電子情報通信学会技術研究報告 PRMU2001-272～300 2002年3月15日　信学技報Vol.101 No.713
ISSN 0913-5685 Page 9-16

オンライン手書き日本語文字認識のための大分類手法の改良

松本　馨　　中川　正樹

東京農工大学大学院工学研究科
〒184-8588　東京都小金井市中町 2-24-16

E-mail: matsu@hands.ei.tuat.ac.jp, nakagawa@cc.tuat.ac.jp

あらまし 本稿では，オンライン手書き日本語文字認識のための大分類手法の改良について述べる．我々は，各種パターンに対して安定した結果が得られるオフライン文字パターン認識手法を応用して大分類を実現してきた．ここでは，8×8の区画ごとに8方向の方向特徴を抽出し，主成分分析を用いた特徴圧縮，使用する特徴次元数の可変化，ソート手法の改良，辞書学習など，各種の改良を試みた．これらの方法と，その効果について報告する．

キーワード 文字認識，オンライン認識，大分類，日本語文字，手書き文字パターン

Improvement of a coarse classification method for on-line recognition of handwritten Japanese characters

Kaoru MATSUMOTO and Masaki NAKAGAWA

Graduate School of Technology Tokyo University of Agriculture and Technology
2-24-16 Naka-cho Koganei-shi, Tokyo, 184-8588, Japan

E-mail: matsu@hands.ei.tuat.ac.jp, nakagawa@cc.tuat.ac.jp

Abstract This paper discribes improvement of a coarse classification method for on-line recognition of handwritten Japanese characters. We have been appling off-line character pattern recognition techniques for the coarse classification. We tried a few attempts to improve its performance. We extract 8-directional features from each of 8 * 8 regions, compress them by principal components analysis and employ different numbers of features for different classes of input character patterns. We elaborate the sorting method of candidates and the prototype learning for preparing the character pattern dictionary. The paper report these attempts with their effects.

Key words Character recognition, On-line recognition, Coarse classification, Japanese character, Handwritten character pattern

1．はじめに

文字認識システムは近年大きな進化を遂げてきたが，未だに多くの課題を抱えている．本来，このようなシステムを必要とする携帯端末は大きさや電力の問題から，現在一般に使用されている計算機より２～３世代前の水準である．つまり，標準的な計算機の水準よりも劣る環境にあわせた高速な認識システムが必要である．また，枠なし文字認識において多数の文字を切り出し,文字認識,文脈後処理のつじつまを取って認識するためにも高速な認識システムが必要となる．これを解決するためには専用のハードウェアを用いて高速化するアプローチも考えられるが，一方で，汎用のプロセッサでも高速化を行えるように大分類といわれる事前に候補を絞り認識エンジンの負担を軽減する手法が有効である．

一般に詳細識別は大分類に比べてその処理に時間がかかる．そのため，一部の処理を大分類に任せることで認識全体の速度を向上させることが可能になる．我々は，各種パターンに対して安定した結果を得られるオフライン文字パターン認識手法を応用して大分類を実現してきた．しかし，この大分類で用いられる手法は，元々オフライン文字認識で利用されているものであり大分類としては時間がかかる．このため，より高速な大分類手法が求められている．

以下，2.では，オンライン手書き文字認識システムの設計について述べ，3.では大分類について述べる．4.では実験を行い，5.でまとめる．

2．オンライン手書き文字認識システム

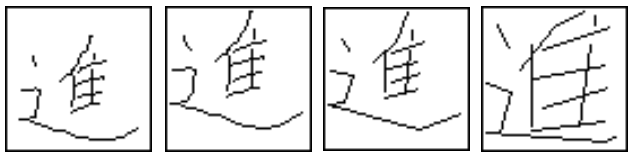
現在のオンライン手書き文字認識システムについて述べる．本システムは，次の処理で構成されている．

