國立清華大學
National Tsing Hua University

INNOVATION ENGINEERING LAB

IIE Final Project

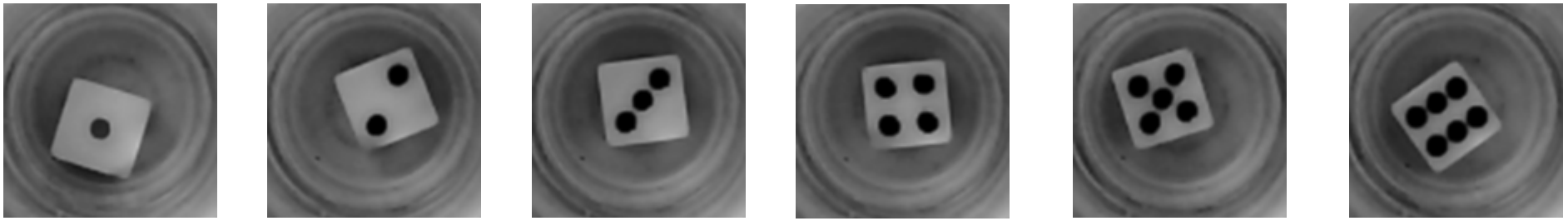
CNN of Dice

Presenter：杜彥陵

Scenario/Topic

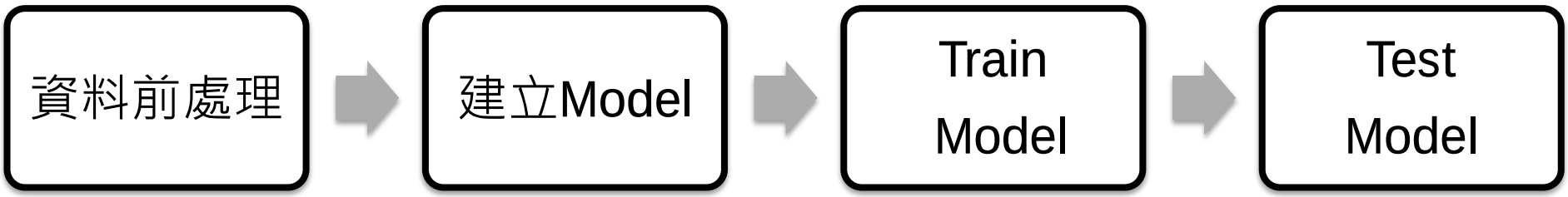
- 透過建立CNN模型，利用圖像辨識之技術辨識出dataset之骰子圖像數值，並計算其準確率。

- Dataset(每一面有400張，共2400張)



Link: DiceDataset

Process



Data-preprocessing process

- 將dataset以資料夾1~6進行分類

名稱	修改日期	類型
1	2019/10/27 下午 ...	檔案資料夾
2	2019/10/27 下午 ...	檔案資料夾
3	2019/10/27 下午 ...	檔案資料夾
4	2019/10/27 下午 ...	檔案資料夾
5	2019/10/27 下午 ...	檔案資料夾
6	2019/10/27 下午 ...	檔案資料夾

- 開啟資料圖像(路徑)，並將圖像以矩陣表示

```
In [2]: image = Image.open('C:/Users/913lab/Desktop/碩一/Project/DiceDataset/1/00000.bmp')
        w = np.array(image)
        print(w)
```

```
[[141 140 140 ... 107 108 109]
 [137 134 131 ... 107 107 109]
 [133 130 127 ... 107 108 109]
 ...
 [ 96  97  99 ... 169 171 173]
 [ 94  95  96 ... 172 173 173]
 [ 93  93  93 ... 175 174 171]]
```

Data-preprocessing process

- 每一面各400張->train300 (75%)、test100(25%)
- image reshape為(1,128,128,1)
- np.concatenate: 將images合併

In [3]:
```
train_images = np.zeros((1,128,128,1))
test_images = np.zeros((1,128,128,1))
for i in range(6):
    for j in range(400):
        if j<300:
            image = np.array(Image.open('C:/Users/913lab/Desktop/碩一/Project/DiceDataset/%d/%05d.bmp'%(i+1,j)))
            image = image.reshape((1,128,128,1))
            train_images = np.concatenate((train_images,image),axis=0)
        else:
            image = np.array(Image.open('C:/Users/913lab/Desktop/碩一/Project/DiceDataset/%d/%05d.bmp'%(i+1,j)))
            image = image.reshape((1,128,128,1))
            test_images = np.concatenate((test_images,image),axis=0)

        if j==399:
            print("We have processed dice %d"%(i+1))
```

```
We have processed dice 1
We have processed dice 2
We have processed dice 3
We have processed dice 4
We have processed dice 5
We have processed dice 6
```

