**台科大 111 學年度「彩色影像處理」作業三：簡化的CNN數值影像辨識**

M11125016 蕭強

Programming language: Python

import random

import cv2

import numpy as np

import matplotlib.pyplot as plt

from sklearn.metrics import confusion\_matrix

data\_path = "C:/Users/cghsi/Desktop/HW3/train1000/"

# 第一層捲積

fil1 = np.array([-1, -1, 1, -1, 0, 1, -1, 1, 1]).reshape(3, 3)

# 第二層捲積

fil2 = fil1.T

def readImg(num1, num2):

    """

    讀取照片

    """

    train = []

    num1 = num1 + 1 if num1 == 0 else num1 \* 100

    num2 = num2 + 1 if num2 == 0 else num2 \* 100

    for i in range(num1, num1 + 100):

        img = cv2.imread(data\_path + str(i) + ".png", 0)

        img = img / 255

        img = cv2.resize(img, (8, 8), interpolation=cv2.INTER\_AREA)

        train.append(img)

    for i in range(num2, num2 + 100):

        img = cv2.imread(data\_path + str(i) + ".png", 0)

        img = img / 255

        img = cv2.resize(img, (8, 8), interpolation=cv2.INTER\_AREA)

        train.append(img)

    return train

def con2d\_1l(input, kernel1=fil1, kernel2=fil2):

    """

    第一層卷積

    """

    output = np.array([cv2.filter2D(input, -1, kernel1), cv2.filter2D(input, -1, kernel2)])

    output = output.transpose(1, 2, 0)

    return output

def con2d\_2l(input, kernel1=fil1, kernel2=fil2):

    """

    第二層卷積

    """

    a = np.array([cv2.filter2D(input[:, :, 0], -1, kernel1)])

    b = np.array([cv2.filter2D(input[:, :, 1], -1, kernel2)])

    output = np.concatenate((a, b))

    output = output.transpose(1, 2, 0)

    return output

def maxPooling(feature\_map, size=2, stripe=2):

    """

    MaxPooling

    """

    in\_ax0, in\_ax1, in\_ax2 = np.shape(feature\_map)

    # output size

    output\_ax0 = int((in\_ax0 - size) // stripe + 1)

    output\_ax1 = int((in\_ax1 - size) // stripe + 1)

    output\_ax2 = in\_ax2

    output = np.zeros((output\_ax0, output\_ax1, output\_ax2))

    for i in range(output\_ax0):

        for j in range(output\_ax1):

            for f in range(output\_ax2):

                start\_y = i \* stripe

                start\_x = j \* stripe

                crop\_map = feature\_map[start\_y:start\_y + size, start\_x:start\_x + size, f]

                max\_value = np.max(crop\_map)

                output[i, j, f] = max\_value

    return output

# def confusion\_matrix(Y\_true, Y\_pred):

#     mat = np.zeros((2, 2))

#     K = len(np.unique(Y\_pred))

#     print(K)

def trainModel(dataset, num1, num2):

    # label X is the feature map after convolution

    X = []

    train\_nums = len(dataset)

    Y = np.zeros(train\_nums)

    Y[int(train\_nums / 2):] += 1

    Y = np.array(Y).reshape(train\_nums, 1)

    for img in dataset:

        # first layer

        img = con2d\_1l(img)

        img = maxPooling(img)

        # second layer

        img = con2d\_2l(img)

        img = maxPooling(img)

        # flatten layer

        img = np.ndarray.flatten(img)

        img = np.append(img, 1)

        X.append(img)

    X = np.array(X)

    # Linear regression

    A = np.dot(np.linalg.inv(np.dot(X.T, X)), (np.dot(X.T, Y)))

    # Predict value

    Yp = np.dot(X, A)

    # Confusion matrix

    Y\_pred = np.where(Yp > 0.5, num2, num1)

    Y\_true = Y

    confusion\_mat = confusion\_matrix(Y\_true, Y\_pred)

    acc = confusion\_mat[0][0] + confusion\_mat[1][1] / train\_nums

    print(f"Confusion matrix:\n{confusion\_mat}")

    print(f"Accuracy:{acc}")

    return Y\_pred, Y\_true

def plot\_result(label, predic, num1, num2):

    plt.figure()

    for i in range(15):

        random\_img = random.randint(0, 199)

        if random\_img < 100:

            img\_num = num1 \* 100 + random\_img

        else:

            img\_num = num2 \* 100 + (random\_img - 100)

        img = cv2.imread(data\_path + str(img\_num) + ".png", 0)

        ax = plt.subplot(3, 5, 1 + i)

        ax.imshow(img, cmap="gray")

        title = str(predic[random\_img]) + "(" + str(random\_img) + ")"

        if label[random\_img] == predic[random\_img]:

            color = "b"

        else:

            color = "r"

        ax.set\_title(title, fontsize=15, color=color)

        ax.set\_xticks([])

        ax.set\_yticks([])

    plt.show()

if \_\_name\_\_ == "\_\_main\_\_":

    num1 = 0

    num2 = 1

    training\_dataset = readImg(num1, num2)

    x = trainModel(training\_dataset, num1, num2)

    Y\_pred, Y\_true = trainModel(training\_dataset, num1, num2)

    plot\_result(Y\_true, Y\_pred, num1, num2)

Result:

(base) C:\Users\cghsi\Desktop\HW3>python hw1.py

Confusion matrix:

[[82 18]

[13 87]]

Accuracy:82.435

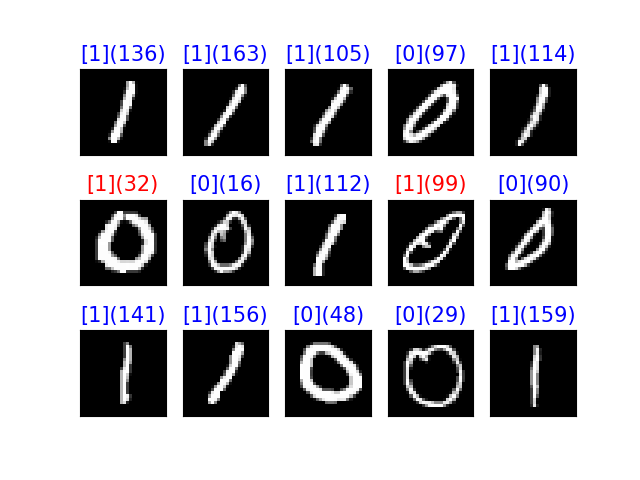


Figure 1 隨機選15張訓練影像顯示辨識成果，以’0’,’1’為例。