(1) 前処理　（正規化・特徴点抽出）
(2) 大分類　（認識候補の大分類）
(3) 詳細分類（線形時間伸縮マッチング）
(4) 文脈処理（認識結果からの文脈判断）

前処理で入力パターンの大きさと辞書パターンの大きさを合わせ，特徴的な筆点だけを抽出する[1]．これにより，入力パターンと辞書パターンとの間で特徴点の位置の比較ができるようになり，マッチングや類似度算出に位置特徴を用いることができる．また，特徴点以外の点を除去することで認識に要する処理が減り，筆跡のぶれなどを除去することができる．正規化には線形正規化と非線形正規化を用いる．大分類では線形正規化の後，特徴点抽出，非線型正規化を行ったパターンを用い，詳細分類では線形正規化の後，特徴点抽出を行ったパターンを用いる．

大分類で非線形正規化を用いるのは，非線形正規化されたパターンとの相性が良く，高い精度が得られるためである．これは使用する手法がオフライン文字認識手法を応用したものであることが関係していると思われる．

前処理の例を図1に示す．



元パタン　線形正規化　特徴点抽出　非線形正規化

図1．前処理の例

次に，大分類で候補をおおまかに絞り，入力パターンに該当すると思われるカテゴリだけを以降の処理に通すことで詳細分類処理の手間を軽減させる．詳細分類処理では類似度の算出のために辞書パターンと入力パターンの特徴点を対応づける．そして，線形時間伸縮マッチング[2],[3]を行いパターン間の類似度を算出し，入力パターンと最も類似度の高い辞書パターンが該当パターンとなる．

その後，必要に応じて文脈処理を行う．文字は単体で見ると判別の難しいものがあるため，文字列の認識結果のつながりから認識候補の順位の入れ替えを行う．

3．大分類処理

オフライン文字パターン認識手法を応用し，大分類を行う．オフライン文字パターン認識手法を使う利点の一つとして，入力パターンの書き順に左右されないということが挙げられる．短所としては時間がかかること，辞書データサイズの増大が挙げられる．

この手法は，オフライン文字認識手法である方向線素特徴ベクトルをとる方法[4],[5]を簡略化したものである．

3.1 方向特徴ベクトル抽出

本手法では，分割数を4×4とした場合と，8×8とした場合，また，それぞれに4方向とするか8方向とすることで４種類の方法が考えられる．ここでは最も簡単な4×4分割4方向について説明する．

まず，図2-(a)に示すように入力パターンを4×

4のメッシュに分割する．文字パターンのサイズが64×64ドットだとすれば，この分割領域一マスのサイズは16×16ドットとなる．そして，このマス2×2 = 4個からなる32×32ドットの小領域を考える（図2-(b)参照）．この32×32ドットのマスクを文字パターンに掛け，このマスクを16ドットずつ移動させることにより，3×3 = 9個の小領域を作る．

次に，各小領域において入力パターンのストロークに4方向（｜，―，／，＼）の成分がどれくらい含まれているかを数える（8方向の場合は，向きの情報を含む）．計算方法は入力パターンのストロークを構成する特徴点間を結んだ線分について，その単位方向ベクトルを求め，それと8方向それぞれの単位ベクトルとの内積の絶対値をとる（単位方向ベクトル同士の内積は，その二つのベクトルのなす角度の余弦（cos）を表す）．内積は方向が近いほど大きい値を示す．そこでこの４つの内積をその線分の方向特徴とする．そして，入力パターンの全ストロークに対してこの計算を行い，各小領域ごとにその和をとる．ただし，その和をとるときに，小領域の中心部からの距離に応じて，図2-(c)のような中心に近いほど大きい重みをつける．