Data-preprocessing process

- 將train images及test images標準化(/255)

In [4]:
```
train_images = np.delete(train_images,0,0)
test_images = np.delete(test_images,0,0)

train_images = train_images/255.0
test_images = test_images/255.0

print('Shape of training data: {}'.format(train_images.shape))
print('Shape of testing data: {}'.format(test_images.shape))
```

```
Shape of training data: (1800, 128, 128, 1)
Shape of testing data: (600, 128, 128, 1)
```

Data-preprocessing process

- 將圖像進行label、進行合併

In [5]:
```
train_labels = np.array([0])
test_labels = np.array([0])

for i in range(6):
    for j in range(400):
        if j<300:
            train_labels = np.concatenate((train_labels,np.array([i])),axis=0)
        else:
            test_labels = np.concatenate((test_labels,np.array([i])),axis=0)

train_labels = np.delete(train_labels,0,0)
test_labels = np.delete(test_labels,0,0)

print('Shape of training label: {}'.format(train_labels.shape))
print('Shape of testing label: {}'.format(test_labels.shape))
```

```
Shape of training label: (1800,)
Shape of testing label: (600,)
```

Data-preprocessing process

- tf.cast:型別轉換函式，把值轉換成浮點數
- tf.data.Dataset.from_tensor_slices((features, labels)) shuffle打亂圖像； batch數據數量

In [6]:
```
train_images = tf.cast(train_images,dtype='float32')
test_images = tf.cast(test_images,dtype='float32')
train_dataset = tf.data.Dataset.from_tensor_slices((train_images,train_labels)).shuffle(1800).batch(30)
test_dataset = tf.data.Dataset.from_tensor_slices((test_images,test_labels))
print(train_dataset)
print(test_dataset)
```

```
<BatchDataset shapes: ((None, 128, 128, 1), (None,)), types: (tf.float32, tf.int32)>
<TensorSliceDataset shapes: ((128, 128, 1), ()), types: (tf.float32, tf.int32)>
```

國立清華大學 National Tsing Hua University

INNOVATION ENGINEERING LAB

Model architecture

```
In [7]: model = models.Sequential()
model.add(layers.Conv2D(filters=3,kernel_size=(3,3), strides=(1, 1), padding='valid',
                        activation=tf.nn.relu, input_shape=(128, 128, 1)))
model.add(layers.Conv2D(filters=6,kernel_size=(3,3), strides=(1, 1), padding='valid',
                        activation=tf.nn.relu, input_shape=(128, 128, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(filters=9,kernel_size=(3,3), strides=(1, 1), padding='valid',
                        activation=tf.nn.relu, input_shape=(128, 128, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(layers.Dropout(rate=0.5))
model.add(layers.Dense(units=6, activation=tf.nn.softmax))
```

CNN parameter	Value
Number of convolution layers	3
Filter size	3X3
Number of pooling layers	2 max-pooling
Pooling filter size	2X2
Number of feature maps	3/6/9
Number of fully connected layers	1
Number of nodes in the fully connected layer	6
Activation function	ReLU
Regularization method	Dropout
Classification function of the output layer	Softmax
Loss function	sparse_categorical_crossentropy

Improve model’s performance

- 調整參數(Filter階層數)>>提高準確率

In [7]:
```
model = models.Sequential()
model.add(layers.Conv2D(filters=3,kernel_size=(3,3), strides=(1, 1), padding='valid',
                        activation=tf.nn.relu, input_shape=(128, 128, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(layers.Dropout(rate=0.5))
model.add(layers.Dense(units=6, activation=tf.nn.softmax))
```

Test accuracy: 0.61

In [7]:
```
model = models.Sequential()
model.add(layers.Conv2D(filters=6,kernel_size=(3,3), strides=(1, 1), padding='valid',
                        activation=tf.nn.relu, input_shape=(128, 128, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(layers.Dropout(rate=0.5))
model.add(layers.Dense(units=6, activation=tf.nn.softmax))
```

Test accuracy:0.88666666

Improve model’s performance

- 調整參數(activation)>>提高準確率

In [14]:
```
model = models.Sequential()
model.add(layers.Conv2D(filters=3,kernel_size=(3,3), strides=(1, 1), padding='valid',
                        activation=tf.nn.sigmoid, input_shape=(128, 128, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(layers.Dropout(rate=0.5))
model.add(layers.Dense(units=6, activation=tf.nn.softmax))
```

