以字元細化分割與 BPNN 辨識檢測 CAPTCHA 防護機制有效性之研究

侯佳利*
國立東華大學資訊管理學系
alexhou@mail.ndhu.edu.tw

摘要

由於電子商務之興起，交易網站若遭受入侵將產生巨額損失，因此需要有效的身份識別與驗證機制，以確保交易的安全性及不可否認性。傳統廣泛採用帳號、密碼的驗證機制做為身份識別之用。然而，密碼的驗證機制很容易遭受程式以字典法或暴力法進行攻擊。為了避免資訊系統遭受惡意程式的攻擊，因此發展出以 CAPTCHA (Completely Automated Public Turing Tests to Tell Computers and Humans Apart)的人機辨識輔助驗證機制，以防止惡意程式直接輕易對密碼進行猜測攻擊。

CAPTCHA 是一種由電腦分辨互動對象是電腦還是人的驗證機制，用以阻斷全自動機器人程式的惡意攻擊行為。本研究將針對大型的電子商務平台進行 CAPTCHA 的人工智慧辨識，進行其有效性之檢測。

本研究透過字元筆劃軌跡細化處理，強化字元的分割作業，將可分割的字元擷取出來，再以倒傳遞類神經網路(Back Propagation Neural Network, BPNN)作為字元辨識的工具。本研究除透過完整字元圖像進行訓練外，並增加擷取圖像 90%與 80%作為輔助網路之訓練樣本，希望透過強化學習字元部分特徵，整合三個類神經網路，以增加辨識的準確率。

根據本研究實驗結果，應用單一網路的字元辨識率為 94%，若整合三個 BPNN 網路的字元辨識率則可提升至 96%，驗證了透過截取不同比例的特徵圖形，進行多網路辨識決策，將有助於提升單一 BPNN 網路的辨識率。也顯示現有應用於電子商務的 CAPTCHA 機制需持續檢測及強化，以發揮充足之保護效果。

關鍵詞：CAPTCHA、字元細化、字元辨識、字元分割、倒傳遞類神經網路

Applying stroke thinning and BPNN recognition for CAPTCHA inspection

Jia-Li Hou*
Department of Information Management, National Dong Hwa University
alexhou@mail.ndhu.edu.tw

* 通訊作者 (Corresponding author.)

Abstract

The Internet developed rapidly along with the rise of e-commerce. In the past, Password-based authentication schemes are the most widely used mechanisms over e-commerce. Password-based authentication schemes may be weak to the dictionary attack. CAPTCHA authentication schemes are proposed for prevent automated script attacking.

Past researchers have never proposed methods to solve overlapping and connecting problems. Therefore, we want to propose an effective method to solve overlapping and connecting problems. We detect the CAPTCHA and extract the divisible parts from the CPATCHA firstly. Then we use thinning method to increase the gap of characters. After extracting characters, we recognize the characters by using back propagation neural network (BPNN). In order to improve the problem of false positives, we will train two additional networks. I trained the network for the normalized image firstly. Then we capture the normalized image 90% and 80% for the other networks' training samples. We hope that the other two networks will learn same features of the part of characters. In this way, there are some bases for adjustment when single network answer wrong. Finally, we integrated the tree networks by voting.

According to the experiment, the recognition rate of single network is 94% and the recognition rate of integrated three networks is 96%. It prove that multi-networks are better than single network.

Keywords: CAPTCHA、character thinning、character recognition、character segmentation、BPNN

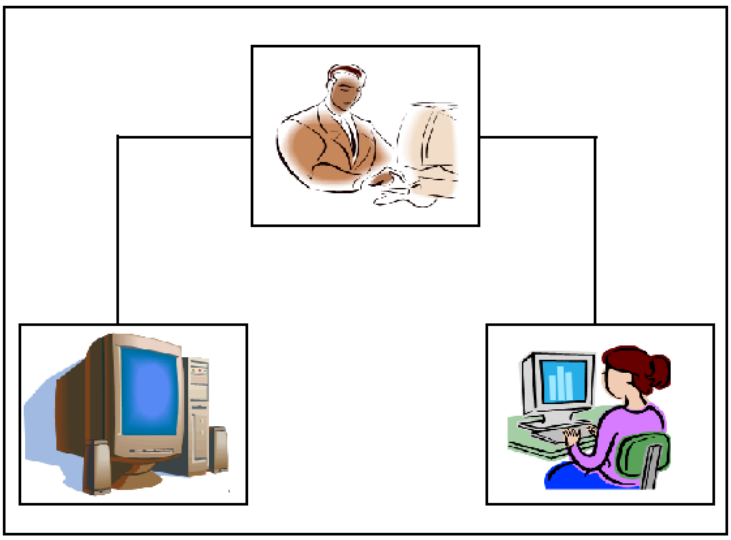
壹、前言

隨著網際網路的快速發展，原本實體店面提供的服務，都紛紛轉而提供網路服務，為了保障使用者和服務提供者的權益，透過使用者鑑別機制保障交易變得更加重要。由於以帳號、密碼為基礎的鑑別方式具有使用簡單且易於管理的特點，因此廣泛運用在電子商務資訊系統上。然而，隨著電子商務的快速發展，網路也出現不少攻擊密碼機制的行為，字典攻擊為其中入侵者主要攻擊方法之一。所謂的字典攻擊即入侵者透過「全自動機器人程式」(Automated Scripts)從字典中，將每一個單字和使用者的密碼進行比對，因一般常見單字約有二萬多個字左右，而程式比對一個單字僅需 1 毫秒，因此字典攻擊法可能在 20 秒內即可猜中使用者的密碼，足見密碼機制對抗程式攻擊其安全性是不足

的。

近幾年來使用全自動機器人程式的攻擊手法在網路上日益猖獗，有心人士更利用全自動機器人程式進行大量註冊帳號、在留言版大量不當留言或重複登入…等等的DoS(Deny of Servie)攻擊，導致網路癱瘓或不當濫用。不僅如此，全自動機器人程式的惡意使用，也造成使用者的公平性受到關注，例如在 2005 年時，有人透過自動訂票程式的工具惡意搶票，造成許多民眾無法順利購得火車票之情況發生[16]。