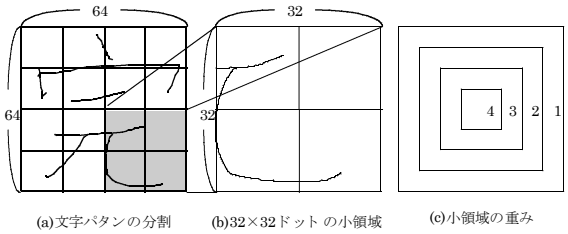


図2．方向特徴ベクトル抽出

こうして，9個の各小領域について４つの方向特徴量が算出され，9領域×4方向＝36次元の方向特徴ベクトルが計算される．各手法による方向特徴の次元数を表1に示す．

表1．次元数

手法	次元数
4×4分割4方向	36 次元
4×4分割8方向	72 次元
8×8分割4方向	196 次元
8×8分割8方向	392 次元

3.2 主成分分析を用いた特徴圧縮

オフライン文字認識手法を用いた大分類は，筆順に左右されないため安定した認識率が得られるという利点がある．しかし，多次元での距離計算を行うため，その処理に時間が多くかかるのが問題となる．この手法をそのまま使うと，特徴量が最小となる4×4分割4方向では36次元，最大となる8×8分割8方向では392次元にもなる．次元数を減らせば処理時間は短縮できるが，その分だけ精度が落ちてしまう．しかも，選択できる次元数は36，72，196，392次元の４段階しかない．

このため，統計手法である主成分分析を用いた大分類処理を行う．この手法を用いると使用する次元数を細かく設定できるため，先に述べた問題が起きなく,柔軟に対応できるという利点がある．しかし，特徴変換にかかる時間的ロスや，変換による精度の劣化，変換に用いる表の分だけ辞書サイズが増大してしまうという欠点もある．

主成分分析は，オフライン手法による大分類の方向特徴ベクトルについて適用される．ここで用いられる方向特徴ベクトルは，最小で36次元（4×4分割4方向使用時），最大で392次元（8×8分割8方向使用時）となる．本手法においては，より大きな次元をもとに圧縮を行う方が候補含有率の点で有利であることが分かっているため[6]，最大次元である392次元（8×8分割8方向）を圧縮前の特徴として用いている．

3.3 特徴変換

方向特徴抽出処理により抽出された方向特徴ベクトルを,主成分による特徴ベクトルに変換する．

まず，入力された方向特徴ベクトルに対して，値の標準化処理（平均0，分散1を持つように標準化）を行う．次に，主成分分析により求められた相関行列を用いて主成分特徴ベクトルに変換する．ここでは，主成分への変換を行うための表を用いて変換を行う．処理内容は，(392×圧縮後の次元数）の行列による積和演算である．

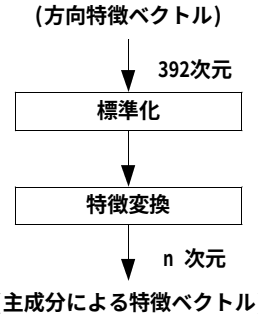


図3．主成分による特徴変換

3.4 文字画数による次元数切り替え

大分類システムにおいて，高画数文字は低画数文字に比べて一意的な特徴が多く含まれ，認識が容易な傾向にある．このため，高画数文字に対して高精度な認識処理を行う必要が少ない.そこで，大分類システムで認識の容易な高画数文字は低次元数で高速処理し，認識の困難な低画数文字は高次元数による高精度処理を行うことで，候補含有率の低下を抑えたまま高速化することを考えた．以後，画数は実際の文字パターンの画数を意味する．

ここで，必要とする次元数を把握するために，予備実験を行った．まず，1次元の特徴で大分類処理を行い，100位以内に正解が含まれているなら1次元の特徴量で十分とする．含まれていない場合，次元数を増やし，2次元で大分類を行い，100位以内に正解が含まれるか見る，といった処理を繰り返し，最大100次元までを調べた．実験では TUAT Nakagawa Lab. HANDS-nakayosi_t-98-09（以下，nakayosi_tと表記）のデータ80人分にて大分類辞書を作成し，残りの80人分のデータで分布を求めた．これにより，文字画数と大分類に必要とする次元数，及び，その文字数の分布が分かる．結果を図4に示す．

