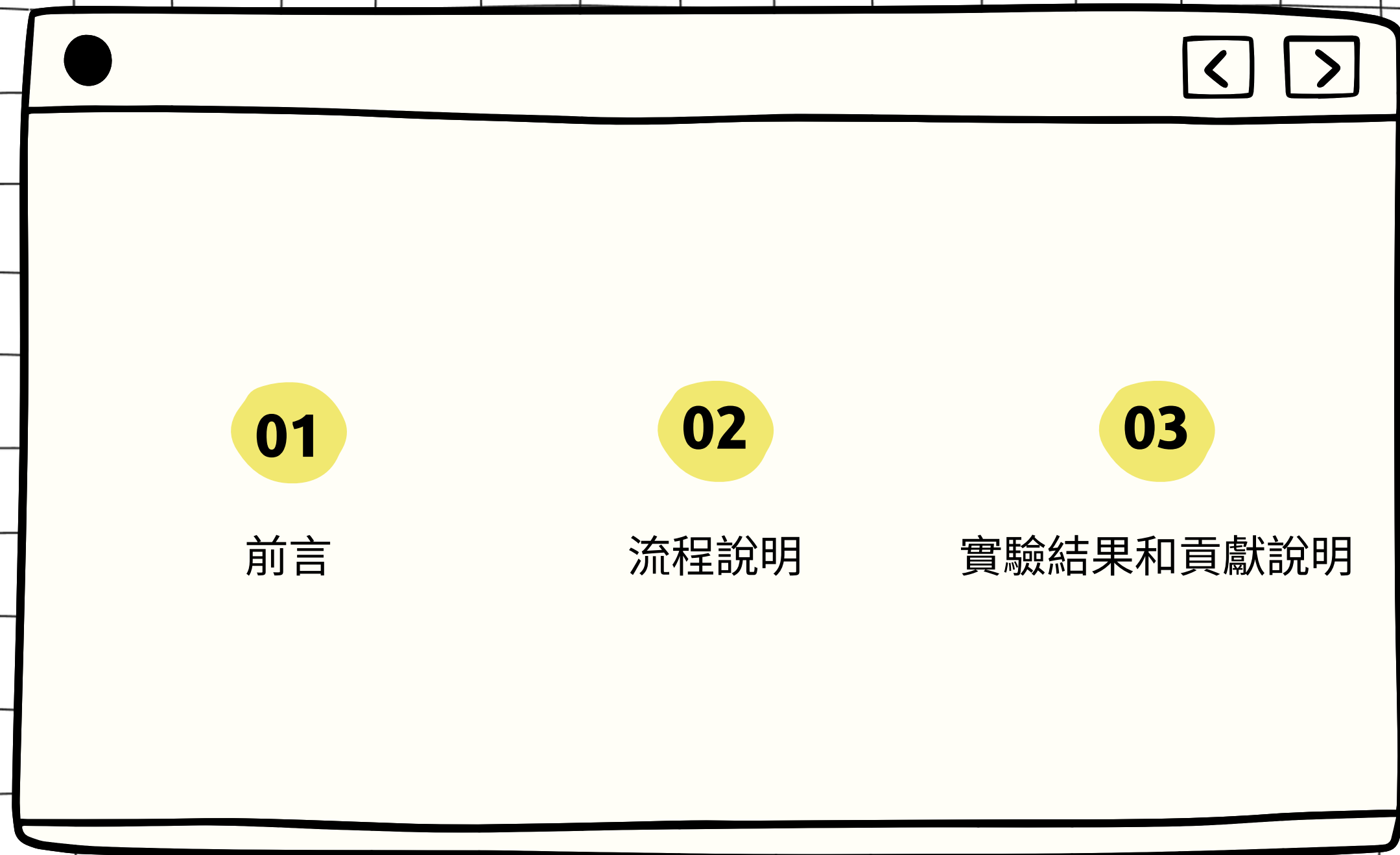


# 北科簡易驗證碼辨識

1102B0005吳威霆

1102B0016陳書涵





**01**

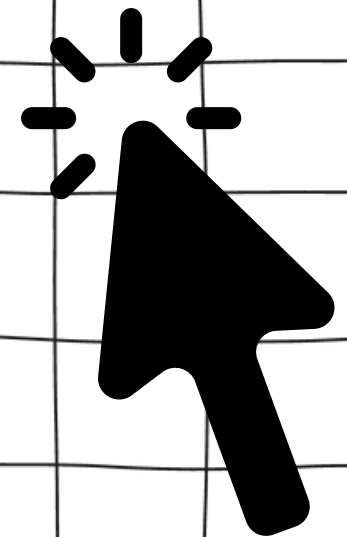
前言

**02**

流程說明

**03**

實驗結果和貢獻說明



# 前言

每一次登入北科的校園入口網站時，都會很容易被驗證碼卡住



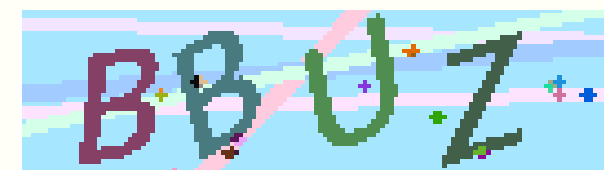
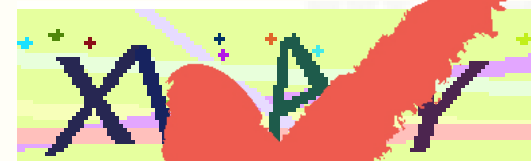
The screenshot shows the Taipei Tech Portal login interface. At the top, it says '校園入口網站' (Campus Entry Website) in orange and 'Taipei Tech Portal' in grey. Below this are four input fields: '使用者帳號 (Account)', '使用者密碼 (Password)', '驗證碼 (Code)', and '請輸入驗證碼 (Keyin Code)'. The '驗證碼 (Code)' field contains a colorful captcha image with the letters 'BBU-Z' and a refresh icon. At the bottom is a blue button labeled '登入 Login'.

