TẠP CHÍ KHOA HỌC TÍNH TOÁN VÀ ĐIỀU KHIỂN TẬP V(1989), SỐ1 (11-15)

MỘT SỐ PHƯƠNG PHÁP VÀ THUẬT TOÁN NHẬN DẠNG CHỮ

Hoàng Kiếm
Giang Vũ Thắng
Nguyễn Văn Điện
Phòng Nhận dạng, Viện Khoa học tính toán và Điều khiển

Nhận dạng chữ (chữ cái, chữ số, chữ dấu...) là một lĩnh vực đã được quan tâm nghiên cứu và ứng dụng từ nhiều năm nay theo hai hướng chính

- Nhận dạng chữ in hoặc chữ đánh máy để phục vụ cho việc tự động hóa đọc tài liệu, tăng nhanh tốc độ và chất lượng nhập thông tin vào máy tính trực tiếp từ các tài liệu in hoặc đánh máy.
- Nhận dạng chữ viết tay (với nhưng mức độ ràng buộc khác nhau về cách viết và kiểu chữ ...) phục vụ cho các ứng dụng đọc và xử lý các chứng từ, hóa đơn, phiếu ghi, bản viết tay chương trình,v.v.

Cho đến nay, đã tồn tại nhiều thuật toán nhận dạng chữ có thể phân thành các nhóm phương pháp sau đây:

- Phương pháp chuẩn hóa chữ và đối sánh sơ cấp: theo phương pháp này, ảnh biểu diễn chữ thu được qua các thiết bị số hóa cần được chuẩn hóa về kích cỡ, vị trí để có thể đối sánh với các biểu diễn chữ đã được lưu sẵn trong máy, phương pháp nầy khó áp dụng để nhận dạng chữ viết tay.
- Phương pháp biến đổi biểu diễn chữ và đối sánh thứ cấp: với các thuật toán biến đổi khác nhau, biểu diễn ban đầu của chữ được chuyển sang biểu diễn mới (không bị ảnh hưởng nhiều bởi nhiều và tương đối bất biến đối với kích cỡ, vị trí của chữ). Quá trình đối sánh các biểu diễn mới này được gọi là đối sánh thứ cấp. Nhược điểm của phương pháp này là độ phức tạp của thuật toán lớn, ảnh hưởng đến tốc độ nhận dạng chữ.
- Phương pháp trích chọn dấu hiệu đặc tả chữ và đối sánh cấu trúc: Đây là phương pháp có nhiều triển vọng nhất hiện nayđể xây dựng các hệ nhận dạng chữ in cũng như chữ viết. Tuy đã có nhiều thuật toán trích chọn dấu hiệu đặc tả nhưng thường là phụ thuộc vào kiểu chữ cụ thể.

Trong bài này, chúng tôi sẽ giới thiệu một số phương pháp và thuật toán nhận dạng chữ đã được thiết kế và cài đặt trong các hệ VIẾT-IN và OCR - định hướng vào việc nhận dạng các kiểu chữ mở rộng với việc bổ sung cơ chế học linh hoạt nhằm nâng cao khả năng thu nhận thông tin, tăng độ chính xác của hệ thống.

I. Sơ đồ tổng quát của hệ nhận dạng chữ (OCR)

Về cơ bản, một hệ OCR bao gồm các khối chính như sau:

- Khối xử lý sơ bộ,
- Khối tách chữ,
- Khối nhận dạng chữ,
- Khối phục hiện chữ (nội dung, hình thức..)

Trong phần này, chúng tôi sẽ phân tích một số giai đoạn xử lý và điểm qua một số thuật toán được sử dụng trong từng giai đoạn.

1. Giai đoạn xử lý sơ bộ

Giai đoạn xử lý sơ bộ văn bản là giai đoạn vô cùng quan trọng, nó ảnh hưởng trực tiếp tới độ chính xác của thuật toán nhận dạng, tuy nhiên nó cũng chậm lại tốc độ xử lý của hệ thống. Cho nên, tùy từng trường hợp, tùy vào chất lượng quét văn bản của Scanner mà chúng ta có thể chọn một hoặc một vài thủ tục con trong khối này. Thậm chí nếu cần ưu tiên tốc độ xử lý và nếu việc quét của Scanner đạt chất lượng cao thì chúng ta có thể không cần phải áp dụng giai đoạn này.