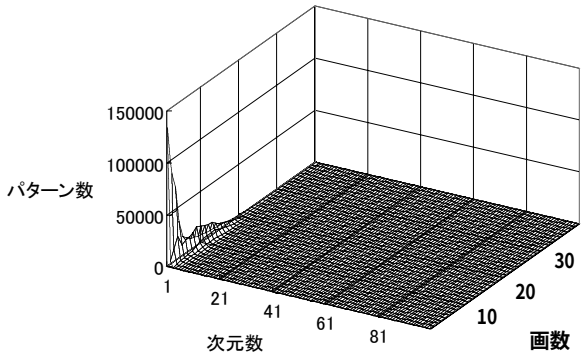


図4．画数・次元数・文字数による分布

この分布から，低画数文字は比較的大きな次元数での処理を必要とするのに対して，高画数文字は小さな次元数で問題の起きないパターンが多いことが分かる．この分布を利用して，画数ごとに割り当てる次元数を設定した．次元の設定は，下記のようにして行った．

図4の次元数をx軸，画数をy軸，パターン数の数値をz(x，y)とする．xの範囲は対象とする次元数が100次元なので1～100，yの範囲は実際に出現した画数の最大値が36であったため，1～36である．ここで，z(1，1)～z(100，36)の値を全て１減ずる．しかし，既にz(x，y)の値が0である場合，そのままとする．これを何回か繰り返す．これは，立体的な山の模型に水を流して，低い部分が水没していく様をイメージすると分かりやすい．その後，z(1，1)～z(100，36)の範囲内で z(x，y)の値が0より大きなもので，かつ，xが最大値である場所を，「画数」と「使用する次元数」の組み合わせとして登録する．z(x，y)の例を図5に示す．

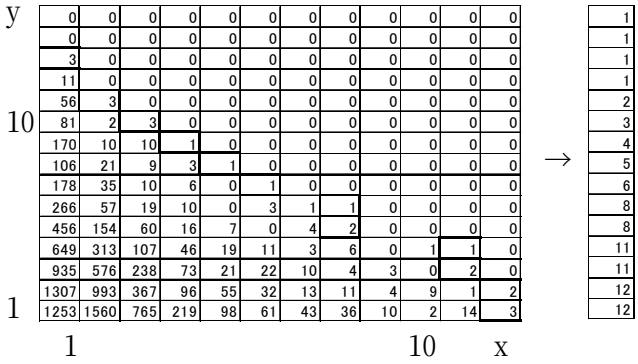


図5．画数から使用次元数の算出

画数と使用する次元数の対応の例を図6に示す．これに従うと，低画数の文字ほど使用する次元数が多く，高画数の文字は使用する次元数が少ないのが分かる．より高速に動作させるために使用する次元数を減らしても，低画数文字については最低限の必要な次元数は保持される．

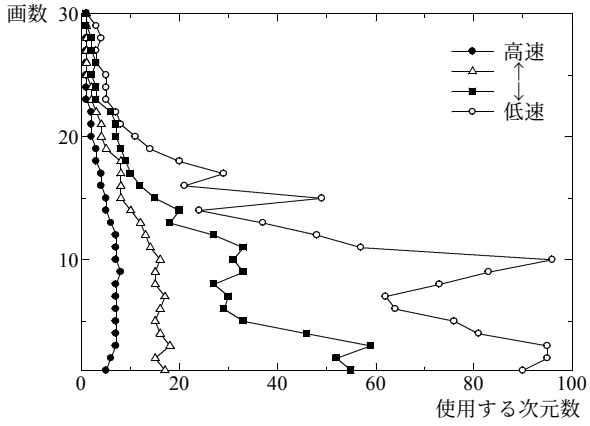


図6．画数と使用する次元数

大分類処理では，入力パターンから画数を算出し，画数と次元数の対応表から使用する次元数を求め，その次元数分の特徴を使用して大分類処理を行う．このため，入力が低画数文字の場合，処理時間は遅いが高精度処理になり，高画数文字では低精度で速い処理になる．

