樣本比對之中文手寫辨識系統
Online Handwriting Recognition by Template Matching

蔡奇峰
Tsai Chi-Feng

中央研究院資訊科學所
Institute of Information Science
Academia Sinica
chi@iis.sinica.edu.tw

馬自恆
Ma Tze-Heng

中央研究院資訊科學所
Institute of Information Science
Academia Sinica
mada@iis.sinica.edu.tw

摘要

這裡所使用的中文手寫辨識方法是以樣本比對爲基礎。首先將使用者輸入的字跡正規化，再和資料庫預存的字跡樣本以 Elastic matching 比對。

關鍵字：中文手寫輸入, 樣本比對, 彈性比對

Abstract

In this paper, we present a chinese handwriting input system. Our approach is based on the template matching method.

Keywords:Chinese OLOCR, Template matching, Elastic

一、簡介

近幾年來各種電腦技術急速地進步，不管是硬體或軟體都和以往大不相同。尤其是電腦的硬體設備，進步的結果使得電腦的價位不斷下降，功能不斷提升。雖然功能越來越強大、複雜，但是使用電腦的方式卻越來越簡單。因此電腦從專業的機器變成幾乎是每個家庭都會擁有的家用電器。使用電腦的人也就不再拘限於某些人。我們可以用電腦來做研究、編輯文章、儲存資料、做美工、連上網際網路（Internet）．．．．等等。而這些工作都離不開一件事，就是輸入資料，尤其是輸入中文資料。

輸入中文資料的方法有很多，比如說使用各種輸入法，像倉頡、注音．．．．等等。但是這些輸入法都必須經過長短不一的學習時間，而且使用者必須要先熟悉電腦鍵盤上按鍵的排列。這種學習過程對於使用者來說是一個不小的負擔。既然電腦普及的程度已經深入各種不同行業的人，所以中文輸入也必須要有一個大眾化的方式讓各種使用者可以更容易輸入資料。用手寫的方式輸入中文就是最自然又不必經過太多額外學習的方法之一。這也是所有人都可以很快適應的方法。

此外，爲了讓使用者可以快速寫入筆跡，我們認爲能接受潦草的輸入並允許連筆、具備高效能的學習功能（讓使用者可以建立自己的辨識環境）、具親和力的使用者介面等等都是一個好的辨識系統必備的條件（當然，一個容易操控的手寫筆也很重要）。我們以樣本比對的觀念爲主發展出一套中文手寫辨識系統－土狼毫。它具備上述幾項條件。而且只要是可以支援滑鼠共用的手寫筆或數位板都可以當作土狼毫的字跡輸入設備。使用者可以依個人習慣選擇輸入設備。

當然，一筆劃寫完一個字是否可以增

加書寫的速度有時是因人而異的，但是潦草的字跡卻是快速書寫不可欠缺的條件。系統必須同時擁有前兩項特性才能讓使用者不改變原本的習慣而可以快速地書寫中文。也就是說，系統接受使用者在書寫一個字的時候可以在任何地方連筆，不必爲了要書寫工整的字體而改變習慣或放慢速度。除此之外，系統必須可以學習使用者潦草的筆跡。在這一方面，我們讓系統直接記下使用者所輸入的筆跡，並把它當作資料庫新的樣本，當一個字同時具有多個樣本時，系統還會根據樣本被比對到的機率來自動刪除資料庫中不需要的字跡樣本。經由這樣不斷的學習，系統辨認的正確率就會提高，辨認速度也會加快。

二、辨識核心

系統的方塊圖如圖一。當系統擷取使用者所輸入的筆跡之後，會先進行筆跡的前置處理。首先會將字跡連成一條曲線，再將這一條曲線正規化成 300 段小向量。每一段小向量都是 16 個固定角度向量的其中之一[1]。然後把小向量合併成數個大向量。最後把這些大向量和資料庫的樣本以 Elastic matching[2] 比對，依序取得最接近的字作爲辨識結果（最多四個）。辨識結束之後我們可以選擇是否要求系統學習這個筆跡及辨識結果。