Chúng tôi sẽ liệt kê dưới đây một số bước cơ bản.

a. Khử nhiễu : Nhiễu là tập các điểm sáng thừa trên Scanner file. Các điểm sáng duoc thêm vào anh trong quá trình quét tài liêu cua Scanner. Nhiễu xuất hiện do hai nguồn gây nhiễu sau:

- Nhiễu hệ thống : Do khuyết tật của Scanner,
- Nhiễu ngẫu nhiên: Do chất lượng của giấy và mực inkhông đảm bảo, hình thành một cách ngẫu nhiên các vùng tối trên giấy, gây nhầm lẫn với dấu hiệu thật của ký tự.

Với nhiễu loại 1,chúng ta cần phải quan sát một cách thật cẩn thận để có thể định vị được vùng nhiễu của thiết bị, tránh việc đưa văn bản vào vùng đó, và tránh việc xử lý các vùng này.

Với nhiễu loại 2, chúng tôi xây dựng hệ mềmnhằm phát hiện các điểm sáng cô lậpvà loại chúng ra khỏi quá trình xử lý sau này.

b. Làm trơn biên chữ

Đôi khi do chất lượng quét tồi, các đường biên của chữ không còn giữ được dáng điệu trơn tru ban đầu của nó và hình thành các đường răng cưa giả tạo. Trong trường hợp đó chúng tôi áp dụng một số thuật toán làm trơn đường biên (Curve Smoothing) để khắc phục.

c. Làm đầy chữ

thủ tục làm đầy chữ được áp dụng đối với các ký tự bị đứt nét một cách ngẫu nhiên. Việc đứt nét gây khó khăn cho thủ tục tách chữ, dễ nhầm hai phần của một ký tự liên thông thành hai ký tự riêng biệt, gây sai lầm trong thủ tục nhận dạng.

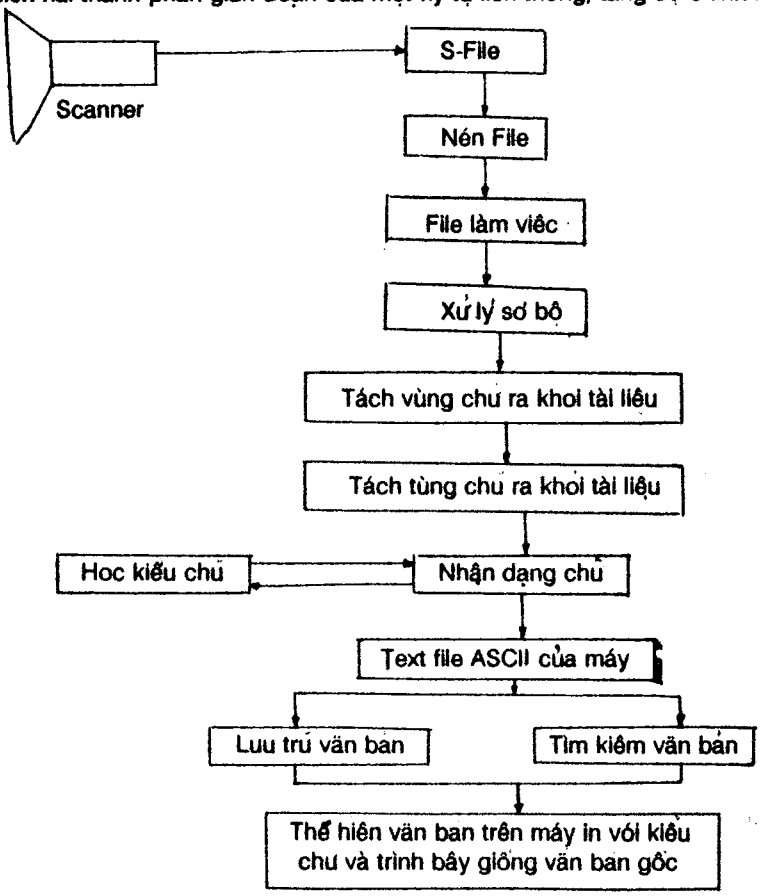
Nội dung cơ bản của thủ tục làm đầy chữ là tìm các điểm biên của ký tự. Nếu gọi P(i,j)là pixel tại toạ độ i,j thì P(i,j) được gọi là điểm biên nếu