就網路上資訊傳遞或電子交易而言，很多時候我們必須確認在網路的另一端的使用者是個「人」，而不是「電腦」，這就是所謂人機區隔(Reverse Turning Test)的概念。Reverse Turning Test 主要是透過電腦辨別受測者是電腦或是人類，其概念主要源自早期的 Turning Test (圖靈測試) [13]，如圖一所示，原本在 Turning Test 中分辨在電腦的另一端是人或電腦的工作，是由人類負責，Reverse Turning Test 則是希望透過電腦能自動完成判斷的動作。



圖一：Turning Test 圖靈測試示意圖

因此，Luis von Ahn 等學者提出一個名為 Completely Automated Public Turing Tests to Tell Computers and Humans Apart (CAPTCHA)的方式 [1]，主要概念為設計一個自動化的程式，用來區隔電腦和人類，其定義如下所述：

- 必須讓人類可以通過
- 必須讓電腦程式無法通過

圖二：目前 eBay 使用的 CAPTCHA

圖三：台灣鐵路局使用的 CAPTCHA

因此各網站紛紛採用 CAPTCHA 的機制，以防止程式的攻擊，在瀏覽網頁或登入服務時，常會看到網站要求使用者依照附有扭曲或背景有雜訊的字串圖片，需鍵入圖片中的字元，判斷目前使用者是否為「人類」的依據。如圖二所示，eBay 商務網站的 CAPTCH 防護圖型。而如圖三所示，台灣鐵路局的網路訂票系統也在 2006 年加入 CAPTCHA 的方式以防止「搶票程式」的攻擊。

貳、文獻探討

本章節將描述 CAPTCHA 歷史以及設計的方法，並探討過去幾年裡關於攻擊 CAPTCHA 的方法，再藉以重新檢視目前全球及台灣使用中的 CAPTCHA 安全性可能面臨的威脅。

2.1 CAPTCHA 發展與面臨的挑戰

根據過去的文獻中指出最早的 CAPTCHA 可以追溯到 1997 年的 AltaVista [11]，當時為了防止 URL 註冊遭到濫用，DEC System Research Center 便設計出最早的 CAPTCHA 系統，其主要為一張簡單印有亂數文字的圖片，雖然對當時光學文字辨識(OCR)的攻擊沒有太大的抵抗力，但在當時它仍然為 AltaVista 阻擋了 95%的不正當使用，同時也成為第一個 CAPTCHA 系統。

圖四: Gimpy-r

圖五：2.2 Gimpy

到了 2000 年的時候，Yahoo!為了防止全自動機器人程式在聊天室裡散布垃圾留言，遂委託卡內基美濃大學為他們設計一套有效的防禦機制，而卡內基美濃的學者在研究過

程中，發現許多能有效提升 CAPTCHA 防禦強度的設計方式，也因此根據所發現的設計原則，提出許多不同的 CAPTCHA，如圖四和圖五所示。

雖然在 Gimpy-r 和 Gimpy 剛提出的當時，的確有效地遏止「全自動機器人程式」的攻擊，但好景不常，在 2003 年之後，分別受到嚴厲的考驗，(1)2003 年， Greg Mori 和 Jitendra Malik 提出如何辨識 Gimpy CAPTCHA 的方法，在大概了解每個位置可能的字元後，再透過字典比對，實驗結果約有 33%的辨識率[9]，2004 年，Gabriel Moy 等學者提出從 Gimpy-r CAPTCHA 分割出每個字元的方法，並針對每個字元辨識，其結果高達 78%的辨識率[7]。

表一:字元在不同扭曲下的辨識率

Characters under typical distortions	Recognition rate
	~100%
	96+%
	100%
	98%
	~100%
	95+%

CAPTCHA 陸續出現破解方法後，學者進一步證實 CAPTCHA 原本宣稱為防止光學文字辨識(Optical Character Recognition, OCR)技術而設計，但在 2003 年，有學者設計一個有效的單一字元辨識的機器學習方法，利用改良卷積神經網絡(Convolutional Neural Networks, CNN)，儘管字元任意旋轉、縮放、扭曲，甚至加入線條以干擾程式自動辨識，其方法都有很好的辨識率[3]。

如表一所示， 2005 年微軟研究所(Microsoft Research)的學者透過實驗更證實了一件事，以當時圖形辨識技術，在處理單一字元辨識時，機器學習法(Machine learning)其實擁有比人類更高的辨識能力[3]，因此若 CAPTCHA 只針對 OCR 的缺失而設計，便顯得安全性不足。最後，在過去的研究中，不難看出眾多學者認為 CAPTCHA 的設計應考量「分割」(Segmentation)的問題，增加從字串中分割每一個字的難度 [4][5]，才是關鍵。

雖然不斷有新的 CAPTCHA 被設計出來，但隨著人工智慧與圖形識別技術的發展，CAPTCHA 的安全強度也會逐漸降低，甚至被破解。因此，定期檢視網站使用之 CAPTCHA 的有效性非常重要，必須持續確保不被機器辨識和降低「全自動機器人程式」所造成的損害，以維護使用者的公平權益及減少不當的使用。

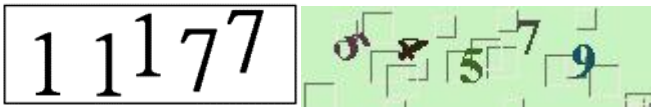
2.2 CAPTCHA 設計方式

文字式 CAPTCHA 主要是由幾個可辨識的字元組成，而過去為了防止既有的電腦識別技術 OCR 的攻擊，便改變圖型解析度或字的大小，以增加 OCR 辨識的難度，但太小的解析度或太小的字，都容易導致使用者辨識上的困難，顯然此類方法是不適合的，所以在而後的發展中，除了以上的方式外，還包含字體樣式(粗體、斜體等)和字體的改變，以及使用哪些字元集合(英文字母、數字、特殊符號等)，以上種種都在設計 CAPTCHA 扮演著重要的角色。目前常見的 CAPTCHA 設計主要針對前景即字元部分與背景部分加入干擾，以增加 OCR 辨識的難度，在下文中將整理過去設計 CAPTCHA 時所曾用過的干擾方式，由於方法眾多將分為前景(字元部分)和背景兩部分來介紹：