3.5 距離計算

前述の特徴変換により変換された主成分特徴ベクトルで，辞書との距離計算を行う．

主成分特徴ベクトルが方向特徴ベクトルと異なる点は，主成分特徴ベクトルは対応する固有値の値が大きな成分ほど多くの変動をもたらすことにある．よって，特徴量の圧縮をするために，変動に大きく貢献しない固有値の小さい下位の主成分を距離計算対象から外し，上位の主成分のみ距離計算を行うようにする．これにより，特徴ベクトルの次元数を減らした状態で距離計算を行うことができ，処理を高速化することができる．また，この距離計算に用いる次元数は任意に変更することが可能であり，資源に余裕のある時は次元数を増やして精度を高めたり，速度を必要とする時は次元数を減らして速度を高めることが可能である．

距離の定義には，Euclid距離，CityBlock距離などがあるが，本手法ではCityBlock距離を採用した．これは，大分類の役割上，計算に高速さが求められること，距離はあくまで相対的評価のためにあり，値自体に絶対的な正確さは要求されていないことを考慮したためである．

3.6 距離順ソート

算出された距離値をもとに，距離順ソートを行い，候補文字を出力する．ソート方法はさまざまなアルゴリズムが存在するが，大分類の距離順ソートは下記のような条件下で行われる．

・値は整数値しか存在しない．
・値の範囲が限定的である．
・使用可能メモリ空間が大きい．
・必要な結果は先頭の100～200番までである．

この条件は，一般に用いられるアルゴリズムよりも限定的で制約が緩い．よって，これを利用してより高速なソート方法を用いた．

仮に距離値の範囲を1～5とする．まず，距離値の範囲分の配列（整数型）を用意する(図7-(a))．次に，距離値ごとに同じ距離値の数量を数えて配列におさめる(図7-(b))．そして，配列の値を上から順に累積値に置き換える(図7-(c))．元の値が 2，1，3，2，2 であった場合,累積値は 2，3，6，8，10 となる．ここで，この累積値が結果の格納先を示すポインタとなる(図7-(d))．図7の例では，距離値1の「あ」はポインタにより，2番目に格納され，距離値５の「い」はポインタにより１０番目に格納される．結果を格納するごとに，使用したポインタの値を１減らす．ここでは，「あ」を格納した段階でポインタの値が２から１になり,「い」を格納した段階で３から２になる．距離値が「あ」と同じである「お」は,「あ」を格納した段階でポインタが１減らされているため，同じ場所ではなく，１だけ小さい場所に格納される．このため，同じ距離値の結果が同じ場所に格納されることはない．この手続きにより，ソートが実行される．