P(i,j) = 1 AND ($\exists 1 \in V_{ij}$) AND ($\exists 0 \in V_{ij}$)

Ở đây $V_{ij} = L_{ij} \setminus P_{ij}$

$L_{ij} = \{ P(u,v) / |u-i| < \varepsilon, |v-j| < \varepsilon \}$

và tại pixel biên P(i,j) này chúng ta sẽ làm nó "nở" ra về mọi phía với kích thước . Việc nở đường biên sẽ nối liền hai thành phần gián đoạn của một ký tự liên thông, tăng độ chính xác trong thủ tục nhận dạng.



d. Làm mảnh chữ

Thủ tục làm mảnh chữ là một thủ tục quan trọng, nhằm phát hiện khung xương (Skeleton) của một ký tự, giữ lại cốt lõi, bất biến của một chữ, bằng việc bỏ dần các dirm cực biên. Độ phức tạp của thật toán làm mảnh chữ tỉ lệ với độ dày của chữ. Thuật toán bao gồm 2 bước, bước thứ nhất nhằm loại bỏ các pixel thoả mãn các điều kiện sau:

1. $2 \leq B_{ij} \leq 6$
2. $A_{ij} = 1$
3. P(i,j+1)*P(i+1,j)*P(i-1,j) = 0
4. P(i,j-10*P(i+1,j)*P(i,j+1) = 0

Đó là nhưng pixel biên bên phải, pixel biên trái hoặc là pixel góc trên trái của một ký tự.

Bước thứ hai nhằm loại bỏ các pixel thoả mãn điều kiện sau dây:

1. $2 \leq B_{ij} \leq 6$
2. $A_{ij} = 1$
3. P(i,j-1)*P(i+1,j)*P(i-1,j) = 0
4. P(i,j-1)*P(i,j+1)*P(i-1,j) = 0

Ở đây $B_{ij} = \sum_{\substack{v=j-1 \\ v \neq j}}^{j+1} \sum_{\substack{u=i-1 \\ u \neq i}}^{i+1} P_{uv}$

Đó là những pixel biên trên, pixel biên trái hoặc các pixel góc dưới bên phải của một ký tự.

Aij là số các cặp 01 gặp trên đường nối các pixel lân cận của Bij,theo chiều quay của kim đồng hồ.

Thủ tục làm mảnh chữ làm dễ dàng việc tìm kiếm các điểm đặc biệt của chữ (như điểm nút, điểm ngã ba, ngã tư), làm dễ dàng việc tìm kiếm các dấu hiệu đặc trưng để nhận dạng. Tuy nhiên đây là một thuật toán có độ phức tạp cao, chỉ áp dụng được trong trường hợp thật cần thiết.

e. Xoay văn bản đi một góc lệch

Do việc đưa trang tài liệu vào Scanner không cẩn thận hoặc do sự cố in ấn, các hàng chữ đã lệch đi so với lề chuẩn một góc khá lớn α . Việc đưa văn bản vào ở vị trí lệch sẽ làm khó khăn cho thủ tục tách chữ, đôi khi không thể tách nổi. Trong trường hợp như vậy chúng ta phải tính lại toạ độ cho các pixel bởi công thức

$x' = x \cos\alpha - y \sin\alpha$

$y' = x \sin\alpha + y \cos\alpha$

2. Giai đoan tách chữ (Character Segmentation)

Giai đoạn tách chữ ra khỏi văn bản là giai đoạn quan trọng và cần thiết. Chỉ khi nào tách và cô lập được đúng một chữ đơn ra khỏi tổng thể văn bản, thì mới có thể nhận dạng đúng được ký tự đó trong giai đoạn nhận dạng. Dưới đây chúng tôi sẽ trình bày một số thuật toán tách chữ, từ đơn giản đến phức tạp, từ thuật toán có độ phức tạp tính toán thấp đến thuật toán có độ phức tạp cao. Tùy vào font chữ cụ thể, tùy vào yêu cầu độ chính xác của bài toán mà ta có thể chọn một thuật toán thích hợp.