Field Label	Field Content
使用者帳號 ( Account )	
使用者密碼 ( Password )	
驗證碼 ( Code )	BBU-Z
請輸入驗證碼 ( Keyin Code )	

登入 Login

# 前言

然而驗證碼的圖對於現在我來說實在是有點麻煩，每次登入每次都要打，尤其是在緊急的情況不小心打錯更是讓人血壓飆高。



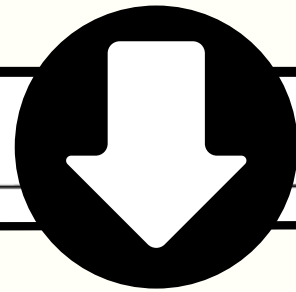
所以我們挑選了字母沒有重疊的圖

決定做一個幫助快速辨識驗證碼的程式

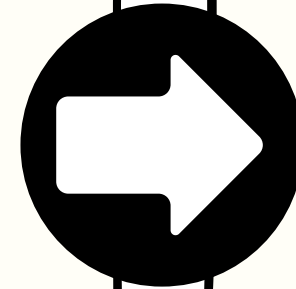


## 驗證碼圖片擷取(1644張)

我們要辨識的是北科校務系統的驗證碼，但是怕爬蟲等方式要花費時間及安全問題



寫一段程式讓它不停地  
截取網頁中的驗證碼並  
存檔成圖片



將每個圖片的黨  
名改成圖片上的  
字母





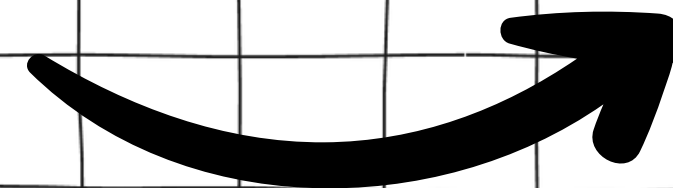
# 程式結構

資料  
預處理

資料  
切割

資料  
轉換

分訓練集  
和測試集



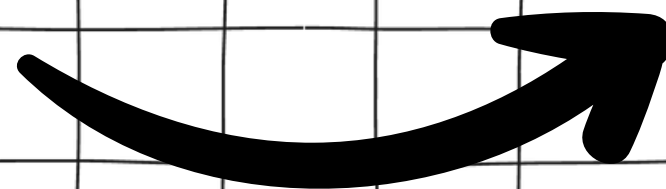
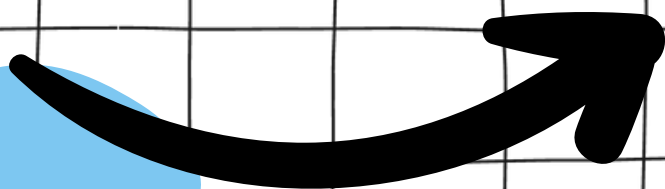
# 程式結構