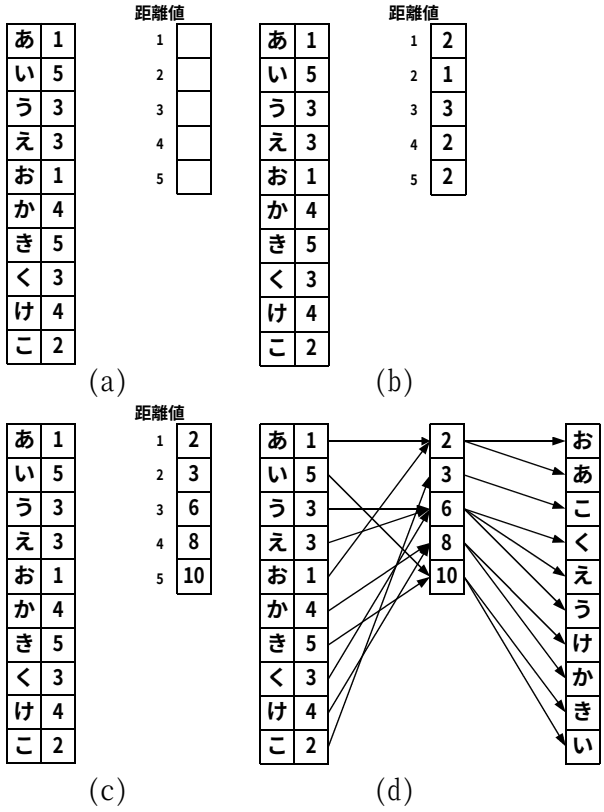


図7．距離順ソート

3.7 文字パターンの辞書学習

大分類辞書は nakayosi_t のデータ163人分を用いて作成している．辞書には１カテゴリごとに１テンプレートの圧縮した方向特徴ベクトルデータを格納している．この値は163人分のデータで平均化された値である．

ここで，疑問が生じる．登録しているパターンが本当に候補含有率向上に貢献しているのかどうか分からないという点である．一部の乱れたパターンが候補含有率を悪化させている可能性もある．

このため，候補含有率を向上させるデータが優先して登録されるように図8のような手順で辞書作成を行った．

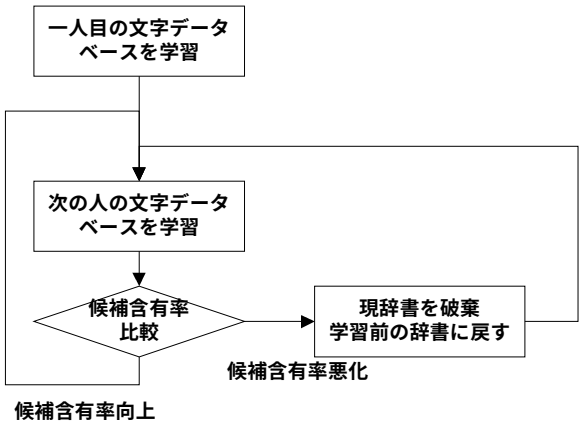


図8．大分類辞書学習

候補含有率の比較は TUAT Nakagawa Lab. HAND S-kuchibue_d-97-06 (以下, kuchibue_dと表記) のデータ1～10番を候補含有率の評価に用い，候補含有率が向上したデータのみ学習対象となるようにした．

学習による候補含有率の変化を図9に示す．この値は kuchibue_d のデータ1～10番より得られた200位候補含有率の平均値である．

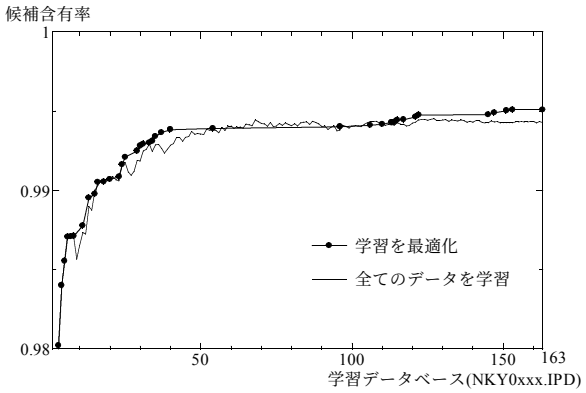


図9．学習による候補含有率の変化

学習が進むにつれて候補含有率が上がり，学習に使うデータを絞ることで，何もしない場合に比べて候補含有率が高めになる傾向にあることが分かる．

3.8 文字画数による候補削減

オンライン手書き文字データベースの収集・解析によると，文字の54.12%が正しい画数で書かれ，43.71%が画数減少，2.17%が画数増加の傾向にある[7]．この結果から，実際に筆記される文字の大半は正しい画数か画数減少の傾向であり，増加分については多くを考える必要がないといえる．この傾向を利用して，文字画数による候補削減を行う．