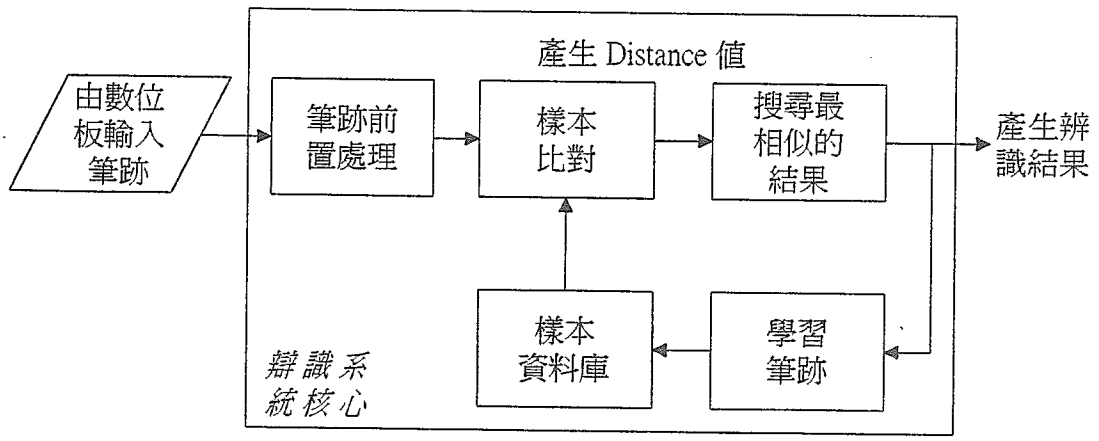
2.1 由數位板輸入筆跡

當使用者在數位板上寫入筆跡時，辨識系統會抓取筆跡上代表 X 軸和 Y 軸的座標位置。我們以 (Xi, Yi) 來表示筆跡上第 i 個點，i = 1, 2, ... 。

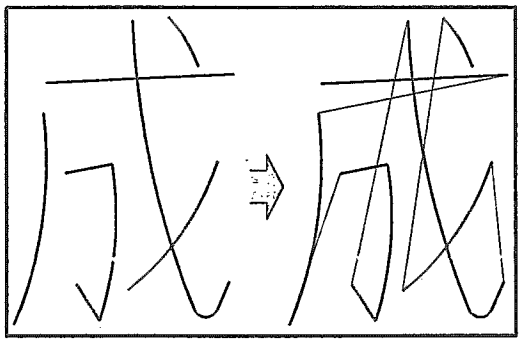
2.2 筆跡的前置處理

系統不管使用者所輸入的筆跡是否分段書寫還是一筆連到底，系統都會捨棄連筆和斷筆的資訊，直接將輸入的筆跡以一條連續的曲線 S 來看待[1]。如圖二。

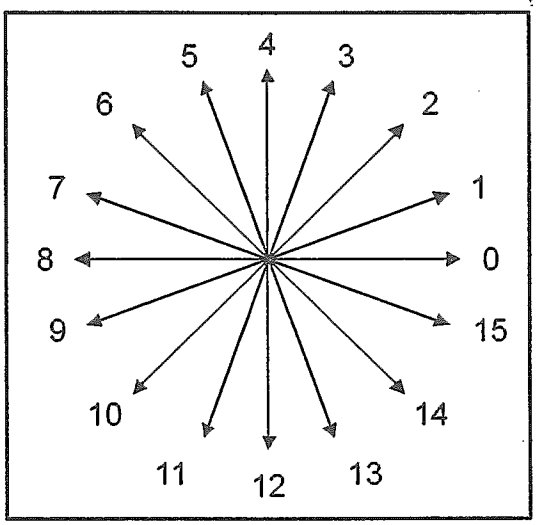
接著將曲線 S 等分成 300 小段，每一小段都是一個向量。然後再將每一個小向量對應到代表固定方向（如圖三）的一個碼[1]。所得到的就是經過正規化的（Normalization）300 段小向量。



（圖一）系統方塊圖



（圖二）將分段的筆跡連成一個曲線



（圖三）由 360 度等分出來的 16 個方向

這 300 段小向量我們以下列式子表示：

$$V = \{v_1, v_2, \cdots\cdots v_{300}\}$$

接著我們將 V 中的小向量合併成一組較大的向量，以下列式子表示之：

$$W = \{w_1, w_2, \cdots\cdots w_k\}$$

其中每一個大向量都是由下列式子所得：

$$w_i = \sum_{j=l}^{l+m} v_j$$

也就是我們取一段$\{v_l, v_{l+1}, \cdots, v_{l+m}\}$來構成一個大向量，且其中任何一對小向量之間的夾角不大於某一θ值。θ值若越小，所得之大向量會較準確，但是大向量數目也會增加，影響計算速度。因此在這個系統中，我們取θ值如下：