建立  
模型

編譯  
模型

訓練  
模型

保存或  
載入模型





## 資料轉換 與分集

- 使用 `keras.utils.to_categorical` 將標籤轉換為 one-hot 編碼。
- 使用 `train_test_split` 函數將資料切割為訓練集和測試集，其中測試集佔總資料的 20%。

```
# 將標籤轉換為 one-hot 編碼
y_list = keras.utils.to_categorical(y_list, num_classes=26)
# 劃分訓練集和測試集
x_train, x_test, y_train, y_test = train_test_split(x_list, y_list, test_size=0.2, random_state=42)
```

# 建立模型

```
model = models.Sequential()
model.add(layers.Conv2D(32, kernel_size=(3, 3), activation='relu',
                        input_shape=(img_rows, img_cols // letters_in_img, 1)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D(pool_size=(2, 2)))
model.add(layers.Dropout(rate=0.25))
model.add(layers.Flatten())
model.add(layers.Dense(128, activation='relu'))
model.add(layers.Dropout(rate=0.5))
model.add(layers.Dense(26, activation='softmax'))
```

- 使用 Keras 的 Sequential 模型建立神經網絡。
- 添加兩個卷積層 (Conv2D) 和池化層 (MaxPooling2D)，以及兩個全連接層 (Dense)。
- 使用 relu 激活函數和 softmax 函數。

輸入層

卷積層 1

32 個 3x3 的卷積核  
啟動函數: ReLU

卷積層 2

64 個 3x3 的卷積核  
啟動函數: ReLU

最大池化層

2x2 的最大池化

Dropout 層

丟棄比例: 25%

平坦層

輸出平坦化

全連接層 1

128 個神經元  
啟動函數: ReLU

Dropout 層

丟棄比例: 50%

全連接層 2

26 個神經元  
啟動函數: softmax

## 編譯及訓練 模型

- 使用交叉熵  
(categorical\_crossentropy) 作為損失函數，Adam 優化器進行模型編譯。
- 使用 model.fit 方法進行模型訓練。

```
# 编译模型
model.compile(loss=keras.losses.categorical_crossentropy, optimizer=keras.optimizers.Adam(), metrics=['accuracy'])
# 训练模型
model.fit(np.array(x_train), np.array(y_train), batch_size=letters_in_img, epochs=epochs, verbose=1,
          validation_data=(np.array(x_test), np.array(y_test)))
```

## 保存或載入模型

如果已經存在  
'cnn\_model.h5' 檔案，則載入模型；否則保存模型。

```
# 保存或加載模型
if os.path.isfile('cnn_model.h5'):
    model = models.load_model('cnn_model.h5')
    print('Model loaded from file.')
else:
    model.save('cnn_model.h5')
    print('Model saved.')
```

 cnn.py

 cnn\_model.h5

## 設定圖片座標和寬跟高



起始X座標:

100

起始Y座標:

100

寬度:

200

高度:

200

按空白鍵截圖

```
tk.Label(self.master, text="起始X座標:").pack()
tk.Entry(self.master, textvariable=self.start_x).pack()

tk.Label(self.master, text="起始Y座標:").pack()
tk.Entry(self.master, textvariable=self.start_y).pack()

tk.Label(self.master, text="寬度:").pack()
tk.Entry(self.master, textvariable=self.width).pack()

tk.Label(self.master, text="高度:").pack()
tk.Entry(self.master, textvariable=self.height).pack()
```

