美瞳鏡片偵測

王均琦
107502545
國立中央大學資訊工程學系
二 B

曾專佩
107502015
國立中央大學資訊工程學系
二 B

***Abstract*—眼睛中的美瞳檢測是提高虹膜識別系統可靠性的一項重要任務。美瞳覆蓋虹膜區域並防止虹膜傳感器捕獲正常虹膜區域。在本文中，我們提出了一種使用深度卷積神經網絡（CNN）的美瞳檢測方法。**

Keywords—CNN, 虹膜, 深度學習, 美瞳, 辨識

I. INTRODUCTION

虹膜被認為是重要的生物特徵，具有準確的驗證/識別性能，因此已用於各種訪問控制應用程序，包括大規模應用程序，如阿聯酋入境口和印度的 UIDAI 項目。虹膜生物計量學的準確和可靠表現歸因於豐富的紋理特徵，這些特徵在受試者之間是非常獨特的。

在影響虹膜生物識別技術性能的各種因素中，接觸鏡的使用已顯示出會降低虹膜識別系統的性能。通常，隱形眼鏡可以是兩種類型，即：（1）透明（或軟）美瞳（2）變形（或美容）美瞳。通常使用軟性美瞳替代眼鏡以矯正視力。由於軟性美瞳會改變虹膜區域的反射特性，因此有望影響虹膜識別系統的性能。此外，位於虹膜邊界上的軟性美瞳的輪廓也將導致虹膜分割失敗。因此，已顯示使用軟性接觸鏡會稍微降低其整體識別的準確性。質感（或裝飾性）接觸鏡片具有外部質感和印刷在鏡片上的顏色圖案。當戴上帶紋理的美瞳時，它們會阻擋自然的虹膜圖案，從而導致無法通過識別系統獲取真實的虹膜紋理信息。因此，軟性和帶紋理的隱形眼鏡的檢測是提高虹膜識別可靠性的重要問題。

II. METHODOLOGY

我們這組使用 Matlab 來建構這個模組，實驗研究方法分成以下幾個部分：

A. Prepare Testing & Training Data

B. Pre-processing

C. CNN Network

D. Evaluate Classifier

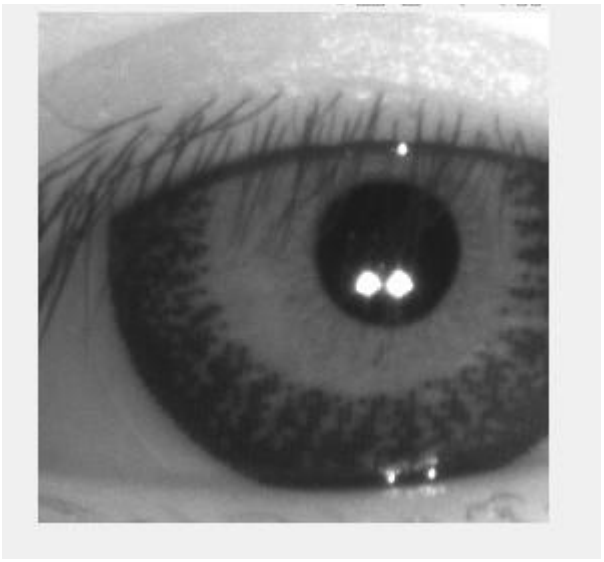
A. Preparing Testing & Training Data

用 imageDatastore 抓全部的照片，並用 splitEachLabel 把兩種 Label 的照片各自分成兩部分，並用 70% Training Data 和 30%的 Testing Data 做比例，最後用’randomize’ 參數讓兩種標籤的照片隨機挑選。

原本我們是直接輸入要用幾張，我們當時設各取 1000 張，總共有 2800 多張，但在 Demo 的時候，老師建議我們直接用比例的方式更好，所以我們就把第二個參數改成 0.3，而不是 1000。