2.2.1 字元干擾方式

A. Translated (平移)：

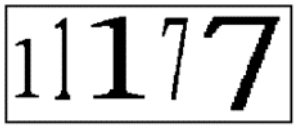
如圖六所示，將 CAPTCHA 中所包含的字元上、下、左、右的移動，使字元不會在特定的範圍內。



圖六：Translated

B. Scaled (縮放)：

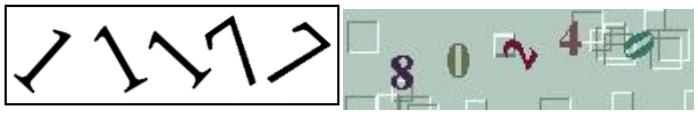
如圖七所示，將 CAPTCHA 中所包含的字元進行 X 軸與 Y 軸方向的伸展或壓縮的動作。



圖七：Scaled

C. Rotated(旋轉)：

如圖八所示，將 CAPTCHA 中所包含的字元進行順時鐘或逆時鐘旋轉。



圖八： Rotated

如圖九所示，在進行旋轉的動作時，其旋轉角度最多不大於 90 °，以 6 和 9 為例，6 順時鐘旋轉 90 °和 9 逆時鐘旋轉 90 °時，如圖九所示，6 與 9 在旋轉之後產生混淆，所以旋轉不能超過。

在過去研究中發現上述 3 種處理的方式，不管是人或電腦都具有很好的辨識能力，故要藉此提高電腦辨識難度是比較無效，因此為了增加 OCR 辨識的難度，便有學者提出如何進行字元扭曲的方式，其中包含 Global Warp 和 Local Warp 兩種[6]。

圖九：「6」與「9」旋轉 90°

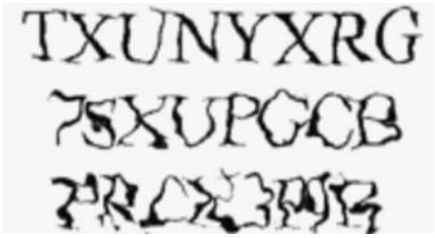
D. Global Warp：

針對整個 CAPTCHA 所有的字元進行扭曲的動作，其方法為隨機選擇一個要位移的區塊，接著進入為指數衰減的低通濾波器(Low-pass filter)。其最後產生的位移的區塊被當作插值用於原來的 CAPTCHA，致使給定的字元彎曲或伸長，而其彎曲程度則於位移距離所含的 Pixel 量成正比，而 Global Warp 運用在文字上的結果則如圖十所示。

圖十：Global Warp 以 reCAPTCHA 為例

E. Local Warp：

如圖十一所示，在 CAPTCHA 內的字元，各自進行扭曲的動作，延著字元的 Pixel 彈性的變形，其利用與 Global Warp 相同的方法，只是將低通過濾器的截止點改為較高的頻率[4]。



圖十一：Local Warp(TXUNYXRG、7SXUPGCB、PRCK3P9R)

F. Overlap(交疊)：

由於旋轉或者扭曲的效果造成在縱軸方向上的交疊，使電腦無法有效的將字元從 CAPTCHA 分割出來。

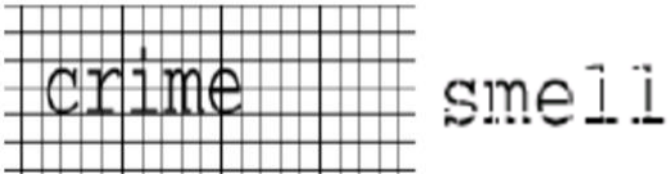
G. Connected：

在 CAPTCHA 中的字元彼此相連，找不到任何直線或曲線能將字元分開。

2.2.1 字元干擾方式

A. Texture(紋路)：

背景加入背景色或前景色(字元顏色)的網狀紋路，如圖十二所示，藉此增加電腦在文字切割上的難度。



圖十二：EZ-Gimpy CAPTCHA

B. Obstacle：

加入直線或弧形，致使與字元交叉，造成電腦無法清楚知道字元所在位置，如圖十三所示，Microsoft 團隊於 2007 年夏天所設計的 CAPTCHA，不僅在當時有效阻擾了電腦，甚至連人類也會誤判直線或弧形為字元部分。

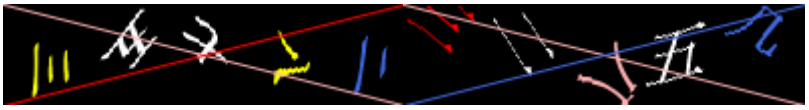
圖十三：2007 年夏天開發的 CAPTCHA(MSN)

C. Color：

藉由設定背景色與字元顏色相近，致使電腦無法有效區分字元位置，如圖十四。

圖十四： PCHome 2010 所使用的 CAPTCHA

文字式的 CAPTCHA 設計方式越來越多，最近甚至加入地域性文化，不再只有英、數字，比如 CAPTCHA 所使用的字元為該國文字，像截至 2011 年，PChome 註冊服務所使用的 CAPTCHA，如圖十五，則是其中的一種應用。



圖十五：PChome 2011 註冊服務所使用的中文 CAPTCHA

2.3 CAPTCHA 攻擊相關研究

自 2000 年 CAPTCHA 發跡以來，已有數位學者針對當時的 CAPTCHA 提出有效的攻擊方法[2][8][12][14]，所以儘管 CAPTCHA 持續更新和演化，但其安全性仍有待考驗和挑戰。

過去研究中主要是針對文字式 CAPTCHA 攻擊，其中攻擊的方式主要分為兩類：

A. 人工辨識

當垃圾郵件、網路購物廣告或交友網等要發送郵件或訊息給使用者時，需要輸入 CAPTCHA 的正確答案時才能發送，若業者未有自動辨識的程式時，便只好雇請「人」來辨識，而此類方式則稱為「human bypass」，目前已有許多網站提供發起交易的平台。