$$\theta = \frac{3}{8}\pi$$

依據我們的樣本資料來看，平均每一個字大約會被分成 23 個大向量。這些大向量就是使用者所輸入的筆跡經過前置處理之後所得的樣本。

2.3 樣本比對

我們以彈性比對（Elastic matching）的方式將使用者的筆跡大向量和樣本資料庫中的大向量做樣本比對。

彈性比對是一個廣泛被使用在曲線比對上的方法，其概念和演算法中的 Dynamic programming [3]並無二致。

我們以下列式子表示使用者輸入的筆跡被轉換成大向量的資料：

$$Y = \{y_1, y_2, \cdots\cdots y_n\}$$

資料庫中的大向量樣本以下列式子表示之：

$$X = \{x_1, x_2, \cdots\cdots x_m\}$$

然後我們以一個陣列 M 來儲存上述兩組大向量之間的差異，確切地說，M[i][j] 表

示曲線 $X' = \{x_1, \ldots x_i\}$ 和 $Y' = \{y_1, \ldots y_j\}$ 的差距。我們以下列式子表示 M[i][j]：

$$M[i][j] = \min\Big\{ M[i-k][j-l] + f\Big(\sum_{p=i-k+1}^{i} x_p, \sum_{q=j-l+1}^{j} y_q\Big) + g(x_{i-k+1}, x_{i-k+2}, \cdots x_i) + g(y_{j-l+1}, y_{j-l+2}, \cdots y_j) \Big\}$$

在上述式子中，f 爲兩個向量的差距，g 爲將一組向量相加所必須付出的代價。f 與 g 均大致由參與向量之長度與其間之夾角決定。實際在運作的時候，爲了節省運算時間，k 和 l 都是很小的常數。

簡單地說，整個辨識核心將一組點座標轉換成一組大向量 Y，再將 Y 與樣本資料庫中的每一個 X 做樣本比對。比對完之後，就取其中差異最小的幾個字當作辨識結果輸出。在這樣的架構之下，學習是一項非常簡單的工作。當使用者認定 Y 所代表的筆跡應該是 C_Y 而不是辨識核心產生的 C_X 時，我們只要把 Y 和對應的 C_Y 直接存入樣本資料庫中即可。因此樣本資料庫中同一個字可能會對應到一個以上的樣本。爲了避免資料庫無限制地增長，我們設立一個自然淘汰的法則，當某一個樣本很少被使用到的時候，系統會自動將它從資料庫中刪除。

當使用者使用這個辨識系統的時間越長，系統對於該使用者的辨識率就越好。即使使用者寫入的筆跡很潦草，對於系統也沒有影響。

三、使用者介面的一些特色

我們將整個實驗實做成一個可以真正應用在中文輸入的手寫辨識系統（我們稱這整個系統叫做土狼毫）。這個辨識系統最終的目地是要將使用者的筆跡完全取代樣本資料庫中原有的筆跡資料。因此一個便利的學習環境就更顯得重要。

使用者大致可以用兩種方式進行更正與學習。較常用的可能是以注音符號來輸入使用者想要更正或學習的字。顧及使用者可能不熟悉使用鍵盤輸入注音，或是在沒有鍵盤的情況下操作。在實做的辨識系統中是以手寫的方式輸入注音符號的筆跡，系統再對注音符號的筆跡加以辨識。當接受一個注音輸入之後，系統會顯示辨認候選字與相似音，所謂相似音就是針對拼音不準確的使用者提供的便利工具。例如我們輸入 ㄑㄧㄣ，系統會自動在相似音的地方顯示 ㄑㄧㄥ。使用者如果發現拼錯音可以直接在相似音選擇一個拼音，不必重新輸入筆跡。

在注音輸入的部分，我們還請 15 個人針對常用字輸入他們認爲正確的拼音。根據這份資料，我們把比較容易被念錯的音也加入注音的候選字中。比如說 “旖” 這個字常會被誤唸成 “ㄑㄧˊ”。我們就把 “旖” 加入 “ㄑㄧˊ” 的注音候選字中，當使用者輸入“ㄑㄧˊ” 的時後也會顯示出這個字，但是在顯示的時候會以特殊的顏色來代表該字是被誤唸的。爲了避免混淆該字的正確念法，當使用者從注音候選字中選取被誤唸的字的時候，系統會自動顯示正確的唸法。

如果使用者面對一個不知如何發音的字，還可以利用難字檢索的工具找到該字。一般系統中通常也有提供類似的字典檢視工具，但都是以字的部首以及筆劃來做搜尋，常常搜尋出來的字數都很多，令人眼花撩亂。因此在我們的系統中是利用

一些直覺的字根及筆劃的方式來縮小檢索的範圍。直覺的字根像是 金、木、水、火、土、雨、田、米..... 等等。使用者看到一個字的時候就可以直覺地決定這個字包含了哪些直覺的字根。比如 "慄"，我們可以直覺地看到這個字有一個 "心" 以及 "木" 的直覺字根。所以我們只要在難字檢索工具中選取這兩個直覺字根就可以找到該字。