Test accuracy: 0.155

In [7]:
```
model = models.Sequential()
model.add(layers.Conv2D(filters=3,kernel_size=(3,3), strides=(1, 1), padding='valid',
                        activation=tf.nn.relu, input_shape=(128, 128, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(layers.Dropout(rate=0.5))
model.add(layers.Dense(units=6, activation=tf.nn.softmax))
```

Test accuracy: 0.61

Improve model’s performance

- 微調Model(增加層數)>>提高準確率

In [7]:
```
model = models.Sequential()
model.add(layers.Conv2D(filters=3,kernel_size=(3,3), strides=(1, 1), padding='valid',
                        activation=tf.nn.relu, input_shape=(128, 128, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(layers.Dropout(rate=0.5))
model.add(layers.Dense(units=6, activation=tf.nn.softmax))
```

Test accuracy: 0.61

In [7]:
```
model = models.Sequential()
model.add(layers.Conv2D(filters=3,kernel_size=(3,3), strides=(1, 1), padding='valid',
                        activation=tf.nn.relu, input_shape=(128, 128, 1)))
model.add(layers.Conv2D(filters=3,kernel_size=(3,3), strides=(1, 1), padding='valid',
                        activation=tf.nn.relu, input_shape=(128, 128, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(layers.Dropout(rate=0.5))
model.add(layers.Dense(units=6, activation=tf.nn.softmax))
```

Test accuracy: 0.90833336

Improve model’ s steps

改善步驟	Accuracy
原始Model	0.61
原始Model+activation(sigmoid->relu)	0.16->0.61
原始Model+filter(3->6)	0.61->0.89
原始Model+一層Conv2D	0.61->0.91

Training

In [11]:
```
import time
start_time = time.time()

rec = model.fit(train_images, train_labels, epochs=15)

print("--- %s seconds ---" % (time.time() - start_time))
```

```
Epoch 1/15
1800/1800 [==============================] - 13s 7ms/sample - loss: 1.7275 - accuracy: 0.2244
Epoch 2/15
1800/1800 [==============================] - 12s 7ms/sample - loss: 1.1410 - accuracy: 0.5767
Epoch 3/15
1800/1800 [==============================] - 12s 7ms/sample - loss: 0.7529 - accuracy: 0.7094
Epoch 4/15
1800/1800 [==============================] - 12s 7ms/sample - loss: 0.6111 - accuracy: 0.7789
Epoch 5/15
1800/1800 [==============================] - 12s 7ms/sample - loss: 0.5167 - accuracy: 0.8183
Epoch 6/15
1800/1800 [==============================] - 12s 7ms/sample - loss: 0.4522 - accuracy: 0.8361
Epoch 7/15
1800/1800 [==============================] - 12s 7ms/sample - loss: 0.4215 - accuracy: 0.8472
Epoch 8/15
1800/1800 [==============================] - 12s 7ms/sample - loss: 0.3696 - accuracy: 0.8711
Epoch 9/15
1800/1800 [==============================] - 12s 7ms/sample - loss: 0.3450 - accuracy: 0.8789
Epoch 10/15
1800/1800 [==============================] - 12s 7ms/sample - loss: 0.2832 - accuracy: 0.8972
Epoch 11/15
1800/1800 [==============================] - 12s 7ms/sample - loss: 0.2791 - accuracy: 0.9017
Epoch 12/15
1800/1800 [==============================] - 12s 7ms/sample - loss: 0.2531 - accuracy: 0.9139
Epoch 13/15
1800/1800 [==============================] - 12s 7ms/sample - loss: 0.2610 - accuracy: 0.9000
Epoch 14/15
1800/1800 [==============================] - 12s 7ms/sample - loss: 0.2577 - accuracy: 0.9072
Epoch 15/15
1800/1800 [==============================] - 12s 7ms/sample - loss: 0.2182 - accuracy: 0.9217
--- 186.15600037574768 seconds ---
```