需要破解會在某些特定網站發布需求，比如辨識幾個 CAPTCHA 多少錢等，例如提供「human bypass」的交易平台[15]，1.2 美元要辨識 1000 個 CAPTCHA，以一般人來說，辨識和輸入字串約需 10 秒，所以約需 3 小時才能賺到約 40 元台幣，可以發現其實金額不多，所以通常接受這類工作者，往往來自印度、孟加拉、巴基斯坦、菲律賓等較落後國家。至今仍未有機制能有效防範此類的問題。

B. 機器辨識

一般辨識 CAPTCHA 的工作，主要包括以下幾項工作，前處理、字元分割、字元辨識[2]，當字元分割或辨識效果不好時，若 CAPTCHA 的字串是從字典中擷取的，可進行拼字檢查，判斷目前分割和辨識的結果可能是哪些字串，能進一步找到可能的答案，以增加辨識能力。

2.3.1 前處理(Pre-Process)

A. 灰階(Gray Scale)

如圖十六中的(1)所示，圖片上每一個 Pixel 都是由 RGB 構成，R、G、B 其範圍皆為 0~255，從 RGB 的值可以求出灰階值。座標 (x, y) 的灰階值以 $f(x, y)$ 表示，而灰階值的求法分為以下幾種：

(1) RGB 平均： $f(x,y)=\frac{R+G+B}{3}$ 。

(2) RGB 最大值：由 RGB 中取其最大值當灰階值。

(3) RGB 平均平方合： $f(x,y)=\sqrt{(R^2+G^2+B^2)/3}$ 。

表二為透過以上三種方法，求取灰階值後的進行灰階處理後的結果。

(255,128,222)	(255,32,111)	(205,32,111)	(15,132,101)
.....	.....	.....	.....

(1)Pixel 陣列　　(2)RGB 0-255

圖十六： RGB 全彩原圖

表二： 灰階結果

灰階方法	灰階後圖形
RGB平均	
RGB最大值	
RGB平均平方和	

255	255	20	255
255	40	255	1
1	20	255	20
50	10	111	15
120	30	255	65

255	255	0	255
255	0	255	0
0	0	255	0
0	0	0	0
0	0	255	0

(1)灰階　　(2)二值化

圖十七： 二值化

B. 二值化(Binarization)

如圖十七所示，預設二值化門檻值為 125，將灰階的圖片，根據與二值化的門檻值比對，大於門檻值則視為 255，若小於門檻值則視為 0。

二值化的門檻值，主要由以下幾個方式取得：

(1) 固定門檻值：

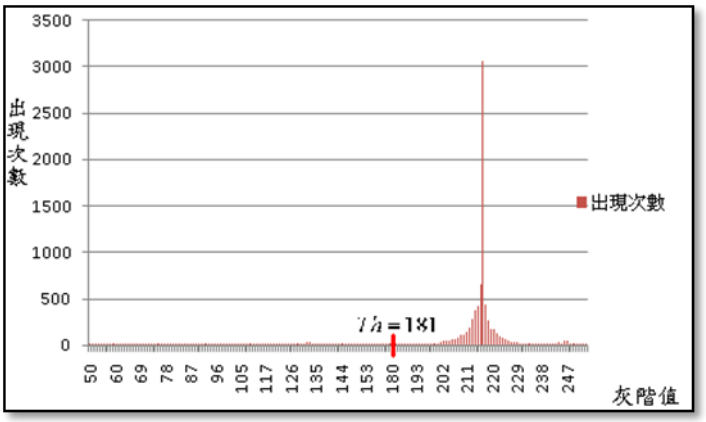
由使用者自己從 0 至 255 的範圍中，挑選一個數值作為門檻值 Th，判斷圖像中每一個點的灰階值，是否大於 Th，是則為亮點，以 255 表示，反之則為暗點，以 0 表示，然而固定門檻值未能針對每種 CAPTCHA 有最佳的二值化效果，故須尋求能分析圖像本身灰階值的分布情況，並提出有效的門檻值。

(2) 灰階值平均：

由於固定門檻值之二值化效果無法適於各種 CAPTCHA，故改利用 CAPTCHA 灰階值的分布情形，由電腦計算適當的值作為二值化之門檻值。假定 CAPTCHA 的高度為 H，寬度為 W，如式 1 所示，$f(x,y)$ 為點 (x,y) 之灰階值，加總每一個點之灰階值再除以該 CAPTCHA 的長和寬，即為灰階平均值 $\overline{f}$ 。

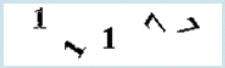
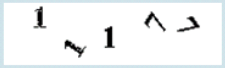
$$\overline{f} = \frac{\sum_{y=0}^{H-1}\sum_{x=0}^{W-1} f(x,y)}{W \times H} \tag{1}$$

求得 $\overline{f}$ 後，需乘於適當的參數值 β，視為其門檻值 $Th = \overline{f} \times \beta$，再由式 2.進行二值化的動作。雖然相對於固定門檻值而言，此方法較具彈性，以及較能代表該圖像之特性，但仍然存在著如何選擇適當之參數值，使二值化能達到最佳化等問題。



圖十八：統計灰階值出現次數之長方圖

表三： 二值化結果整理

灰階方法	固定門檻值	灰階平均值	Otsu門檻值
RGB平均	 *Th* = 100	 *Th* = 206	 *Th* = 160
RGB最大值	 *Th* = 100	 *Th* = 228	 *Th* = 181
RGB平均平方和	 *Th* = 100	 *Th* = 207	 *Th* = 161

(3) Otsu[10]：

每一個灰階值出現的次數，繪於圖十八中之直方圖。Otsu 提出的方法在於如何從灰階之直方圖中選取最適當值，能將圖中分為 D、L 兩個群體，假定目前為最佳門檻值*Th*，直方圖能被分為暗、亮兩個集團，則必須滿足以下兩個條件：