## 導進 模型

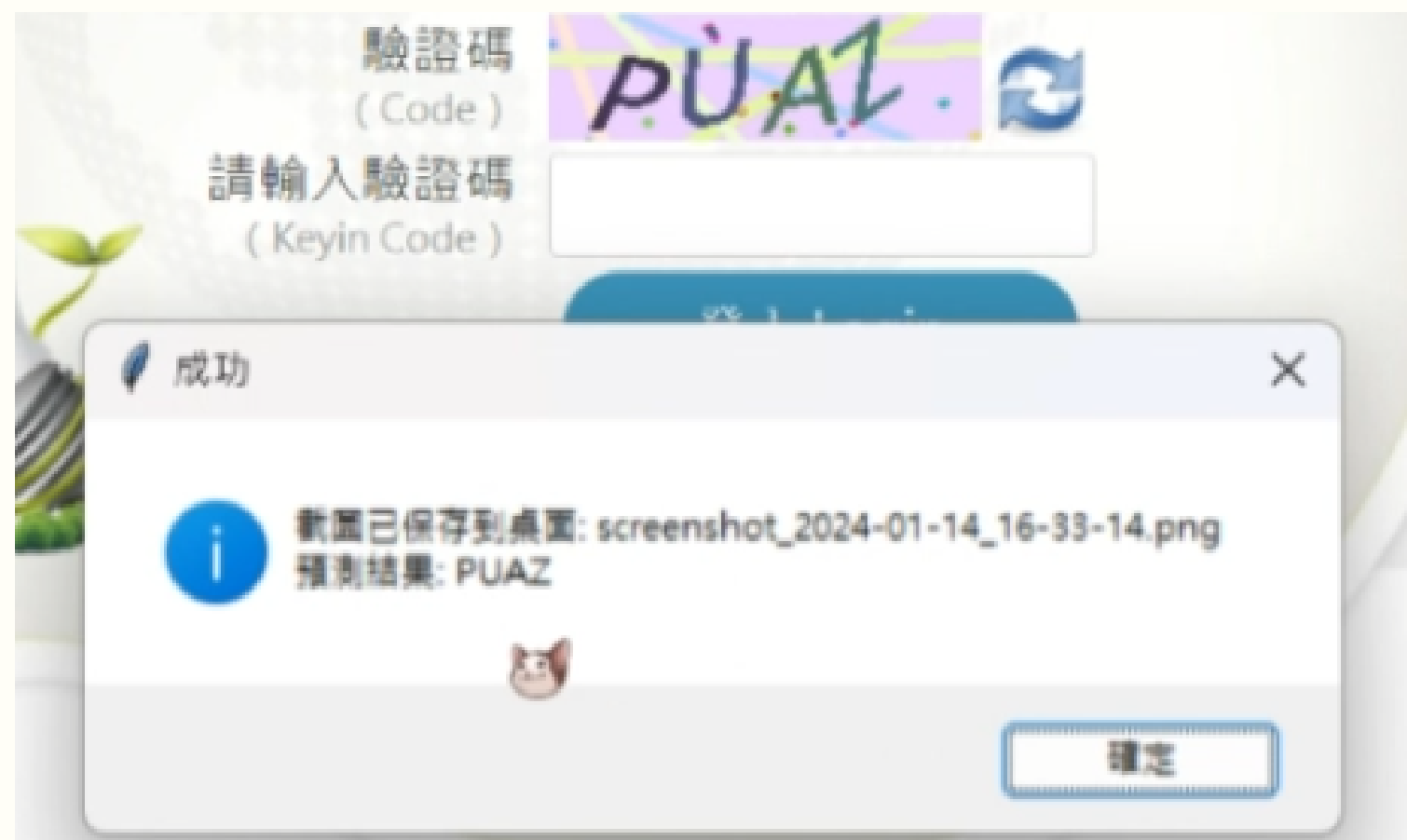
- 將截圖後的結果導入我們所訓練的模型裡面

```
self.model = load_model('cnn_model.h5')
```

```
predictions = self.model.predict(preprocessed_image)
```