四、測試數據

雖然土狼毫可以接受一萬多字的中文字集，但在初期的資料庫只建有 3800 個較常用的中文字。這樣做的理由有兩個；一是這個資料庫將被使用者的字跡替代，它的存在只是聊備一格，讓使用者在初期使用的階段不致於有太大的挫折感。其二是根據統計，3800 個字可以涵蓋 99% 以上的使用率。我們相信使用手寫中文輸入時所用的字彙不會超過 7000 個。而且即使使用者要輸入的字不在 3800 之中，也可以透過學習功能將字跡加入資料庫中。

我們請十位測試者寫入 3800 個字，取其較具代表性的的筆劃順序建立一個資料庫。以此資料庫與原有十份原始資料作比對，平均的正確率爲 91%。另外還有三位測試者將 3800 字寫入兩遍，我們取其中之一爲資料庫與另一組筆跡進行比對，正確率在 97% 以上。由此可以看出使用者若以自己的筆跡建立資料庫，在辨識的時候可以獲得很好的結果。

對於不同的中文手寫辨識系統進行比較是相當困難且具有爭議性的。我們能提供的是觀察土狼毫與我們測試過的產品比較，結果是土狼毫有較佳的學習能力。

初期在使用土狼毫時，辨識速度約爲 2 字/秒 （在 Pentium 120 的個人電腦上），當使用過一段時間之後將資料庫中可辨識的字彙增加到 5400 個字時，仍然感覺不到擾人的延遲。一般人手寫輸入的速度大約不會超過每分鐘 60 個字。因此對於裝有 Pentium 100 以上的個人電腦，土狼毫的執行速度將不構成問題。值得一提的是，當 CPU 的速度大幅增長，在做樣本比對時我們可以放鬆若干參數使辨識率更加提高。

五、結語

土狼毫的主要貢獻在於提供一個新的思維方向，也就是重新檢討怎麼樣的中文手寫輸入系統才是一個符合使用者需要的系統。誠如在[4]中作者指出，在手寫辨識的領域中，存在兩中不同的方法，一個是以樣本比對（Template matching）爲主，另外一個則是以結構分析（Structure analysis）爲主。在以往因爲受限於電腦的運算速度，所以中文即時手寫辨識系統要採用完全的樣本比對是很困難的。但是現在由於個人電腦的計算能力不斷加強，計算速度也越來越快，所以我們可以在土狼毫的辨識核心中大量地使用樣本比對，而且土狼毫會將輸入的筆跡連成一條曲線看待，所以可以達到完全連筆的效果。如果系統以樣本比對的方法爲主，再加上允許使用者連筆書寫，則辨識系統比較可以忍受潦草的字跡，使用者輸入的速度就可以加快。如果再經過辨識系統學習使用者所輸入的筆跡，要適應使用者個人化的字跡應該都不是問題。這裡指的個人化字跡就是說使用者習慣於某一種筆順的書寫，或是潦草的寫法。因爲這種系統要學習使用者的筆跡是很容易的，它只要把使用者所輸入的筆跡直接存入資料庫中即可。我們相信土狼毫所引用的概念將會是日後中

文手寫辨識的發展趨勢。但它同時也有受筆順限制的特性。我們不認爲這是一個嚴重的缺點，因爲藉著好的學習功能，使用者可以很快地建立個人的筆跡資料庫。根據我們的觀察，一個固定的使用者所寫出的筆順具有相當高的一致性。而且我們還可以在初期的資料庫中針對容易有筆順問題的字多建立幾個不同筆順的樣本來彌補剛開始使用土狼毫的不足。

如果辨識系統是採用結構分析的方法，系統要從字跡中淬取出特徵，然後用系統的規則來判斷這些特徵以決定這個字跡要對應到哪一個字。好處是系統比較不會有筆順的限制，甚至可以沒有筆順的限制。但以目前來說，當使用者所輸入的字跡比較潦草的時候，要從字跡中淬取出特徵就比較不容易了，因此也比較容易判斷錯誤。這種系統在學習比較潦草的字跡的時候也比較不容易成功。

六、參考文獻

[1] K.Yoshida and H.Sakoe. Online handwritten character recognition for a personal computer system. IEEE, 1982.

[2] C.C.Tappert. Cursive Script Recognition by Elastic Matching. IBM J. RES. DEVELOP. Vol.26 No.6, November 1982.

[3] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. The Design and Analysis of Computer Algorithms. Addison-Wesley, Reading, Ma, 1974

[4] Toru Wakahara, Hiroshi Murase and Mazumi Odaka. On-Line Handwriting Recognition. Proceedings of the IEEE. Vol.80 No.7, July 1992