B. Pre-processing

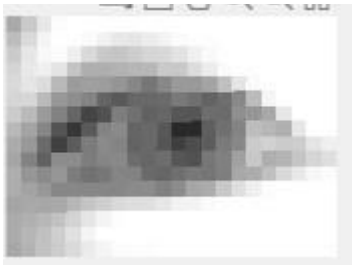
我們用 augmentedImageDatastore 這個 function 進行資料前處理。其中，參數 'ColorPreprocessing' 設為 'rgb2gray'。因為我們 load 進來的照片是 bmp 檔，會分成 RBG 三個 channel，但其實老師提供的圖片本身就是 gray scale 了，所以我們用此方法去掉多餘的 channel。'OutputSizeMode'參數，我們設為'centercrop'，因為瞳孔都在圖片中間的位子，所以我們設定裁切保留中間的部分，而大小，我們最後決定為 255x255。裁切後的照片如圖一。



(圖一)255x255 CenterCrop WithCCL

我們曾經因為怕電腦跑不動，把整張 480x640 的照片直接 resize 成 16x16，但就讓照片失去關鍵特徵了，如圖二和圖三，即便是用肉眼也很難分辨得出來。

(圖二) 16x16 Resize WithCCL



(圖三)16x16 Resize Without_CCL

C. *CNN Network*

我們的 imageInputLayer 是 255x255x1。總共用了四層 Layer，每一層 Layer 包含：

1) convolution2dLayer，其中有 32x(Layer)個 3x3 的 Padding

2) batchNormalizationLayer

3) reluLayer，讓負值全部變成零

4) maxPooling2dLayer，其中 pool size 為 2x2、stride 為 3x3

四層後，加上 dropoutLayer，這樣可以避免 overfitting 和加快處理速度。

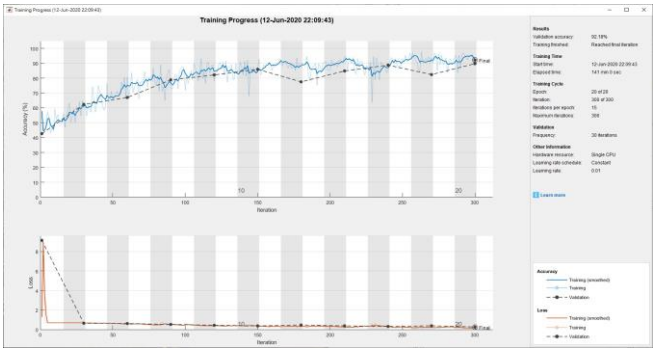
最後用 fullyConnectedLayer(2)，把全部資料分成 2 類。softmaxLayer、classificationLayer，作為結尾。架構如圖四。

ANALYSIS RESULT

	Name	Type	Activations	Learnables
1	imageinput 255x255x1 images with 'zerocenter' normalization	Image Input	255×255×1	-
2	conv_1 32 3x3x1 convolutions with stride [1 1] and padding 'same'	Convolution	255×255×32	Weights 3×3×1×32 Bias 1×1×32
3	batchnorm_1 Batch normalization with 32 channels	Batch Normalization	255×255×32	Offset 1×1×32 Scale 1×1×32
4	relu_1 ReLU	ReLU	255×255×32	-
5	maxpool_1 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	127×127×32	-
6	conv_2 64 3x3x32 convolutions with stride [1 1] and padding 'same'	Convolution	127×127×64	Weights 3×3×32×64 Bias 1×1×64
7	batchnorm_2 Batch normalization with 64 channels	Batch Normalization	127×127×64	Offset 1×1×64 Scale 1×1×64
8	relu_2 ReLU	ReLU	127×127×64	-
9	maxpool_2 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	63×63×64	-
10	conv_3 128 3x3x64 convolutions with stride [1 1] and padding 'same'	Convolution	63×63×128	Weights 3×3×64×128 Bias 1×1×128
11	batchnorm_3 Batch normalization with 128 channels	Batch Normalization	63×63×128	Offset 1×1×128 Scale 1×1×128
12	relu_3 ReLU	ReLU	63×63×128	-
13	maxpool_3 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	31×31×128	-
14	conv_4 256 3x3x128 convolutions with stride [1 1] and padding 'same'	Convolution	31×31×256	Weights 3×3×128×256 Bias 1×1×256
15	batchnorm_4 Batch normalization with 256 channels	Batch Normalization	31×31×256	Offset 1×1×256 Scale 1×1×256
16	relu_4 ReLU	ReLU	31×31×256	-
17	maxpool_4 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	15×15×256	-
18	dropout 50% dropout	Dropout	15×15×256	-
19	fc_1 64 fully connected layer	Fully Connected	1×1×64	Weights 64×57600 Bias 64×1
20	relu_5 ReLU	ReLU	1×1×64	-
21	fc_2 2 fully connected layer	Fully Connected	1×1×2	Weights 2×64 Bias 2×1
22	softmax softmax	Softmax	1×1×2	-
23	classoutput crossentropyex	Classification Output	-	-