條件一：D 與 L 間的變異數為最大。

條件二：D 內的變異數加上 L 內的變異數為最小。

簡而言之，將 0~255 中每一個數字當做門檻值，並求出其變異數後，再來比較，最小者其所對應的門檻值即為最佳的門檻值，也就是 Otsu 門檻值。

針對以上介紹二值化之門檻值的求法後，對上述的灰階圖像分別處理，其結果整理如表三所示，由表中可以發現 Otsu 的方法是較穩定的，才能有效去除背景部分。

C. 侵蝕(Erode)

侵蝕為型態學影像處理上常用的型態運算之一，假設有 A、B 兩個圖像集合，A 若由 B 侵蝕，則以 $A \otimes B$ 表示，其中 A 為待處理的圖形，B 為遮罩元素。B 為 3×3 的 9 方格遮罩元素，而 B 遮罩在 A 圖型上移動，若遮罩所及之點為黑點，則判斷其周遭 8 點是否為黑點，若其周遭 8 點中有任意點不為黑點時，則將遮罩所及之點視為白色。是否周遭都為黑點則保留該點呢？可依照需求，設定週圍需有多少黑點才保留遮罩所及之點，其範圍為 1~8。

D. 膨脹(Inflation)

膨脹為型態學影像處理上常用的型態運算之一，假設有 A、B 兩個圖像集合，A 藉由 B 膨脹的話，則以 $A \oplus B$ 表示，其中 A 為待處理的圖形，B 為遮罩元素， B

為3×3的9方格遮罩元素，B在A上移動，若遮罩所到之點為黑色，則其遮罩範圍中其他各點則視為黑色，得出處理後的結果。

2.3.2 字元切割(Segmentation)

A. 垂直切割(Vertical Segmentation)

可從圖的任兩邊開始，而圖中則從左向右開始掃描，其詳細運算規則如下所述：

Step1.由左往右移動，偵測縱軸是否有黑點。

Step2.若遇到該縱軸上有黑點時，則設為該字元的起點 P_{Start} 。

Step3.找到起點 P_{Start} 後，則繼續往右尋找，若遇到縱軸沒黑點時，其橫軸往左移動一點，即為該字的終點 P_{End} 。

Step4.重複 Step2 和 Step3，直至圖的邊界。

當在重複 Step2 和 Step3 時，若在 Step2 有找到字元的起點 P_{Start} ，但在進行 Step3 時，直至圖的邊界仍未找到字元的終點 P_{End} 時，其終點則以該邊界代之。

文字式 CAPTCHA 主要是由幾個可辨識的字元組成，而過去為了防止既有的電腦識別技術 OCR 的攻擊，便改變圖型解析度或字的大小，以增加 OCR 辨識的難度，但太小的解析度或太小的字，都容易導致使用者辨識上的困難，顯然此類方法是不適合的，所以在而後的發展中，除了以上的方式外，還包含字體樣式(粗體、斜體等)和字體的改變，以及使用哪些字元集合(英文字母、數字、特殊符號等)，以上種種都在設計 CAPTCHA 扮演著重要的角色。目前常見的 CAPTCHA 設計主要針對前景即字元部分與背景部分加入干擾，以增加 OCR 辨識的難度，在下文中將整理過去設計 CAPTCHA 時所曾用過的干擾方式，由於方法眾多，將分為前景(字元部分)和背景兩部分來介紹：

參、方法

本研究主要分為訓練及執行兩個階段，各個階段所負責的工作簡述如下：

A. 訓練階段：

本研究將透過自動程式從 eBay 網站中擷取約 500 個 CPATHCA，作為訓練樣本，並透過前處理與字元擷取等模組的處理，從所在的 CAPTCHA 中將字元一一分割出來，並標記分割出來的字元是甚麼字，待每一個字元蒐集到足夠樣本時，則將剛剛標記完的字元集作為 BPNN 的訓練樣本，以調整每一個連結(Link)的權重，之後作為辨識字元的依據。

B. 執行階段

此階段將從字元擷取模組中分割出來的字元，轉換成 BPNN 的樣本，並輸入訓練階段所得的 BPNN 中，並就執行完的結果作為辨識的依據。

3.1 前處理模組

前處理模組中包含從現行網站中擷取 CAPTCHA 和將擷取來的影像進行灰階和二值化等處理，本階段主要透過灰階值和二值化處理，以降低色彩造成計算複雜度的問題，並透過此模組處理，將字串從背景中劃分出來，以利下一階段處理。

3.1.1 影像擷取

本研究主要針對 eBay 所使用的 CAPTCHA 做為檢測目標，由於未有目標 CAPTCHA 的設計工具，因此只能從其現行網站中擷取，而自動化擷取影像的步驟如下：

Step1.取得影像網址，如圖十九所示，在註冊頁面上的 CAPTCHA 按右鍵，便可複製該 CAPTCHA 的影像網址：

「https://scgi.ebay.com/ws/eBayISAPI.dll?LoadBotImage&tokenString=fapuqAUAAAA%3D&siteid=0&co_brandId=2」

圖十九： eBay 註冊頁面

Step2.存取該影像網址，並將其影像儲存。

在取得影像網址後，只要透過重複 Step2 即可蒐集到大量的樣本。

3.1. 2 灰階處理

文獻中發現求取灰階值的方式有 RGB 的平均、最大值與平均平方和等三種方法，而在本研究則採用平均平方和的方式以計算灰階值。

在擷取影像中每一個 Pixel 的色彩資訊 R、G、B 後，將 R2 、G2 、B2 加總後平均，再將其平均開根號即為該 Pixel 的灰階值。

3.1. 3 二值化處理

文獻求取二值化門檻值的方式有固定門檻值、灰階平均值與 Otsu 等三種方法[10]，而在本研究則採用 Otsu 的方式以計算其門檻值，利用 Otsu 方法求灰階後圖片之門檻值後，拜訪該灰階圖片上每一個點，若其灰階值大於門檻值時，則設為白色，反之則為黑色。

