手寫數字辨識

- 使用演算法: CNN

In [1]:
```
import matplotlib.pyplot as plt
from PIL import Image
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Dropout,Convolution2D,MaxPooling2D,Flatten
from tensorflow.keras.optimizers import Adam
```

In [2]:
```
mnist = tf.keras.datasets.mnist
```

In [3]:
```
(X_train, Y_train), (x_test, y_test) = mnist.load_data()
```

In [4]:
```
X_train = X_train.reshape(-1, 28, 28, 1) / 255.0
x_test = x_test.reshape(-1, 28, 28, 1) / 255.0
```

In [5]:
```
Y_train = tf.keras.utils.to_categorical(Y_train, num_classes=10)
y_test = tf.keras.utils.to_categorical(y_test, num_classes=10)
```

In [6]:
```
#開始建立model
model = Sequential()
```

In [7]:
```
#新增建立一個卷積核處理二維資料
model.add(Convolution2D(
                         input_shape=(28, 28, 1), #資料維度
                         filters=32, #輸出資料維度
                         kernel_size=5, #卷積核大小
                         strides=1, #進行卷積核計算時移動步幅大小
                         padding='same', #設定圖形邊界資料處理策略為補零
                         activation='relu'#設定啟動函數為relu,當輸入值>0則輸出線性函數,
                        ))
```

In [8]:
```
#對卷積核層輸出的空間資料進行最大值池化策略
model.add(MaxPooling2D(
                        pool_size=2, #設定池化視窗的維度
                        strides=2, #設定在做池化石的步幅
                        padding='same',
                       ))
```

In [9]:
```
model.add(Convolution2D(64, 5, strides=1, padding='same', activation='relu'))
```

In [10]:
```
model.add(MaxPooling2D(2, 2, 'same'))
```

In [11]:
```
model.add(Flatten()) #將輸入資料壓平成一維
```

In [12]:
```
model.add(Dense(1024, activation='relu'))#進行全連接標準神經網路
```

In [13]:
```
model.add(Dropout(0.5)) #將神經完按照50%機率隨機失效,避免過擬合情況發生,也加速訓練速度
```

In [14]:
```
model.add(Dense(10, activation='softmax'))
```

In [15]:
```
adam = Adam(lr=1e-4) #設定學習率
```

In [16]:
```
model.compile(optimizer=adam, loss='categorical_crossentropy', metrics=['accuracy'])
```

In [17]:
```
model.fit(X_train, Y_train, batch_size=64, epochs=10, validation_data=(x_test, y_tes
```

```
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
60000/60000 [==============================] - 71s 1ms/sample - loss: 0.3310 - accur
acy: 0.9075 - val_loss: 0.0907 - val_accuracy: 0.9733
Epoch 2/10
60000/60000 [==============================] - 72s 1ms/sample - loss: 0.0931 - accur
acy: 0.9730 - val_loss: 0.0570 - val_accuracy: 0.9816
Epoch 3/10
60000/60000 [==============================] - 75s 1ms/sample - loss: 0.0672 - accur
acy: 0.9794 - val_loss: 0.0377 - val_accuracy: 0.9874
Epoch 4/10
60000/60000 [==============================] - 77s 1ms/sample - loss: 0.0524 - accur
acy: 0.9841 - val_loss: 0.0341 - val_accuracy: 0.9884
Epoch 5/10
60000/60000 [==============================] - 71s 1ms/sample - loss: 0.0427 - accur
acy: 0.9869 - val_loss: 0.0324 - val_accuracy: 0.9888
Epoch 6/10
60000/60000 [==============================] - 72s 1ms/sample - loss: 0.0361 - accur
acy: 0.9888 - val_loss: 0.0262 - val_accuracy: 0.9906
Epoch 7/10
60000/60000 [==============================] - 73s 1ms/sample - loss: 0.0313 - accur
acy: 0.9902 - val_loss: 0.0257 - val_accuracy: 0.9906
Epoch 8/10
60000/60000 [==============================] - 72s 1ms/sample - loss: 0.0286 - accur
acy: 0.9917 - val_loss: 0.0248 - val_accuracy: 0.9910
Epoch 9/10
60000/60000 [==============================] - 72s 1ms/sample - loss: 0.0240 - accur
acy: 0.9927 - val_loss: 0.0236 - val_accuracy: 0.9910
Epoch 10/10
60000/60000 [==============================] - 73s 1ms/sample - loss: 0.0217 - accur
acy: 0.9931 - val_loss: 0.0221 - val_accuracy: 0.9926
```