文字画数による候補削減に対応するために，辞書に正しく書かれた場合の画数情報を追加し，さらに画数順に降順ソートを行った．

まず，入力パターンのペンアップ・ダウン回数から画数を算出し，辞書画数の大きい文字から順に距離計算を行う．徐々に画数の小さい文字に処理が移り,辞書画数が入力パターンの画数－α(任意の設定値)になった段階で距離計算を終了する．これにより，入力パターンより画数がα - 1画少ない候補を処理の対象から外すことができる．

4．実験

下記環境で実験を行った．

CPU : Intel PentiumPRO 200MHz (256KB Cache)
RAM : 128MB
OS : Microsoft Windows NT 4.0 Workstation

特に明記しない場合100次元の特徴ベクトルを持つ辞書(精度重視)を利用している．

(a) 文字画数による次元数切り替え

入力文字画数により，大分類で用いる特徴の次元数を切り替えた場合の結果を図10に示す．本手法では，大分類に使用する次元数は最大で100次元，最小で1次元に切り替わる．

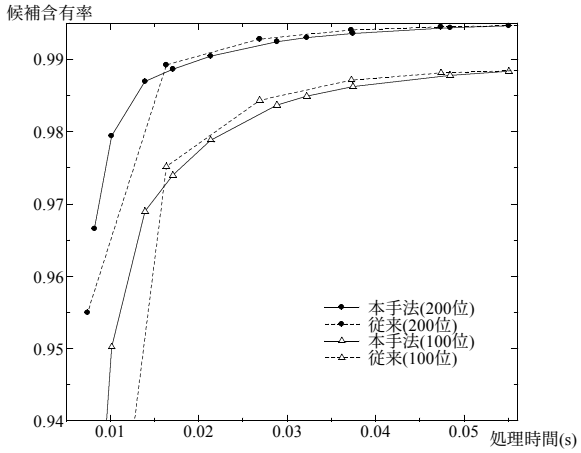


図10．文字画数による次元数切り替えと従来手法

大分類で用いる次元数を少なくしたとき(処理時間が短い場合)に効果が現れている．逆に，次元数が多めのとき(処理時間が長い場合)には従来

手法の方が優れていることが分かる．これは，単純に使用する次元数を減らす従来手法に比べ，処理の破綻が起きにくいためと考えられる．

(b) 距離順ソート方法比較

距離順ソート方法を，一般的なソートアルゴリズムであるクイックソートと，本手法(改良型)とで比較した．結果を表2に示す．処理速度は一文字あたりの処理時間である．

表2．ソート方法比較

ソート方法	処理速度(ms)
クイックソート	27.23
本手法	10.75

処理時間は 1/3 程度に高速化された．

(c) 文字パターンの辞書学習

学習データを絞って最適化した場合と，全ての文字を学習させた従来辞書での候補含有率をkuchibue_dのデータ120人分で評価した．結果を表3に示す．この値は，40次元辞書を利用した100位候補含有率である．

表3．辞書比較

使用辞書	候補含有率(%)
最適化辞書	98.76
従来辞書	98.62

辞書の最適化により，120人中83人が認識率の向上，2人が変化なし，35人が認識率の低下となった．

しかし，単純に全てのパターンを学習している場合でも，学習データ量が増えてくれば徐々に候補含有率が向上しているのが図9から分かる．データを絞る場合，学習が途中で進まない状態が続いたため，部分的に候補含有率が逆転してしまっている．この学習が進まなかったという問題は，学習させる・学習させない，の判断をデータベース１人ごとに行っていたのが原因である可能性が高い．これをデータベース１人単位でなく，各１文字単位で学習させるか・しないか，の判断をさせることで，より最適化が進むと考えられる．

(d) 文字画数による候補削減

入力文字画数により，候補数を削減した大分類の結果を表4に示す．ここでの画数処理なしとは文字画数によるソートを行った辞書を用いるが，画数による候補削減処理を入れなかった場合を示す．