## 顯示判斷結果



```
messagebox.showinfo("成功", f"截圖已保存到桌面: {file_name}\n預測結果: {predicted_result}")

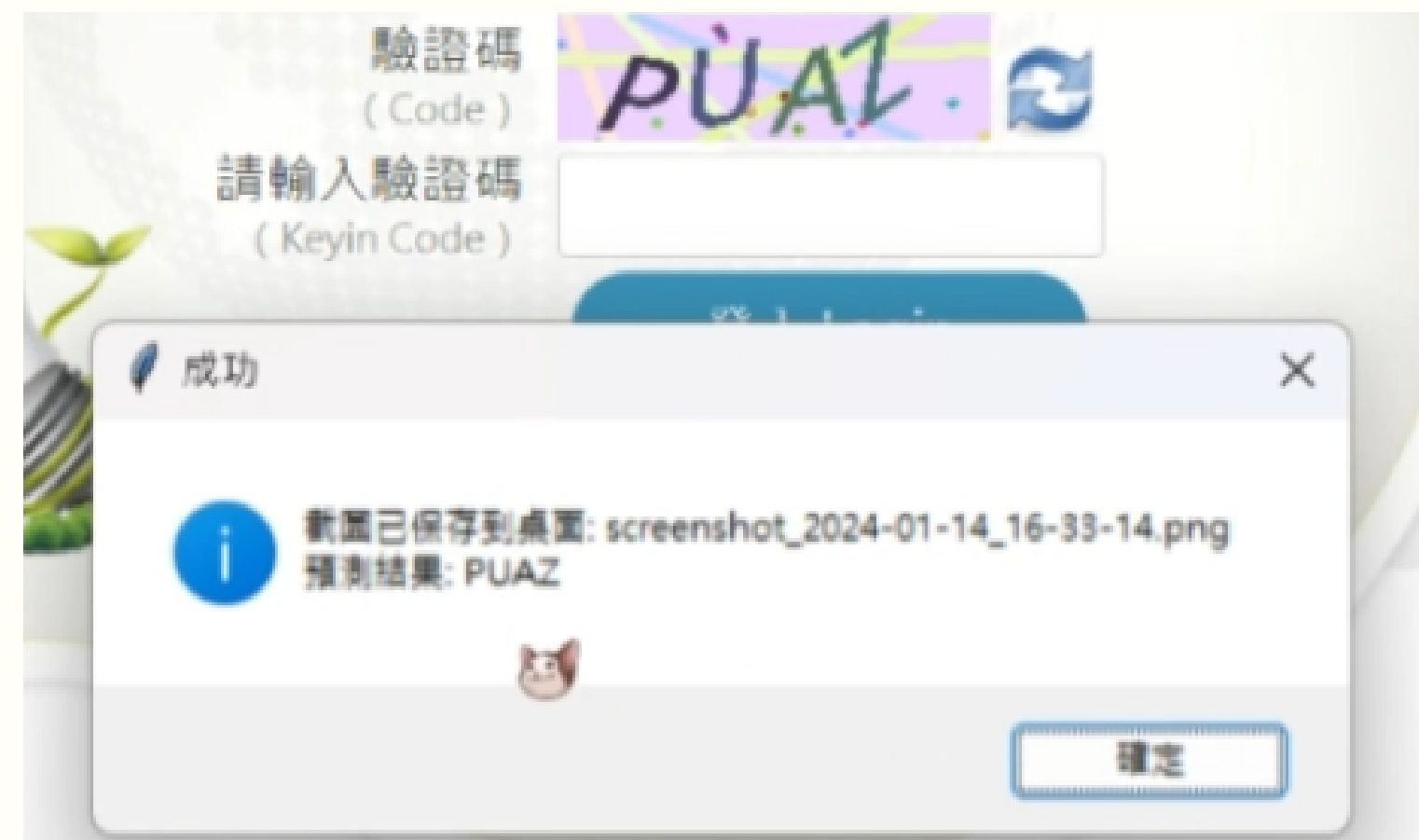
def preprocess_image(self, image_path):
    img = load_img(image_path, color_mode='grayscale', target_size=(38, 135))
    img_array = img_to_array(img)

    # 將圖像切分為單個字母
    x_list = []
    for i in range(4): # 4是驗證碼中的字母數
        step = 135 // 4
        x_list.append(img_array[:, i * step:(i + 1) * step] / 255)

    return np.array(x_list)
```




## 實驗結果





## 貢獻說明

- 1.我們收集了至少1600百張的圖片並且一一改名
  - 2.因為原本網路上的架構是數字，我們改成英文字母辨識
  - 3.將原本螢幕截圖的程式融合了導入模型的程式讓他截圖之後可以直接顯示出結果
- 

## 結語

這次做的驗證碼辨識從一開始的資料準備，到後期的程式修改，我們都花了許多時間在處理，完成訓練後，我們結合之前資料處理的程式跟辨識的程式，成功讓在網頁上完成，之後也可以結合網頁做運用，做成網路應用程式等等。