.compile(loss=keras.losses.categorical_crossentropy,
                       optimizer=keras.optimizers.Adadelta(),
                       metrics=['accuracy'])
```

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:66: The name tf.get_default_graph is deprecated. Please use tf.compat.v1.get_default_graph instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:541: The name tf.placeholder is deprecated. Please use tf.compat.v1.placeholder instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:4432: The name tf.random_uniform is deprecated. Please use tf.random.uniform instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:4267: The name tf.nn.max_pool is deprecated. Please use tf.nn.max_pool2d instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:148: The name tf.placeholder_with_default is deprecated. Please use tf.compat.v1.placeholder_with_default instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:3733: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/optimizers.py:793: The name tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:3576: The name tf.log is deprecated. Please use tf.math.log instead.

```
In [7]:  model.summary()

Model: "sequential_1"
_____
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 22, 22, 32)        1600
_____
conv2d_2 (Conv2D)            (None, 16, 16, 64)        100416
_____
max_pooling2d_1 (MaxPooling2 (None, 8, 8, 64)          0
_____
dropout_1 (Dropout)          (None, 8, 8, 64)          0
_____
flatten_1 (Flatten)          (None, 4096)              0
_____
dense_1 (Dense)              (None, 128)               524416
_____
dropout_2 (Dropout)          (None, 128)               0
_____
dense_2 (Dense)              (None, 10)                1290
=================================================================
Total params: 627,722
Trainable params: 627,722
Non-trainable params: 0
_____
```

```
In [8]:  history = model.fit(x_train, y_train,
            batch_size=batch_size,
            epochs=epochs,
            verbose=0,
            validation_data=(x_test, y_test))
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_core/python/ops/math_grad.py:1424: where (from tensorflow.python.ops.array_ops) is deprecated an
d will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:1033: The name tf.assign_add is deprecated. Please use tf.compat.v1.ass
ign_add instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:1020: The name tf.assign is deprecated. Please use tf.compat.v1.assign
instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:3005: The name tf.Session is deprecated. Please use tf.compat.v1.Sessio
n instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:190: The name tf.get_default_session is deprecated. Please use tf.compa
t.v1.get_default_session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:197: The name tf.ConfigProto is deprecated. Please use tf.compat.v1.Con
figProto instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:207: The name tf.global_variables is deprecated. Please use tf.compat.v
1.global_variables instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:216: The name tf.is_variable_initialized is deprecated. Please use tf.c
ompat.v1.is_variable_initialized instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:223: The name tf.variables_initializer is deprecated. Please use tf.com
pat.v1.variables_initializer instead.
```

```
In [9]:  score = model.evaluate(x_test, y_test, verbose=0)
         print('Test loss:', score[0])
         print('Test accuracy:', score[1])
```