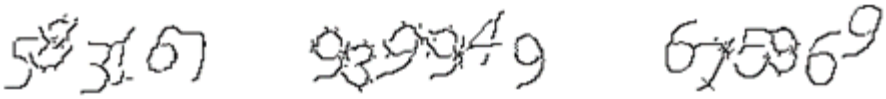
3.2 字元擷取模組

3.2.1 字元分割

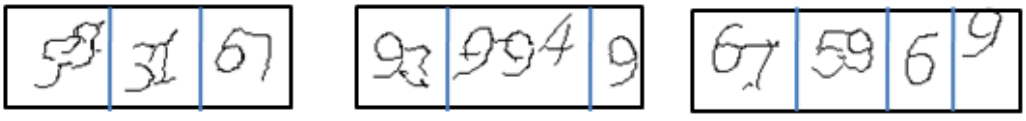
在進行完筆劃細化的處理之前，將先進行初步的字元切割，之所以這麼做主要以下兩點考量：(1)由於字元大小不一的情況下，若直接針對整張圖像進行細化的話，由於筆畫的方向是利用所有圖像中的線段加總平均而來，因此部分較粗的筆畫可能比此平均高時，將導致追蹤至一半的筆畫可能因大於線段平均而被停下來，以致於細化後仍留下部分雜訊，如圖二十所示：(2)採納 Divide-and-Conquer 的方法，透過先分割處理，把原本可分割的字元事先擷取出來，將減少後續慣性處理運算上的成本。基於以上兩點理由，將在細化前先進行字元分割處理。

本研究字元分割處理的部分，主要採用上一章節提及的 Snake 分割，其演算法先求出每一個合法的 Snake(路徑)後，再透過判斷每兩個 Snake 中間是否有字元的顏色(前景色)，如果沒有則把其中一條視為多餘的 Snake，並予以刪除。如圖二十一所示，由於該方法耗費太多時間在計算無意義的路徑上，故本研究提出改良的發法，以減少計算時間，並增加兩個規則，如圖二十二所示以提升目前字元的分割性。

圖二十： eBay CAPTCHA 原始圖像



圖二十一： eBay CAPTCHA 直接細化



圖二十二：eBay CAPTCHA 分割後再細化

根據 Snake 切割的概念，修改而來的首先找出每一個字元的起始位置 xSt 後，再從起始位置的右邊方向尋找可能的路徑(Snake)，若有找到則把起始位置到 Snake 間的黑點

另存成新圖像，再刪除原始圖像中，Snake 與起始位置的所有黑點的部分，演算法包含以下幾個步驟：

(1) 尋找字元起始位置

透過拜訪二值化圖像，以找尋字元的起始位置 xSt，拜訪的過程是由左至右，若發現該列座標出現黑點時，則視為字元的起始位置，並在該座標之後尋找可能的路徑(Snake)。

(2) 尋找可能的路徑(Snake)

尋找可能路徑的方法主要依據第二章所提及之 Snake 分割的 8 項規則，並額外加入 2 項規則：

- 往斜邊方向移動

  除了上、下、左、右外，增加可往斜邊移動的可能，其優先順序改為下>右>左>斜邊(左下)，透過斜邊可移動的原則，以增加字元的分割性。

- 回溯處理

  當 Snake 在移動時，若不能往下移動，而可以往左右移動時，根據移動優先順序原則，Snake 將往右邊移動，在此時記錄該點位置，並記錄從此處開始 Snake 所經過的各點，若發現在當時右轉時，會進入封閉情況，則把過去 Snake 經過的各點刪除，回溯到當時右轉的情形，並在該處改為左轉繼續 Snake 的路徑尋找，當原本路徑往右轉時，發現除了向上不再有移動點時，則回溯自右轉處，並左轉繼續尋找可能路徑。

(3) 儲存 Snake 與字元起始位置間的前景

若尋找到可能路徑時，則將起點位置與路徑間的前景色的地方(字元)儲存成為新的圖像，儲存完之後，再將剛儲存的各點從原本圖像中刪除，然後再尋找下一字元的起始位置，重複上述動作，直到整張圖拜訪完畢。

3.2.2 字元細化

為了解決目前 CAPTCHA 設計中字元連接的問題，所以希望提出筆劃細化處理的方式，使得字元間相連的部分可以有效分離，其概念為自舊有線段中找尋一個寬為一個像元的線段，以取代原來筆畫，而細化出筆畫的線段。

透過判斷每一個筆劃的寬度後，保留該筆劃之中間點，進一步發現如果能順利找的每一個字元的中間線的話，其相連的地方便有機會分開，以利後續字元擷取的處理。

在書寫字元時，可以發現絕大多數的筆畫是由橫、豎、撇、捺所構成，其方向可分為垂直、水平、左斜與右斜等 8 個方向，因此為了找尋字元筆畫的軌跡，便須計算字元裡每一筆畫在不同方向的長度陣列，透過長度或鄰近點決定追蹤的筆畫該往哪裡移動。