Epochs=15
Training time=186.16secs

國立清華大學 National Tsing Hua University

INNOVATION ENGINEERING LAB

Results

- Test image->Test accuracy=0.93

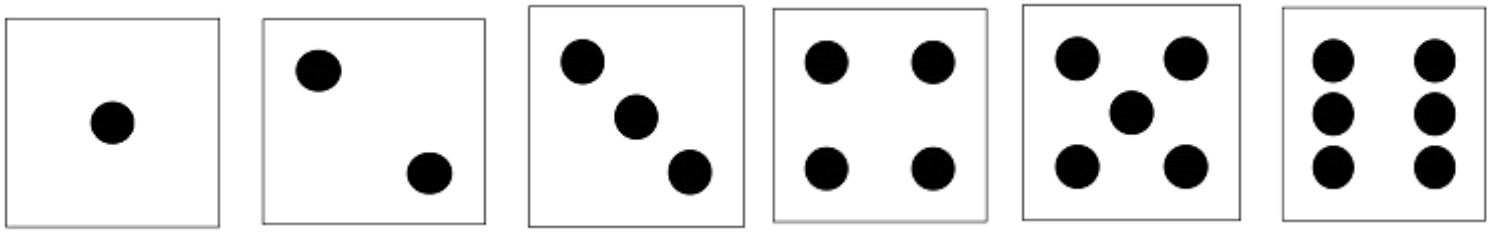
```
In [12]: test_loss, test_acc = model.evaluate(test_images, test_labels)
         print('\nTest accuracy:', test_acc)

         600/600 [==============================] - 1s 2ms/sample - loss: 0.2181 - accuracy: 0.9333

         Test accuracy: 0.93333334
```

Test for the new image

- 每一面各兩張，共12張



```
In [16]: test_new_loss, test_new_acc = model.evaluate(test_new_image, test_new_label)
         print('\nTest accuracy:', test_new_acc)

         12/12 [==============================] - 0s 2ms/sample - loss: 1263.4225 - accuracy: 0.3333

         Test accuracy: 0.33333334
```

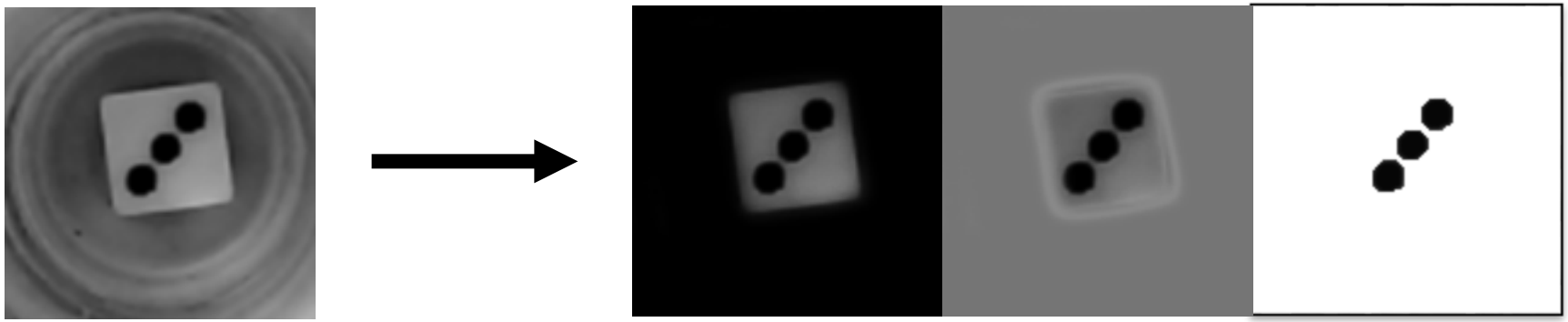
Test accuracy: 0.33

Discussion

- 雖然可以透過上述的方法來提高Model的準確率，但由於本dataset較為簡單與單一，所以可能造成new image的測試準確率不高。

Future work

- 增加dataset的複雜度
- 進行更多前處理-將image灰階化，讓顏色區分為黑與白，以利後續辨識。



Discussion

- 遇到的困難

  1.程式用語不懂

  2.準確率難提升

- 如何克服

  上網搜尋類似例子(修改參數)

Reference

- Dataset:
  https://github.com/tomitomi3/DiceRecognitionDatasetForML
- Model:

  助教上課內容(W6-CNN)

  Keras Documentation https://keras.io/layers/convolutional/

  Tensorflow https://www.tensorflow.org/
- New image:
  https://cdn.clipart.email/a4671e739e665dc3dbe557549d36b5d8_dice-clipart-1-clipartfest-3-cliparting-com-picturesque-learnfreeme_2453-1588.jpeg

國立清華大學
National Tsing Hua University

INNOVATION ENGINEERING LAB

Thank You for Your Listening