(圖四)- analyzeNetwork 顯示架構

trainingOptions 我們使用'sgdm', 學習速率'InitialLearnRate'為 0.01, 學習次數'MaxEpochs'為 20, 測資'ValidationData' 取自於 augValidation 、'ValidationFrequency'為 30, 最後以視覺化顯示訓練狀態'Plots'為'training-progress'，如圖五。



(圖五)Trainning Progress-Accuracy 92.18%, Elapsed time 141mins

D. *Evaluate Classifier*

Demo 時老師說最好可以輸入一張照片，然後讓他輸出判斷是否有帶美瞳。所以我們用 activations、fitcecoc、readimage，最後用 predict，讓他輸出一個 Label。

III. EXPERIMENTS AND RESULTS

實驗數據為 480x640 .bpm 照片，分成兩種分類，With_CCL 1349 張，Without_CCL 1469 張，共 2818 張照片。

最後我們的準確率達到 92.18%，Training 費時 141 mins，如圖六。

Results

Validation accuracy:	92.18%
Training finished:	Reached final iteration

Training Time

Start time:	12-Jun-2020 22:09:43
Elapsed time:	141 min 0 sec

Training Cycle

Epoch:	20 of 20
Iteration:	300 of 300
Iterations per epoch:	15
Maximum iterations:	300

Validation

Frequency:	30 iterations

Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Constant
Learning rate:	0.01

(圖六)Training Result 6/12 Validation Accuracy 92.18%

IV. CONCLUSION

Convolutional Neural Nework 這種深度學習最厲害的就是在處理照片類型的資料，所以能夠邊處理邊把

照片顯示出來，是最好調整架構、參數的方式。例如，在資料前處理的時候，就可以隨便顯示一張經過處理的照片，以資料視覺化的形式，直接看看是不是跟你預想的一樣、是否達到自己設想的效果，就像平常寫程式在 debug 一樣。

ACKNOWLEDGMENT *(Heading 5)*

謝謝老師在我們 Demo 的時候幫我們注意到我們的問題，讓我們從那時的 83%進步到現在 92.18%！謝謝老師讓我們更進步，我們也學到很多處理問題的方式，我們受益良多!

REFERENCES

[1] Contact lens detection in iris images --Jukka Komulainen, Abdenour Hadid and Matti Pietikainen
http://www.ee.oulu.fi/~jukmaatt/papers/IETbook2017ch12.pdfJ.

[2] https://www.mathworks.com/solutions/deep-learning/convolutional-neural-network.html

[3] https://www.itread01.com/content/1546959428.html

[4] https://www.mathworks.com/help/vision/examples/image-category-classification-using-deep-learning.html

[5] https://www.mathworks.com/help/deeplearning/ug/deep-learning-in-matlab.html