細化處理的流程須先計算出筆畫在每一方向的長度陣列，並取 4 個方向中最小者視為該點筆畫應該行徑的方向，透過上述動作，則可求得圖像中每一個筆畫經過的點，其

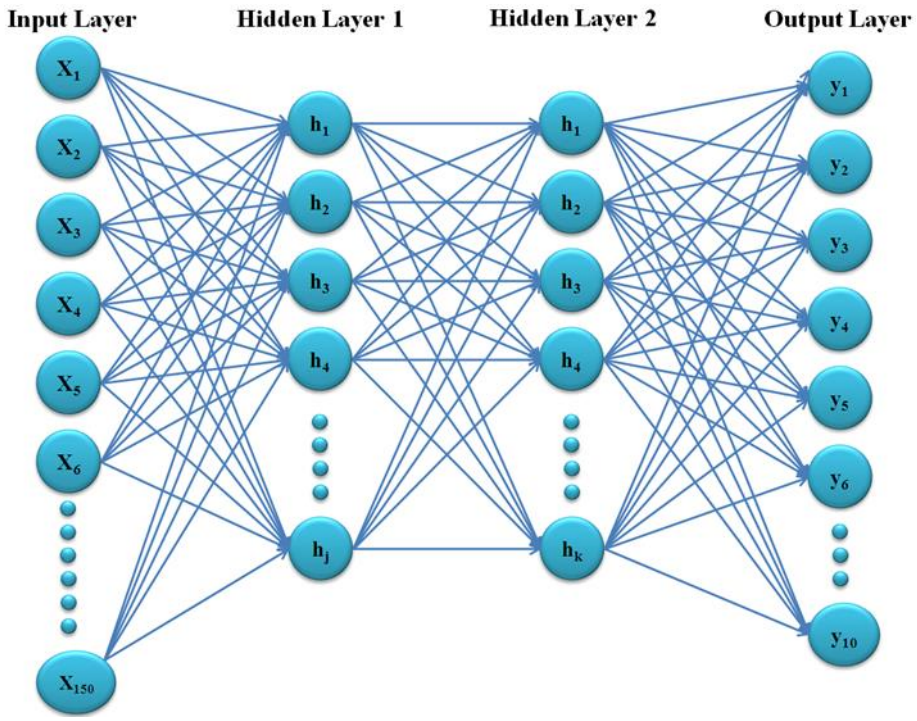
應行經的方向與長度。在得知上述資訊後，則可從中計算筆畫長度的平均*mean*，視為一般正常筆畫的長度，以此作為是否為筆畫經過的判斷依據。在完成上述的前置工作後，即可進行細化處理，其中包含 3 項主要工作：(1)尋找筆畫的進入位置，以進行筆畫追蹤的工作；(2)尋找筆畫可移動點，以判斷目前追蹤到一半的筆畫該往哪一個方向移動；(3)調整下一移動點，透過判斷目前的筆畫是否與下一個移動點相鄰，若否則進行調整，使筆畫得以連貫。細化處理經由分析輸入圖像，可以取得線寬陣列(四個方向寬度最短者)，完成前置工作後才進入細化處理的主要工作，首先嘗試尋找筆畫的起始或結束位置做為進入點，若整張圖都找到符合條件者，則以另一種方法尋找進入點—以可移動點最多者作為進入點，尋找到進入點之後，則開始沿著字元的筆畫開始追蹤。

3.3 BPN 訓練模組

本研究將修改 Neuro.Net 函式庫所提供的 BPN 部分，以進行字元辨識的實作，並透過前測的檢驗結果，以決定各參數之數值設定，研究中所採用的 BPN 架構如圖二十三所示，其中 Output 層個數由欲辨識之字元集決定，比如要辨識數字 0-9，則 Output 便有 10 個 node，若欲辨識英文字母 A-Z，則 Output 就有 26 個 node，依此類推。

在訓練之前須把圖像轉換為一維陣列輸入於 Input 層的各 node 中，並在 Output 層標記每一個 node 的期望值，其中以圖對應到的 node 期望值為最高 ，作為調整權重之依據。BPN 訓練模組透過訓練資料的訓練，學習目前字元的特性，以辨識未來從 CAPTCHA 中擷取的字元。

以往研究多採取單一網路進行辨識處理，但往往不夠客觀，而且若判斷錯誤時，並未有效的機制能修正結果，故本研究分別訓練 3 個 BPN，透過 3 個網路的運作，可以進一步比較每一個網路的結果，若結果不一致時，則可利用排名加權等機制，將可能的誤判調整回來。



圖二十三： 研究之 BPN 架構

由於 BPN 的學習易受旋轉、變形、大小等影響，故需將取得的字元進行正規劃的處理，其中分為兩階段處理，(1)調整校正旋轉的字元，(2)正規化每一個字元為高 15、寬 10 像元的標準大小。

不管在執行或者訓練階段，皆須把擷取的字元轉換成 BPN 的訓練樣本，在此把正規劃後的圖像之每一像素作為其特徵，將正規化圖像之二值化的值，輸入至 BPN 網路中所對應之位置，並設定 Output 層所有 node 的期望值，如下圖為 0，則將對應到 0 的 Output node 之期望值範圍為 ，並將其餘各個 Output node 設較低的期望值。

此外，除了訓練原本圖像正規化後的網路外，額外訓練了原始圖像長、寬 90%和長、寬 80%的網路，希望透過不同的網路發揮互補的作用，以提升字元的辨識力。而 90%和 80%網路之 Input 層個數則依擷取後的圖像之像素數量來決定。

在類神經網路的學習(訓練)過程中，鍵結權重的修正方式可分為圖樣模式(pattern learning)和批次模式(batch learning)：

A. 圖樣模式：

在圖樣模式裡，每當一筆訓練資料被輸入至網路中執行後，便立刻執行鍵結權重的修正運算，此模式可以減少記憶體需求，但運算成本較高。

B. 批次模式：

在批次模式中，等到所有訓練資料都輸入置網路中執行完後，才進行鍵結權重的修正運算，由於需等到全部訓練資料執行完才進行修正運算，故需較大的記

憶體需求，以記錄過去每一筆訓練資料的鍵結權重之修正量。

模式的選擇不同，學習的結果亦不同，因批次模式可以較有效地估測區域梯度函數，故本文將選擇批次模式進行訓練，其實影響 BPN 學習成效的因素眾多，舉凡(1)該用幾層架構？(2)每一層的類神經元的個數該為多少？(3)學習率該設為多少？(4)訓練何時終止？等問題，甚至在每次的學習循環中，訓練資料輸入的順序不同，也會影響學習結果。上述 BPN 參數的問題，目前未存在一個標準答案，故在訓練網路之前，將先進行前測以檢驗每一個參數對 BPN 學習效果的影響，而每一參數的設定，將取決其對辨識檢測樣本的成功率，擇其辨識率較高且較穩定者為之。

根據前測結果，將針對以下參數做設定後，再進行網路的訓練：

- 學習率：0.7。
- 隱藏層：2 層隱藏層，隱藏層 node 個數與輸入層和輸出層呈遞減關係。
- 輸出層 node 期望值：輸出層的目標 node 的期望值為 0.9，其餘 node 則為 0.1。
- 終止條件：以代數作為停止條件的依據，設定為 250 代。