2.1. Tách chữ theo chiều nằm ngang và thẳng đứng

Khác với chữ viết tay, kích thước và kiểu chữ có độ tự do cao, chữ in phải tuân theo một số quy định của ấn loát. Các chữ phải nằm trọn trong một khuôn, nên việc cô lập một ký tự đơn có thể đồng nhất với việc tìm ra khuôn cua chữ đó, tại vị trí của nó trong văn bản.

Tách chữ theo chiều nằm ngang và thẳng đứng là tìm một hình chữ nhật, có các cạnh thẳng đứng và nằm ngang, chứa trọn một ký tự ở trong.

Tách chữ theo kiểu này đơn giản và thực hiện nhanh, tuy nhiên trong một số trường hợp, việc tách chữ theo chiều ngang và thẳng đứng to ra kém hiệu quả, có khi không thực hiện nổi.

Chúng ta hãy xem kiểu chữ nghiêng sau đây:

A B C

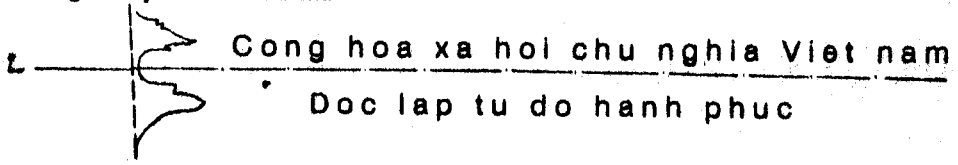
2.2. Tách chữ dựa vào tính liên thông của chữ

LABEL : = {ϕ};
FOR I:=1 to SIZE DO
BEGIN
Xác định các khoảng liên thông TI1, TI2,...,TInI trên hàng I;
FOR j:=1 to nI DO IF$\exists$K để (Ti-1,I1$\cap$Tij $\neq \phi$) $\wedge$ (Ti-1,I2$\cap$Tij $\neq \phi$)$\wedge$
... $\wedge$(Ti-1,Ik$\cap$Tij $\neq \phi$) THEN
BEGIN
Label(Tij) = min (label(Ti-1,I1),...,label(Ti-1,Ik)
FOR F:=1 to k DO label(Ti-1,If):=label(Tij);
END
ELSE BEGIN
label(Tij):=max(LABEL)+1;
LABEL:=LABEL$\cup${label(Tij)};
END;
END;
TRong dó
LABEL: Tập các nhãn hiện đang được gán và
label(T11),label(T12),...,label(T1n1) được gán ban đầu là
1,2,3,...,n1 một cách tương ứng.

2.3 Tách chữ dùng lược đồ độ sáng

trong một số trường hợp, với các font chữ đặc biệt, các dòng trong văn bản được tách nhau một dấu cách khá rõ ràng, song trên thực tế việc tìm đường phân cách hai dòng theo nghĩa thông thường cực kỳ khó khăn.

Chúng ta hãy xét font chữ sau



Trong trường hợp này, không thể tìm đường phân cách L theo nghĩa cũ mà phải hiểu là đường phân cách về cơ ban với số điểm cắt 2 dòng ít nhất. Khi đó ta phải xây dựng lược đồ độ sáng của các dòng chữ và đường nằm ngang tại đáy của thung lũng lược đồ chính là đường phân cách cần tìm

II. Các thuật toán nhận dạng chữ in hoặc đánh máy

Giai đoạn quan trọng nhất, quyết định độ chính xác của hệ thống chính là giai đoạn nhận dạng riêng từng chữ. Sau đây chúng tôi xin trình bầy một số thuật toán đơn giản, nhưng khá hiệu quả của chúng tôi.

1. Thuật toán đối sánh từng điểm xuất phát từ trọng tâm

Đây là một thuật toán đơn giản. Sau khi cô lập chữ ra khỏi văn bản, trọng tâm(TT) của chữ được tính toán và được xác định tọa độ. Tiếp đó chữ mới và chữ chuẩn được đối sánh với nhau từng pixel một theo chiều từ trọng tâm ra ngoài biên. Các hình vành khăn lồng nhau có trọng tâm tạo thành các lớp pixel có cùng trọng số:

Khoảng cách giữa hai pixel

$$DIS(x,x') := \begin{cases} 0 \text{ nếu } x = x' \\ \omega_x \text{ nếu } x \neq x'; \omega_x \text{ là trọng số của lớp chứa x.} \end{cases}$$

$$DIS(X,X') := \sum_{x \in X,\, x' \in X'} DIS(x,x')$$

là khoảng cách giữa 2 ký tự. Ký tự x được coi là ký tự x' nếu $DIS(x,x') \leq \varepsilon$ (ε là một hằng số định trước).