```
Test loss: 0.016189836317225672
Test accuracy: 0.9938
```

```
In [10]:  # summarize history for accuracy
          plt.plot(history.history['acc'])
          plt.plot(history.history['val_acc'])
          plt.title('model accuracy')
          plt.ylabel('accuracy')
          plt.xlabel('epoch')
          plt.legend(['train', 'test'], loc='upper left')
          plt.show()
          # summarize history for loss
          plt.plot(history.history['loss'])
          plt.plot(history.history['val_loss'])
          plt.title('model loss')
          plt.ylabel('loss')
          plt.xlabel('epoch')
          plt.legend(['train', 'test'], loc='upper left')
          plt.show()
```
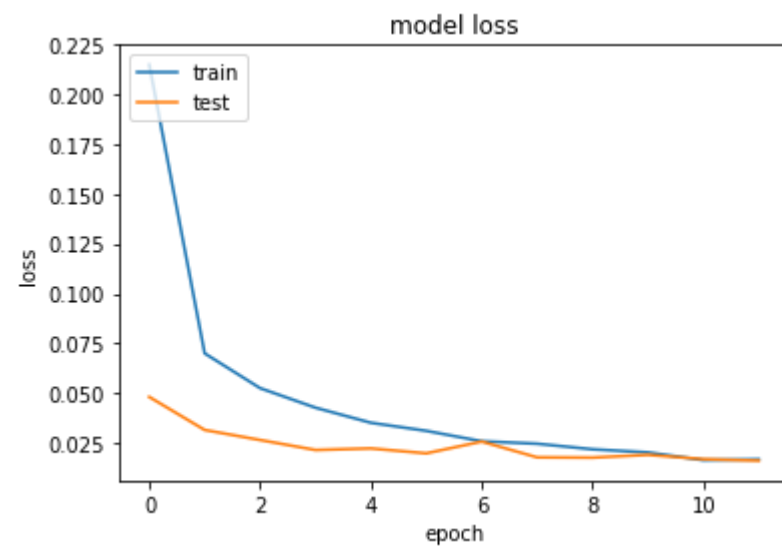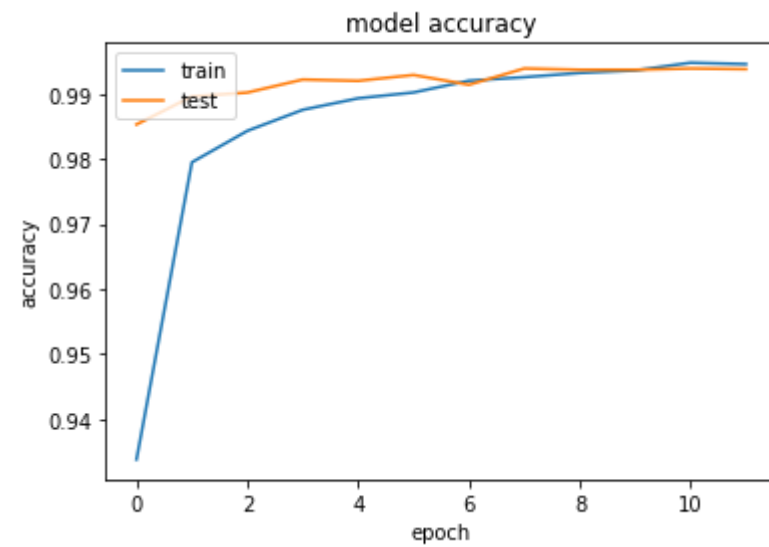
```
In [13]: table.add_row([2,'(7,7)','0.99','0.016'])
         print(table)
```

```
+-------------------+-------------+----------+-------+
| No of Conv Layers | Kernel Size | Accuracy |  Loss |
+-------------------+-------------+----------+-------+
|         2         |    (3,3)    |   0.99   |  0.02 |
|         2         |    (5,5)    |   0.99   |  0.02 |
|         2         |    (7,7)    |   0.99   | 0.016 |
+-------------------+-------------+----------+-------+
```

```python
In [0]: # convert class vectors to binary class matrices
        y_train = keras.utils.to_categorical(y_train, num_classes)
        y_test = keras.utils.to_categorical(y_test, num_classes)

        model = Sequential()
        model.add(Conv2D(64, kernel_size=(5, 5),
                         activation='relu',
                         input_shape=input_shape))
        model.add(Conv2D(32, (5, 5), activation='relu'))
        model.add(MaxPooling2D(pool_size=(2, 2)))
        model.add(Dropout(0.25))
        model.add(Conv2D(32, kernel_size=(3, 3),
                         activation='relu',
                         input_shape=input_shape))
        model.add(Conv2D(16, (3, 3), activation='relu'))
        model.add(MaxPooling2D(pool_size=(2, 2)))
        model.add(Dropout(0.25))
        model.add(Flatten())
        model.add(Dense(128, activation='relu'))
        model.add(Dropout(0.5))
        model.add(Dense(num_classes, activation='softmax'))

        model.compile(loss=keras.losses.categorical_crossentropy,
                      optimizer=keras.optimizers.Adadelta(),
                      metrics=['accuracy'])
```

```
In [16]: model.summary()
```

```
Model: "sequential_2"
_____
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 24, 24, 64)        1664
_____
conv2d_4 (Conv2D)            (None, 20, 20, 32)        51232
_____
max_pooling2d_2 (MaxPooling2 (None, 10, 10, 32)        0
_____
dropout_3 (Dropout)          (None, 10, 10, 32)        0
_____
conv2d_5 (Conv2D)            (None, 8, 8, 32)          9248
_____
conv2d_6 (Conv2D)            (None, 6, 6, 16)          4624
_____
max_pooling2d_3 (MaxPooling2 (None, 3, 3, 16)          0
_____
dropout_4 (Dropout)          (None, 3, 3, 16)          0
_____
flatten_2 (Flatten)          (None, 144)               0
_____
dense_3 (Dense)              (None, 128)               18560
_____
dropout_5 (Dropout)          (None, 128)               0
_____
dense_4 (Dense)              (None, 10)                1290
=================================================================
Total params: 86,618
Trainable params: 86,618
Non-trainable params: 0
_____
```