表4．文字画数による辞書ソート後

	処理時間(ms)	200位含有率(%)
画数処理なし	59.35	99.08
- 1画 まで	51.87	99.14
- 2画 まで	52.41	99.12
- 3画 まで	53.07	99.10

画数処理なしの場合に比べて処理速度が向上している．また，候補含有率も向上している．しかし，全体として，候補含有率が何もしない場合(辞書最適化は行ったが，文字画数による辞書ソートを行わなかった辞書での候補含有率 ： 99.48%)よりも劣っている．これは，本来なら同じ値になるはずである．しかし，文字画数による候補削減を用いた方が候補含有率が低いという結果が出た．これらの方式の違いは，文字候補登録の順番が異なるという程度でしかない．

文字画数で降順ソートを行った場合，距離計算は，画数が多く出現頻度の低い，高難易度の漢字から始まる．一方，辞書作成を普通に行った辞書は文字コード順に並んでおり，先頭には比較的出現頻度の高い文字(数字，記号，ひらがな，カタカナ)から始まる．このため，同点のカテゴリは出現頻度の低い文字から先に登録される傾向になる．ソートアルゴリズムによる違いはあるが，本手法では，先に登録した文字が高順位に位置してしまう．ここで，距離順に固定数の結果出力を行うと，大分類による評価値が同点であったカテゴリでも，固定数の出力をするために同点のカテゴリを一部破棄することになる．そこで候補含有率に差が生じたものと思われる．この問題を解決するには，認識結果に信頼度を持たせ，候補数を可変とすることが必要だと思われる．

5．まとめ

入力画数に応じて次元数を切り替える方法は，低次元での高速処理時に良い結果が出た．

ソート方法の比較では，改良型ソートの方がクイックソートを利用した場合に比べ，ソート対象になる値の範囲を限定する必要があるが，より高速であることを確認した．

辞書学習においては，辞書に登録するパターン

を絞った場合の方が良い結果が出た．

文字画数による候補削減では，画数処理による高速化は認められたが，全体として候補含有率が何もしない場合に比べて低くなった．候補数を固定する弊害の一つが明らかになった．

謝辞

ソートアルゴリズムに関し，貴重なアドバイスをいただいた，株式会社富士通研究所の田中宏氏に感謝いたします．

参考文献

[1] K. Ishigaki, T. Morishita, “A Top-Down On-line Handwritten Character RecognitionMethod via the Denotation of Variation, ” Proc. 1988 Int'l Conf. on Computer Processing of Chinese and Oriental Languages, pp.141-145, (1988).

[2] 秋山 勝彦，中川 正樹, “ ストロークのつながりに寛容なオンライン手書き文字認識, ” MIRU94，pp.67-74，(1994).

[3] 秋山 勝彦，中川 正樹, “ オンライン手書き日本語文字認識のための線形処理時間伸縮マッチングアルゴリズム, ” 信学論(D-II), Vol.J81-D-II No.4，pp.651-659，(1998).

[4] 若林 哲史，鶴岡 信治，木村 文隆，三宅 康二, “ 手書き文字認識における特徴量の次元数と変数変換に関する考察, ” 信学論(D-II), Vol.J76-D-II No.12，pp.2495-2503，(1993).

[5] 孫 寧，安部 正人，根本 義章, “ 改良型方向線素特徴量および部分空間法を用いた高精度な手書き文字認識システム, ” 信学技報 PRU94-31，pp.33-40，(1994).

[6] 松本 馨，中川 正樹, “ オンライン手書き文字認識のための高速な大分類手法の研究開発 ”, 情処第56回全大, 分冊2 pp.111-112, (1998).

[7] K. Matsumoto, T. Fukushima, M. Nakagawa, “Collection and analysis of on-line handwritten Japanese character patterns, ” Proc. ICDAR, (2001).