2.Thuật toán nhận dạng chữ dựa trên việc đối sánh các dãy điểm cắt dọc và ngang

Thuật toán đối sánh pixel chúng tôi trình bày trên,thực hiện khá nhanh, song nếu chất lượng quét của Scanner hơi tồi một chút, số các pixel của chữ bị mất tương đối nhiều làm lệch trọng tâm chữ thì kết quả nhận dạng rất kém.

Khắc phục thiếu sót của thuật toán trên, chúng tôi dựa vào thuật toán đối sánh theo số điểm cắt dọc và cắt ngang.

Giả sử chữ được cô lập có kích thước SIZEH x SIZEV, chúng ta hãy duyệt theo chiều ngang để tìm số điểm cắt ngang.

Gọi Hi là số điểm cắt ngang tại dòng i, vậy tập hợp các điểm cắt ngang sẽ là một dãy ký hiệu H1H2...Hsizev.

Gọi Vi là số điểm cắt dọc tại cột thứ i của ký tự. Tập hợp các điểm cắt dọc sẽ tạo thành một dãy ký hiệu V1V2...Vsizeh.

Thực tế, có rất nhiều phần tử ở đầu và cuối dãy H1H2...Hsizev và V1V2...Vsizeh bằng 0 (có thể do không có điểm đối tượng ở đó, cũng có thể chữ bị mất nét trong quá trình quét bởi Scanner).

Bỏ các phần tử bằng 0 ở đầu và cuối 2 dãy chúng ta sẽ đạt được 2 dãy con là:

H = Hi1Hi2...Hini và

V = Vj1Vj2...Vjnj

Qui tắc nhận dạng sẽ được xác định như sau:

X' được xem là X nếu Hx' là dãy con của Hx hoặc ngược lại và Vx' là dãy con của Vx, hoặc ngược lại.

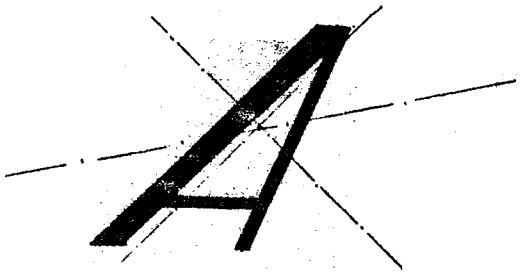
Thuật toán đơn giản, tốc độ cao, với thử nghiệm ban đầu của chúng tôi, thuật toán tỏ ra khá tốt. Kết quả nhận dạng không bị ảnh hưởng bởi việc mất pixel ở biên chữ.

3. Thuật toán nhận dạng bằng đối sánh cấu trúc đồ thị của chữ

Dựa theo trục chính, chữ được phân chia thành k phần bởi các tia lệch nhau một góc α ($k\alpha = 360^0$).

Mỗi phần của chữ sẽ được tính vị trí của trọng tâm. Do vậy k trọng tâm t1,t2,...,tk của các phần tử chữ tạo thành một "hoa thị" đặc trưng cho cấu trúc đồ thị của chữ đó.

Việc so sánh các hoa thị đã được chuẩn hóa là một bài toán lý thú của lý thuyết đồ thị. Thuật toán này to ra thật sự có hiệu quả, với độ chính xác cao và đặc biệt là độ chính xác của kết quả nhận dạng không phụ thuộc vào các kích cỡ khác nhau đối với một font chữ bất kỳ.



III. Thuật toán nhận dạng chữ viết tay có ràng buộc.

1. Một số ràng buộc

- Các chữ nhận dạng được là các chữ in latinh và các số từ 0 đến 9 viết theo một trong năm kiểu chữ thông thường nhất là Dual, Gothic, Script, Orator, OCRA và OCRB.