```
In [0]: history = model.fit(x_train, y_train,
                batch_size=batch_size,
                epochs=epochs,
                verbose=0,
                validation_data=(x_test, y_test))
```

```
In [18]: score = model.evaluate(x_test, y_test, verbose=0)
         print('Test loss:', score[0])
         print('Test accuracy:', score[1])
```

```
Test loss: 0.01816003255521646
Test accuracy: 0.9936
```

```
# summarize history for accuracy
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
# summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```
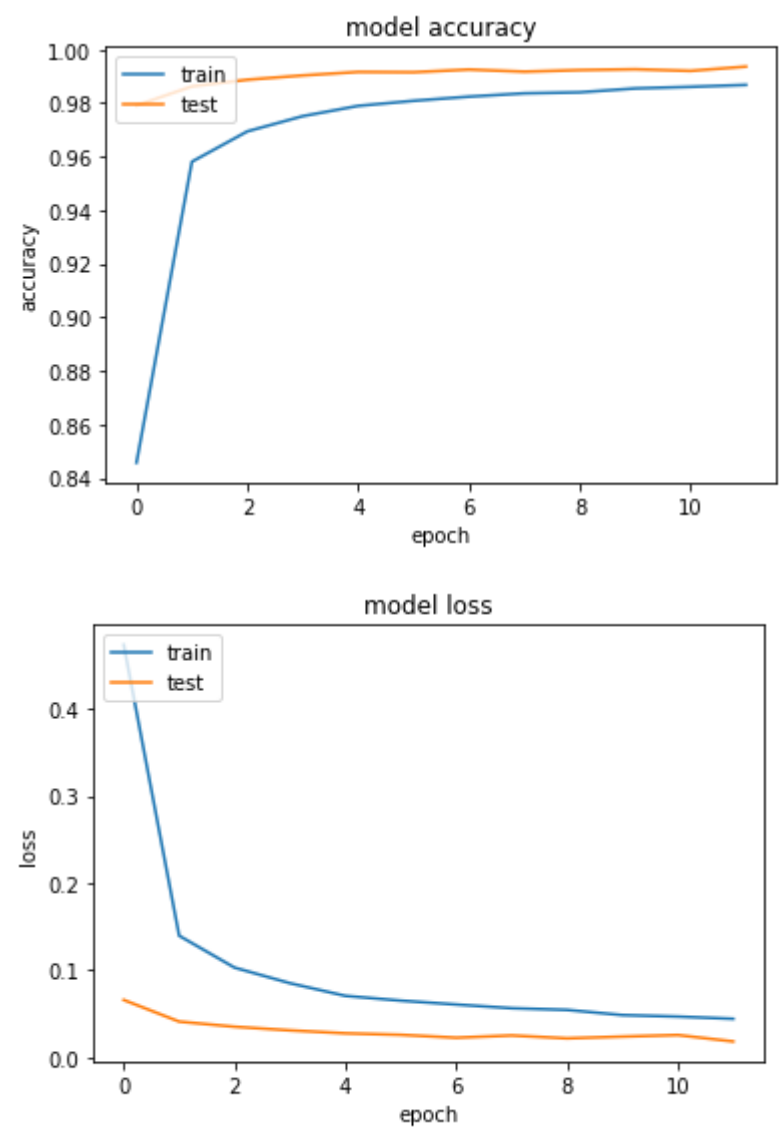
```
table.add_row([4,'(3,3),(5,5)','0.99','0.018'])
print(table)
```

```
+------------------+-------------+----------+-------+
| No of Conv Layers | Kernel Size | Accuracy |  Loss |
+------------------+-------------+----------+-------+
|        2         |    (3,3)    |   0.99   |  0.02 |
|        2         |    (5,5)    |   0.99   |  0.02 |
|        2         |    (7,7)    |   0.99   | 0.016 |
|        4         | (3,3),(5,5) |   0.99   | 0.018 |
+------------------+-------------+----------+-------+
```

```
In [0]:  # We can clearly infer from this that, for MNIST images the CNN models will be
         # far superior compared to the MLP models
         # Also we see that increase in depth of the model and(or) increase in kernel
         # size increase the feature handling capability of the model, hence better accuracy
```