此外，為了解決 3 個網路意見分歧實的問題，提出整合 3 個網路的方式，將利用 3 個網路運算後的結果，再進行加總或配分之運算，目前研究所使用的整合方式分為 Output node 的加總和排名加權兩種：

(1) Output Node 加總法：

將計算後 3 個網路的 Output 的對應 node 的結果進行加總，如將 1 的 Output node 加總的話，則將取每一個網路 Output node 1 的結果進行加總，計算後的結果如表所示，再從加總結果中取其具最大值的 node，作為最後決策。

(2) 排名加權法：

此方法將取每一個網路的前 3 名，並依照第 1 名 3 分、第 2 名 2 分、第 3 名 1 分的方式給分，其餘則為 0 分，最後再加總每一個 Output node 的得分，取其得分最大者作為最後決策，若有得分相同者，則比較 Output node 的加總結果。

肆、結論

本研究提出結合字元細化分割及類神經網路辨識技術，可以有效識別 CAPTCHA 的防護圖案，並利用不同字元分割百分比形成複合辨識網路，可有效提升辨識率。即便針對 eBay 的 CAPTCHA 機制，在重複執行實驗十次，可以發現如表三所示，單一類神經網路字元辨識率平均都可達 94.3%，採用三個網路複合比對則辨識率可以提升至 96.12%，顯示本研究有相當高的穩定性與正確性。因此用於防護網站安全的 CAPTCHA 機制也必需持續更新，並盡可能混用多種混淆技術，讓產生的 CAPTCHA 更難以被人工智慧機制辨識攻擊，達到更安全的網站防護效果。

表三：單一網路與整合式網路的比較

	單一網路			整合式網路	
	100%	90%	80%	加總	排名加權
實驗 1	0.933	0.930	0.901	0.951	0.951
實驗 2	0.942	0.944	0.913	0.953	0.960
實驗 3	0.933	0.96	0.901	0.955	0.960
實驗 4	0.944	0.944	0.919	0.964	0.960
實驗 5	0.946	0.944	0.926	0.960	0.964
實驗 6	0.939	0.944	0.926	0.964	0.969
實驗 7	0.957	0.928	0.924	0.971	0.980
實驗 8	0.946	0.935	0.928	0.966	0.973
實驗 9	0.946	0.955	0.908	0.966	0.966
實驗 10	0.948	0.946	0.895	0.962	0.957
平均	0.9434	0.943	0.9141	0.9612	0.964

[誌謝]

本研究感謝振興發科技透過產學合作案提供部分研究計畫經費協助，並感謝李信陽先生提供實驗協助，潘冠君小姐提供部份前置資料蒐集作業協助。

參考文獻

[1] L. Ahn von, M. Blum and J. Langford, "Telling humans and computers apart automatically," *Communications of ACM*, vol. 47, pp. 56-60, 2004.

[2] A. A. Chandavale, A. M. Sapkal and R. M. Jalnekar, "Algorithm to Break Visual CAPTCHA," *International Conference on Emerging Trends in Engineering and Technology (ICETET)*, pp. 258-262, 2009.

[3] K. Chellapilla, K. Larson, P. Simard and M. Czerwinski, "Computers beat humans at single character recognition in reading-based Human Interaction Proofs(HIPs)," *Proceedings of the 2nd Conference on Email and Anti-Spam (CEAS)*, 2005.

[4] K. Chellapilla, K. Larson, P. Simard and M. Czerwinski, "Designing human friendly human interaction proofs (HIPs)," *Proceedings of the SIGCHI conference on Human factors in computing systems*, 2005.

[5] K. Chellapilla and P. Y. Simard, "Using Machine Learning to Break Visual Human Interaction Proofs," *Neural Information Processing Systems*, 2004.

[6] R. Deriche, "Fast algorithms for low-level vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence ,* vol. 12, pp. 78-87, 1990.

[7] R. Gossweiler, M. Kamvar and S. Baluja, "What's up CAPTCHA?: a CAPTCHA based on image orientation," *Proceedings of the 18th international conference on World wide web*, 2009.

[8] C. W. Lin, Y. H. Chen and L. G. Chen, "Bio-inspired unified model of visual segmentation system for CAPTCHA character recognition," *IEEE Workshop on Signal Processing Systems*, pp. 158-163, 2008.

[9] G. Mori and J. Malik, "Recognizing objects in adversarial clutter: breaking a visual CAPTCHA," *IEEE Computer Society Conference* on *Computer Vision and Pattern Recognition*, pp. I-134-I-141, vol.1, 2003.

[10] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, pp. 62-66, 1979.

[11] C. Pope and K. Kaur, "Is it human or computer? Defending e-commerce with Captchas," *IT Professional*, vol. 7, pp. 43-49, 2005.

[12] S. Sharma and N. Seth, "Survey of Text CAPTCHA Techniques and Attacks," *International Journal of Engineering Trends and Technology (IJETT)*, vol. 22, pp. 240-245, 2015.

[13] A. M. Turing, "Computing machinery and intelligence," *Mind A Quarterly review of Psychology and Philosophy*, vol. 49, pp. 433-460, 1950.

[14] J. Yan and A. S. El Ahmad, "CAPTCHA Security: A Case Study," *Security & Privacy,*, vol. 7, pp. 22-28, 2009.

[15] http://www.freelancer.com/projects/by-tag/captcha-human-bypass.html (2011).

[16] 劉力仁、黃敦硯， "搶票程式癱瘓台鐵網路 設計者到案,"*自由時報*, 2005.

[作者簡介]

侯佳利博士於 2007 年取得國立中央大學資訊管理學系博士學位，於 2007 年起任教於國立東華大學資訊管理學系擔任助理教授並兼任企業資源規劃中心執行秘書。研究興趣包含資訊安全、人工智慧、財務工程、企業資源規劃、商業智慧、巨量資料分析及物聯網等領域。