- Số 0 và chữ O được phân biệt ở chỗ số 0 có thêm một dấu gạch ở giữa.
- Trên mỗi trang, các dòng và các chữ không được dính vào nhau.
- Các nét chữ không lổ chổ, không đứt nét, không ngoằn ngoèo quá và không nghiêng quá 15°

2. Trích chọn các dấu hiệu đặc tả và nhận dạng chữ.

Mỗi chữ viết ra được nhận biết và phân biệt qua cấu trúc của nét tạo nên chúng như: Số các nét, vị trí, hình dáng của các nét. Chúng được thể hiện bằng số các lát cắt ngang, các nét cong hay thẳng, mở hay đóng và các điểm cực trị.

Tìm một khoảng ổn định nhất của chữ và xét số các lát cắt cũng như tính chất của các lát cắt trên khoảng đó. Ta chia các chữ thành hai nhóm

- Nhóm một là nhóm các chữ có ít nhất một lát cắt một, thông thường là các chữ

C E F G I J L P S T Y Z
1 2 3 4 5 6 7 9

- Nhóm hai là nhóm còn lại của các chữ

A B D H K M N O Q R U V X 0 8

Dùng thêm tính đóng mở ta chia nhóm 2 thành 4 nhóm con:

- Đóng trên, đóng dưới D O Q 0 8
- Đóng trên, mở dưới A M R
- Mở trên, đóng dưới M U V
- Mở trên, mở duới H K N X

Từ đó xét thêm tính cong, thẳng của nét bên trái hoặc phải ta sẽ phân biệt được các chữ trong mỗi nhóm nhỏ với nhau.

Đối với nhóm 1 dùng các dấu hiệu đặc trưng, đều không phân biệt chính xác được nên ta dùng kiểu mắt na di động.

Tìm hai lát cắt 1 khác nhau làm điểm tựa, ta chia chữ thành 6 phần như sau:

I	II
III	IV
V	VI

Với mỗi chữ ta được một vectơ F = (F(1),...,F96)) mà

$$F(i) = \begin{cases} 1 \text{ Nếu có điểm đen trên phần i} \\ 0 \text{ nếu không} \end{cases}$$

Ví dụ

F(C) = (0,1,1,0,0,1)
F(I) = (1,1,0,0,1,1)
F(J) = (1,1,0,0,1,0)

Với véctơ F đó, ta có thể phân biệt từng chữ trong nhóm 1 cung có khi phải làm thêm như đối với C và G thì F(C) = F(G). Trong nhưng trường hợp nhập nhằng như vậy, Một số dấu hiệu đặc trưng phụ trợ sẽ được trích trọn. Chúng tôi không trình bày chi tiết ở đây.

3. Chương trình nhận dạng: Hệ Việt - IN

Hệ VIET-IN là chương trình nhận dạng chữ in viết tay theo các thuật toán trên, viết bằng ngôn ngữ TURBO PASCAL, được dùng trên các PC-AT/XT và tương thích.

Có thể dùng một trong 3 kiểu chữ là: dùng bàn phím, con chuột (Mause) hoặc lấy từ file. Các trang viết cần nhận dạng được đưa qua Scanner tạo thành các file dữ liệu và được hệ nhận dạng. Kết quả nhận dạng được in ngay trên màn hình, hoặc lưu trữ dưới dạng file nếu mong muốn.

Nhận ngày 10-9-1988

TÀI LIỆU THAM KHẢO CHÍNH

1. Rosenfeld A. Kak A. C. Digital Picture Processing. Newyork 191
2. Gonzalet, Tu: Principles of Pattern Recognition. Newyork 1981.
4. Eighth Internationnal Conference on Pattern Recognition - 1986 Proceedings (Section 2.3.1 OCR)
5. Kobalebckuu B.....

ABSTRACT

Some Effective Methods and Algorithms
For the Character Recognition.

In this paper, some metods preprocessing the characters were presented in order to improve the quality of recognition processes. At the same time, three our effective character recognition algorithms were described in detail, too. First experiments were perpormed with a help of PC computer and Microtek scanner on four widely-used fonts of characters. The first results with a high qualyti of character recognition have been received, that confirms the correctness and effctness of our applied algorithms.