Out[17]:
```
<tensorflow.python.keras.callbacks.History at 0x2b8ba139348>
```

In [18]:
```
#將訓練出的model儲存
model.save('mnist.CNNmodel')
```

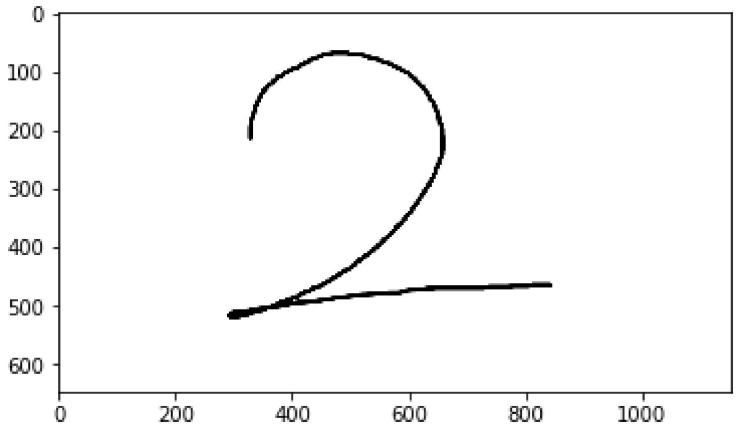
```
WARNING:tensorflow:From C:\Users\NickLin\AppData\Roaming\Python\Python37\site-packag
es\tensorflow_core\python\ops\resource_variable_ops.py:1786: calling BaseResourceVar
iable.__init__ (from tensorflow.python.ops.resource_variable_ops) with constraint is
deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
INFO:tensorflow:Assets written to: mnist.CNNmodel\assets
```

In [19]:

```
#開始進行手寫數字預測
img = Image.open('data\\img\\number2.jpg')
```

In [20]:
```
plt.imshow(img)
```

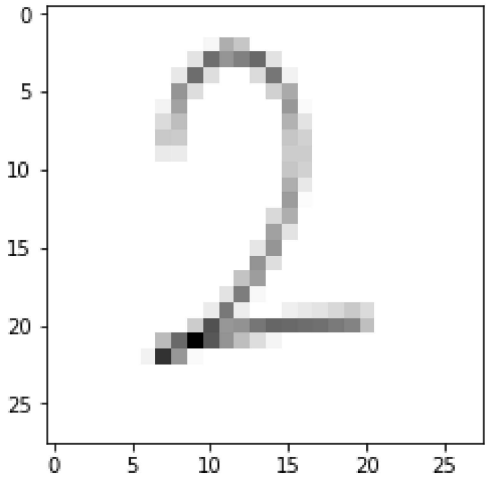
Out[20]: <matplotlib.image.AxesImage at 0x2b8ba376148>



In [21]:
```
image = np.array(img.resize((28, 28)).convert('L'))
```

In [22]:
```
plt.imshow(image, cmap='gray')
```

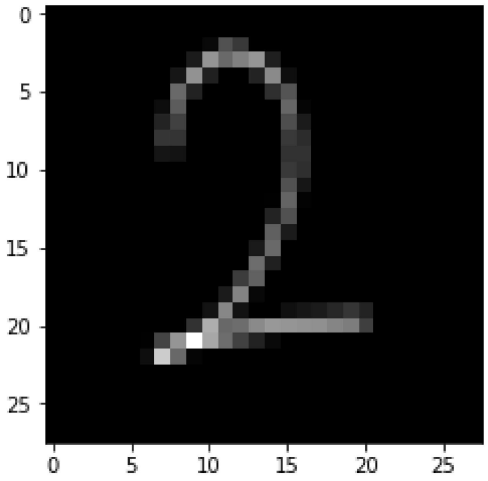
Out[22]: <matplotlib.image.AxesImage at 0x2b8bbd73a88>



In [23]:
```
image = (255 - image) / 255.0
```

In [24]:
```
plt.imshow(image, cmap='gray')
```

Out[24]: <matplotlib.image.AxesImage at 0x2b8bbd42ac8>



In [25]:
```
image = image.reshape((1, 28, 28, 1))
```

In [26]:
```
from tensorflow.keras.models import load_model
model = load_model('mnist.CNNmodel')
```

In [27]:
```
prediction = model.predict_classes(image)
print(prediction)
```

```
[2]
```

In [ ]:

In [ ]: