PAPER *Special Section on Document Image Understanding and Digital Documents*

Structuring search space for accelerating large set character recognition

Yiping Yang *student member*, Bilian Zhu *student member* and Masaki Nakagawa, *member*

Summary
This paper proposes a "structuring search space" (SSS) method aimed to accelerate recognition of large character sets. We divide the feature space of character categories into smaller clusters and derive the centroid of each cluster as a pivot. Given an input pattern, it is compared with all the pivots and only a limited number of clusters whose pivots have higher similarity (or smaller distance) to the input pattern are searched in, thus accelerating the recognition speed. This is based on the assumption that the search space is a distance space. We also consider two ways of candidate selection and finally combine them. The method has been applied to a practical off-line Japanese character recognizer with the result that the coarse classification time is reduced to 56% and the whole recognition time is reduced to 52% while keeping its recognition rate as the original.

Key words:
Character recognition, Large character set, Search space, Pivot, Candidate selection.

1. Introduction

Chinese, Japanese or Korean have a large character set with several thousands different categories, so that recognition takes more time than Latin alphabet or number recognition. Although the hardware development for personal computers has been remarkable, the faster the character recognition, the more capacity can be provided to run multiple recognizers, combine them with context post-processing, incorporate them in handwritten text search and so on. Moreover, there is high demand for character recognition on small devices like mobile phones or PDAs. Therefore, to accelerate large character set recognition has been studied to have practical importance.

The two-stage architecture of coarse classification (pre-classification or candidate selection) and fine classification has been employed in many practical systems [1, 2]. Coarse classification should be significantly faster than fine classification. Selecting a limited number of candidates robustly, prototype patterns to match with the input pattern in the fine classification are reduced from several thousands to several hundreds or even less. Consequently, the whole recognition process is accelerated.

Many approaches use simpler distance measures than those for fine classification [3,4]. The confidence evaluation provides even more precise candidate selection [5]. Others employ simple features different from those for fine classification [6]. Sequential (multi-stage)

Classifications employing a partial set of features at each stage are also applied as in [7]. This type of candidate selection is made during recognition so we call it a dynamic process.

The other possible method is to structure the search space so that the search for candidates can be made only to some portion in the search space as in [8]. The feature space of character categories is divided into smaller clusters and the centroid of each cluster is derived as a pivot. Given an input pattern, it is compared with all the pivots and only a limited number of clusters whose pivots have higher similarity (or smaller distance) to the input pattern are searched in, with the result that we can accelerate the recognition speed. Since this is made before the recognition process, it is a static process and we distinguish it from the dynamic process mentioned above. We call this a "structuring search space (SSS)" method and we have applied it to our off-line recognizer of handwritten Japanese characters with notable effect [8].

There have been some methods to structure the search space and make the search faster. The simplest is the ordered space. Another is the tree structure. They can be applied when prototypes are ordered in some sense or classified into a tree structure. On the other hand, the SSS method does not assume such structures and we make a structure by clustering.

For printed character recogniton, Tseng et al. proposed a static approach by clustering prototypes employing simple and small number of features [9]. Fujimoto et al. showed another approach where coarse classification is made in a sub-space much smaller than that for fine classification [10]. Taking multiple small sub-spaces makes it even faster. The above two approaches, however, adds the additional problem of finding a feature space for coarse classification. On the other hand, our method works in the original feature space for fine classification. We only have to assume distance space for this single space. We tried a similar approach with a smaller set of features for handwritten Japanese character recognition but we could not speed up recognition without sacificing recognition rate significantly.

This paper is a formal and updated version of [8]. Section 2 of this paper describes the off-line recognizer of handwritten Japanese characters for which the proposed method has been evaluated. Section 3 presents the SSS method and its evaluation. Section 4 proposes a hybrid

Manuscript received November 2004.
Manuscript revised .
The authors are with Graduate School of Engineering, Tokyo University of Agriculture and Technology, 2-24-16 Naka-cho, Koganei-shi, Tokyo, 184-8588 Japan.

candidate selection, incorporates it into the SSS method and evaluates its effect. Section 5 concludes the paper.

2. Off-line Character Recognizer

The off-line recognizer of handwritten Japanese characters used for this research represents each character as a 256-dimensional feature vector. It scales every input pattern to a 64x64 grid by non-linear normalization [11]. Then, it decomposes the normalized image into 4 contour sub-patterns representing directional features of the 4 main orientations. Finally, it extracts a 64-dimensional feature vector for each contour pattern from the convolution with a blurring mask (Gaussian filter).

The coarse classification step precedes the actual fine classification. The original form of the coarse classification selects 40 candidates with the shortest Euclidian distances between the categories' mean vectors and an input pattern. The fine classification employs a modified quadratic discriminant function (MQDF2) [12]. We may call this step as a fine recognizer.

It was trained with a training set composed of 400 persons' handwritten character patterns (1,285,600 patterns of 3,214 categories) in the HP-JEITA database. HP-JEITA database includes digits, Latin alphabet characters, symbols, hiragana (a set of phonetic characters), katakana (another set of phonetic characters) and Japanese Kanji characters of Chinese origin.

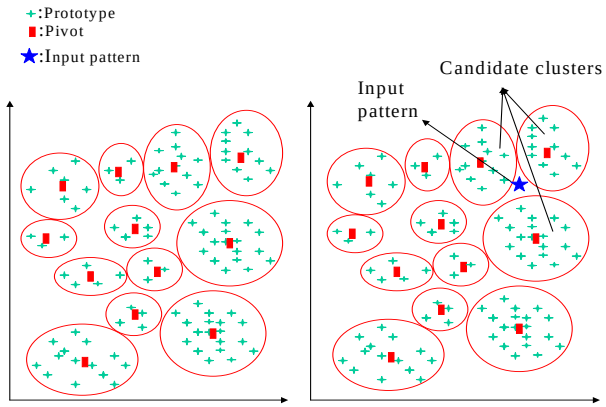

Fig. 1 Conceptual figure of structuring the search space

3. Structuring the Search Space

In this section we present the basic idea and details of the design of the SSS. Then, we show the evaluation through recognition experiments.

3.1 Basic model

For large category set recognition, we consider to divide the feature space of these categories into smaller clusters taking the centroid of each cluster as a pivot. Fig. 1 is a conceptual figure of the feature space drawn in two-dimensional space (although typical feature space for large character set recognition takes 256 or 512 dimensions). It should be noted that each cluster is made up of different categories rather than multiple prototypes of a single category.

Given an input pattern, it is compared with all the pivots and only a limited number of clusters whose pivots have higher similarity (or smaller distance) to the input pattern are searched in, with the result that we can accelerate the recognition speed. This is based on the assumption that the search space is a distance space.

After the search space is structured, it is adopted before the coarse classification as shown in Fig. 2.

The problem here is to find the appropriate number of clusters (i.e., the number of pivots). The more the number of pivots is, the more clusters must be searched in although each cluster becomes smaller.

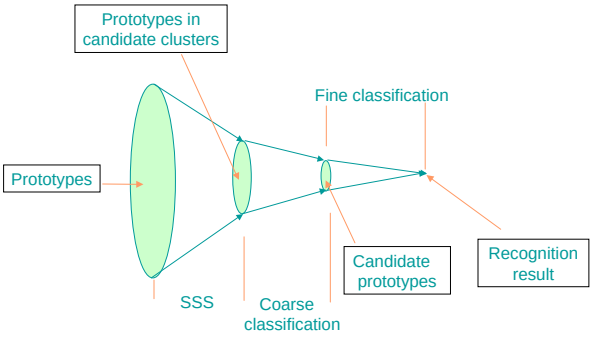

Fig. 2 Structure of a recognizer with SSS

3.2 Detailed design

The ultimate goal is to achieve the highest recognition rate with the smallest recognition time. Generally speaking, however, there is a trade-off between recognition rate and its speed. Moreover, the optimal point depends on the application and environment. Therefore, we need to clarify the characteristic of our method, i.e., the relationship between the recognition rate and recognition time and find the optimal parameters to gain the best balance between speed and precision.

3.2.1 Candidate selection

Given an input pattern, and having compared it with all the pivots, the next question is how to select which candidate clusters we search in. There are two ways to select them:

(1) We set a constant for the number of candidate clusters, and select *l* clusters whose centroids have the shortest Euclidian distance to the *l-th* shortest Euclidian distance to the input pattern (Algorithm-I).
(2) We find the nearest centroid with the distance d_{min}, and then we set a multiplying coefficient *m* ($m>1$) so that all the clusters within the distance $m*d_{min}$ become candidate clusters (Algorithm-II).

3.2.2 An optimal number of clusters

The problem posed in 3.1 remains to be solved. That is to find the appropriate number of clusters (i.e., the number of pivots). Actually, it is difficult to find out an optimal number of clusters through inference. Our base recognizer has 3,214 prototypes and they can be divided into from 2 to 3,213 clusters. The optimal number of clusters may depend on the method for selecting candidate clusters (section 3.2.1) and on the method for selecting original centroids (section 3.2.3). We have to make experiments to determine the optimal number. They follow in section 3.3.

3.2.3 Details of clustering

We employ the LGB algorithm [13] for clustering since it is one of the simplest and most effective methods. In order to start the algorithm, however, we have to define initial clusters or centroids, which somehow determine the result. We have tried the following three methods for selecting initial clusters where "n" is assumed as the number of clusters:
(1) Measure the distance from the origin point to every prototype in the feature space and partition the distance (1-dimensional space) from the shortest distance to the longest distance into *n* sections of the equal length. Then, place all the prototypes into those sections according to the distances. The resulting sections are the initial clusters.
(2) Choose only one axis (dimension) among 256 dimensions of the feature space and partition the axis (1-dimensional space) from the smallest value to the largest value into *n* sections of the equal length. Then, place all the prototypes into those sections according to their values of the chosen axis. The resulting sections are the initial clusters.
(3) Divide all the prototypes into *n* initial clusters of equal size according to the order of the character code.
Experimental results have revealed that the influence of different initial centroids is very subtle. Therefore, we employed the simplest method (3).

3.2.4 Management of clusters

After all the prototypes are divided into small clusters, we manage them to speed up the search. We store the following information in a file: the total number of clusters, the number of prototypes and the centroid of each cluster, indexes of all the prototypes in each cluster and the dimension of feature vectors. Thus, all the pre-calculated information is loaded from the file when the search process is carried out.

3.3 Experiments

In this section, we have three tasks to do. The first task is

Table 1. Performances when prototypes are clustered into 400.

Number of Clusters: 400 (Training pattern set)				
Number of Candidate clusters	Coarse Classification Rate	Coarse Classification Time (msec.)	Whole Recognition Rate	Whole Recognition Time (msec.)
1	50.2	3.16	50	4.65
2	63.9	3.22	63.7	6.76
5	79.8	3.39	79.4	8.32
10	88.5	3.75	88	9.78
20	94.1	4.15	93.5	10.2
30	96.3	4.57	95.5	10.6
40	97.2	4.96	96.5	11.08
50	97.7	5.34	97	11.46
60	98.2	5.72	97.4	11.85
70	98.4	6.09	97.7	12.2
80	98.6	6.46	97.9	12.59
90	98.8	6.82	97.9	12.96
100	98.9	7.18	97.9	13.32
110	99	7.52	98.1	13.68
130	99.1	8.17	98.2	14.31
150	99.1	8.83	98.2	15.04
170	99.1	9.46	98.2	15.64

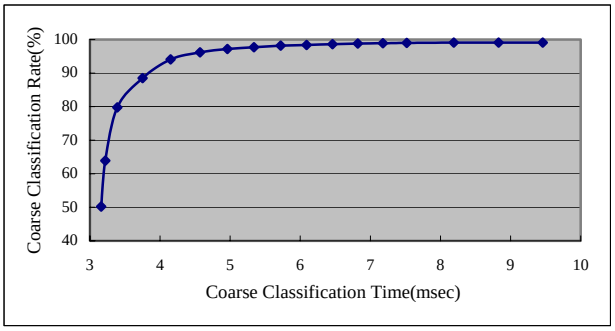

Fig. 3 Coarse classification performances for 400 clusters.

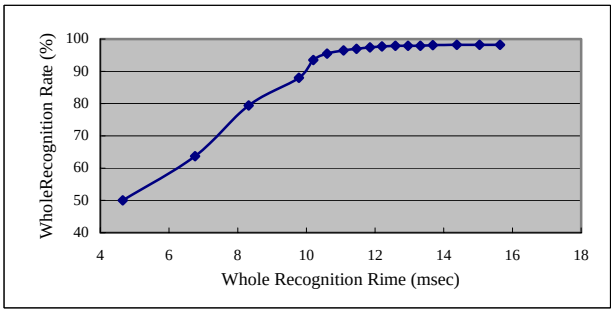

Fig. 4 Whole recognizer performances for 400 clusters

to find how many clusters the search space should be split into. The second is to know how many candidate clusters or pivots should be taken for comparison with respect to the number of clusters. The third is to decide the multiplying coefficient *m* for Algorithm-II and the constant *l* for the number of candidates for Algorithm-I. These should be considered to speed up the recognition time while keeping the recognition rate.

We start form the second task, then proceed to the first task and the third task.

3.3.1 Preparation for experiments

(1) Training set, testing set and environment

We use the same training set as that used to train the off-line character recognizer, i.e., 400 persons' handwritten character patterns x 3,214 categories (1,285,600 patterns) form the HP-JEITA database. As for the testing set, we prepared 100 persons' handwritten character patterns for 3.214 categories from the HP-JEITA database. These 100 writers are different from the writers for the training pattern set.

We made experiments on a PC with an Intel Pentium4 CPU of 2.4GHz and 512M RAM employing Microsoft Windows XP Professional.

(2) Original state of the recognizer

Before the use of SSS, the coarse classification of the original character recognizer produced 40 candidates with a coarse classification rate (the rate that the correct answer is within the candidates) of 99.1% in 13 milliseconds and the whole recognition produced 98.2% recognition rate in 20.1 milliseconds for the training set.

For the testing set, the coarse classification rate is 99.5% in 13 milliseconds and the whole recognition produced 97.7% recognition rate in 20.1 milliseconds.

3.3.2 Fixed number of candidate clusters

(1) Optimal number of candidate clusters

Table 1 shows performances of the coarse classifier and the whole recognizer with respect to various numbers of candidate clusters when the prototypes are divided into 400 clusters. Fig. 3 presents the relation between the rate and the time of the coarse classification.

The coarse classification rate goes up significantly up to about 96.3% as the number of candidate clusters increases up to about 30, then goes up gradually up to the best rate of 99.1% as the number of candidate clusters increases up to 130. This is less than 1/3 of the total cluster number (400), and the coarse classification time has been reduced to 8.19 msec. i.e., 63% from the original 13 msec. without sacrificing the recognition rate. When we look at the effect in the whole recognizer, 20.1 msec. is reduced to 14.38 (71.5%).

It would be needless to say that we can even speed up the recognition process if we can sacrifice the coarse classification rate and recognition rate to some extent as shown in Table 1 and Fig. 3 and Fig. 4.

Table 2 and Fig. 5 and Fig. 6 shows the result for 100 clusters. It shows almost the same relation and improvement as was obtained for 400 clusters.

Table 2. Performances when prototypes are clustered into 100.

Number of Clusters: 100(Training pattern set)				
Number of Candidate clusters	Coarse Classification Rate	Coarse Classification Time (msec.)	Whole Recognition Rate	Whole Recognition Time (msec.)
1	50.4	2.15	49.3	6.69
2	65.8	2.34	65.5	8.05
5	84	2.93	83.6	9.05
10	92.9	3.71	92.3	9.85
15	96	4.42	95.3	10.55
16	96.4	4.45	95.6	11.01
17	96.7	4.47	95.9	11.12
20	97.4	5.12	96.6	11.27
25	98.1	5.77	97.2	11.96
30	98.4	6.4	97.6	12.59
35	98.6	7	97.9	13.19
40	98.8	7.56	98	13.75
45	99	8.12	98.1	14.32
50	99.1	8.64	98.2	15.16
55	99.1	9.16	98.2	15.86

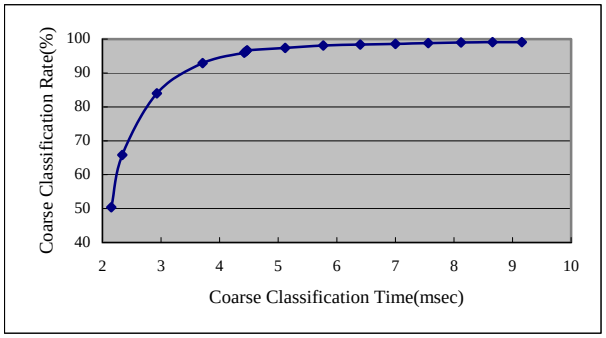

Fig. 5 Coarse classification performances for 100 clusters.

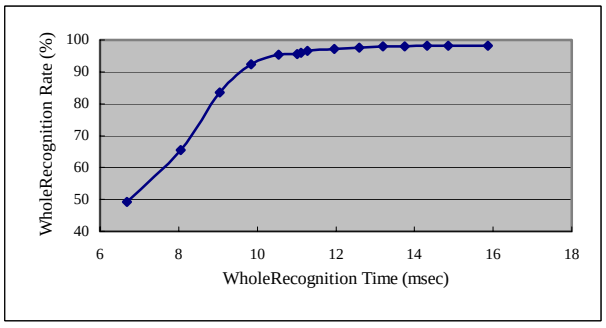

Fig. 6 Whole recognizer performances for 100 clusters

(2) Optimal clustering size for Method-I

Table 3 shows the performance for different choices of the number of clusters, and the number of candidate clusters that satisfy the condition that the coarse classification rate is around 99.1% and recognition rate around 98.2%.

Fig. 7 shows that the coarse classification time increases if the cluster number is smaller than 150 or bigger than 500. In the range 150-500, however, the coarse classification time is stable and reduced up to 8.11 msec.

Table 3. Performance when coarse classification rate is around 99.1% and whole classification rate is around 98.2%.

Number of Cluster	Number of Candidate cluster	Coarse Classification Time (msec.)	Whole Recognition Time (msec.)
2	2	24.4	14.3
50	26	16.56	9.66
100	50	15.16	8.64
150	65	14.59	8.36
200	70	14.39	8.21
250	84	14.09	8.11
300	95	14.1	8.13
350	115	14.16	8.15
400	130	14.31	8.17
450	135	14.37	8.23
500	138	14.45	8.48
550	141	14.63	8.61
600	145	14.75	8.9
650	146	14.94	9.095
700	153	15.12	9.31
900	160	16.14	10.01
1100	172	17.44	10.83
1300	180	18.46	11.64

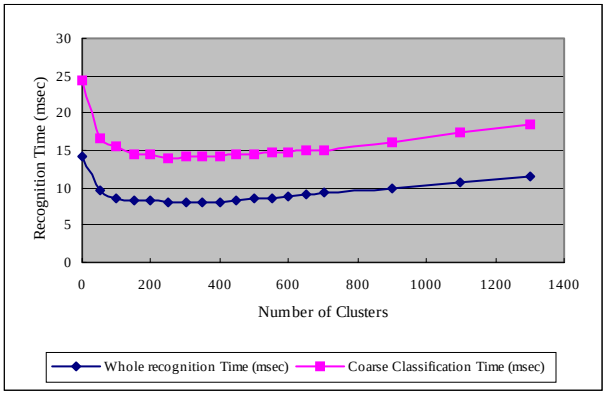

Fig. 7 Coarse classification and whole classification settings satisfying 99.1% rate for Method-I.

i.e., 62.4% from the original 13 msec. while the number of candidate clusters need to be linear to that of the clusters but it is only from 5 to 15% of the total number of clusters. When we look at the effect to the recognizer as a whole, 20.1 msec. is reduced to 14.09 msec. (70.1%) while achieving a 98.2% recognition rate.

3.3.3 Unfixed number of candidate clusters

(1) Optimal number of candidate clusters

Table 4. Recognition performances for multiplying coefficient *m*.

Number of Clusters: 400				
Multiplying Coefficient	Coarse Classification Rate	Coarse Classification Time (msec.)	Whole Recognition Rate	Whole Recognition Time (msec.)
1	50.2	3.16	50	4.82
1.05	61.6	3.19	61.2	5.71
1.1	71.9	3.25	71.4	6.6
1.15	80.2	3.33	79.6	7.34
1.2	86.3	3.45	85.7	7.92
1.25	90.6	3.62	90	8.42
1.3	93.6	3.84	92.9	8.67
1.35	95.5	4.1	94.8	9.32
1.4	96.8	4.41	96	9.75
1.45	97.6	4.77	96.6	10.21
1.5	98.1	5.58	97.3	10.71
1.55	98.5	6.02	97.6	11.2
1.6	98.8	6.46	97.9	11.7
1.65	98.9	6.92	98	12.2
1.7	99	7.8	98.1	12.74
1.75	99.1	8.09	98.2	13.29
1.8	99.1	8.63	98.2	13.7

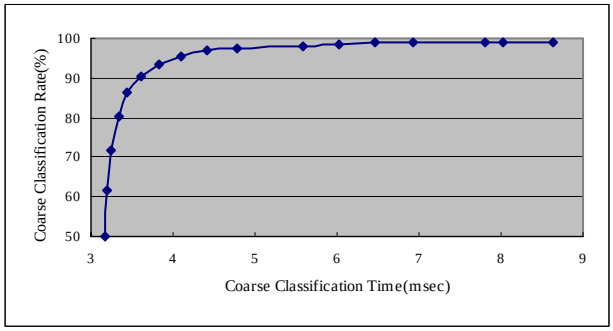

Fig.8 Coarse classification performances for 400 clusters.

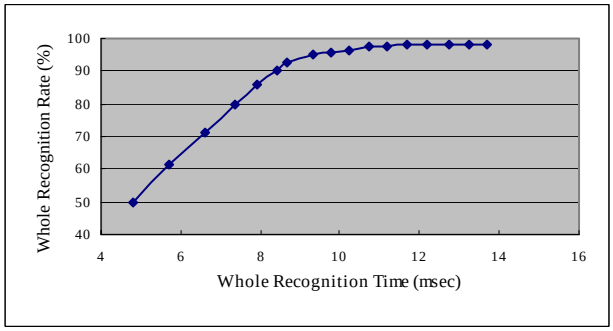

Fig.9 Whole recognizer performances for 400 clusters

This section considers the performances when the number of candidate clusters is not fixed and determined according to Method-II described in 3.2.1.

Table 4 , Fig. 8 and Fig.9 show the performances of the coarse classifier and the whole recognizer with respect to the range of the multiplying coefficient *m* when the prototypes vectors are divided into 400 clusters. When the multiplying coefficient *m* is 1.75, the same coarse classification rate as the original is achieved for 8.02 msec. That is 62% of the original 13 msec. and the same recognition rate is realized in 13.24 msec. (65.9%) from 20.1msec.

Table 5. Performances when coarse classification accumulative rate is around 99.1% and whole classification rate is around 98.2%

Cluster Number	Multiplying Coefficient	Coarse Classification Time (msec.)	Whole Classification Time (msec.)
2	1.7	10.31	16.1
50	1.675	9.03	14.87
100	1.7	8.62	14.25
150	1.715	8.19	13.71
200	1.715	8.01	13.49
250	1.71	7.98	13.31
300	1.745	8.08	13.28
350	1.765	8.11	13.35
400	1.75	8.09	13.29
450	1.785	8.17	13.29
500	1.875	8.21	13.25
550	1.785	8.31	13.26
600	1.785	8.36	13.28
650	1.795	8.47	13.29
700	1.8	8.53	13.32
900	1.84	8.79	14.01
1100	1.85	9.51	14.85
1300	1.85	10.1	15.19

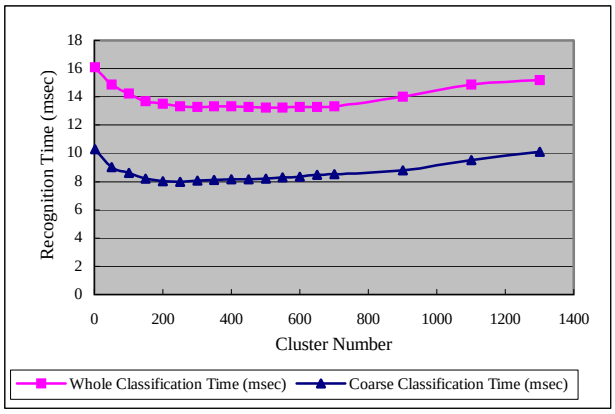

Fig. 10 Pre-classification settings satisfying 99.1% rate for Method-II.

(2) Optimal clustering size for Method-II

Table 5 shows the coarse classification time and the whole recognition time when the coarse classification rate is around 99.1% and the recognition rate is around 98.2%. It is achieved for different clustering numbers (2 to 1,300) for the prototypes with the multiplying coefficient adjusted. Fig. 10 summarizes them.

Although the whole recognition time is linear to the coarse classification time in the meaningful range of the number of clusters in Method I, here we can see that the whole recognition time is almost flat regardless of the coarse classification time.

It is clear that the least coarse classification time can be obtained when the number of clusters is around 250, and the coarse classification time increases fast as clusters are less than 150 and increases gradually as they are larger than 500. For the whole recognition time, however, we find that the minimum time is achieved when the number of clusters is 500. This means that the best point for the fine recognition time is shifted from that of the coarse classification and has a relatively wide optimal range.

The reason of these can be explained as follows. When Method II is used to select candidate clusters, there might be a few clusters near the input pattern for some test patterns with each cluster having only a small number of prototype (as the number of clusters increases, the number of prototypes in each cluster decreases), thus the total number of candidate prototypes could less than 40, which leads to the final recognition time reduced too. While the number of clusters is 250, although the coarse classification time is shortest, there are almost 40 candidate prototypes need to be transferred to the fine classification for every input pattern. While the number of clusters is 500 or more, the coarse classification time is slightly longer, but the number of candidate prototypes is likely less than 40 with the result that the recognition time is less than that when the number of clusters is 250. As the number of clusters increases from 250, the coarse classification time increases gradually, but the final classification time decreases due to the above reason, so the increase of the total recognition time is very small.

Although the characteristic of Method II is a bit different from Method I, we can also obtain similar advantages using Method I.

When we divide it into 250 clusters, the coarse classification time is reduced to 7.98 msec. (i.e., 61.4% from 13 msec.) and the whole recognition time reduced from 20.1 msec. to 13.31 msec. (66.1%) while achieving recognition rate of 98.2%.

When we divide it into 500 clusters, the coarse classification time is reduced to 8.21 msec. i.e., 63.2% from 13 msec. and the whole recognition time reduced from 13.24 msec. to 20.1 msec. (66%) while achieving a 98.2% recognition rate. Since the total performance is more important, this condition should be selected.

The multiplying coefficient is stable to produce the best performances if it is in the range 1.7-1.8.

3.3.4 Evaluation on the testing set

Through the experiments above, we have understood the characteristic of SSS, obtained the necessary parameters of

the two candidate selection methods and found out the optimal number of clusters. Then, we adopt these obtained data to test the testing set in order to further verify the characteristic of SSS. The result shows that the coarse classification and whole recognition are accelerated too, and the characteristic of SSS is valid to the test patterns as well as the training patterns.

Looking at the table 6, when we adopt the candidate selection method-I and corresponding parameters, the coarse classification time is reduced to about 63% from the original 13 msec., and the whole recognition time is reduced to 14.38 msec. (71.1% of original 20.1 msec.), without sacrificing the original recognition rate at all. If we sacrifice the rates we can get further reduction of processing time.

When we adopt the candidate selection method-II and corresponding parameters, if we keep the recognition rate as original, the acceleration effect is as same as for the training set, If we allow the rates degraded, we can get further speed-up.

4. Hybrid candidate selection

Although the two candidate selection algorithms produced almost the same results, detailed examination revealed different characteristics between them.

When Algorithm-I is employed, the final prototype candidates in the candidate clusters can be limited to a relatively small and stable range, but this algorithm must spend much time to sort all the candidates by their scores

Table 6. Compare of SSS between the training set and the testing set

		Coarse Classification Time (msec.)	Coarse Classification Rate (%)	Whole Recognition Time (msec.)	Whole Recognition Rate (%)
Candidate Cluster Selection Method-I	Training Set	8.21	99.1	14.39	98.2
	Testing Set	8.25	99.5	14.38	97.7
Candidate Cluster Selection Method-II	Training Set	8.01	99.1	13.24	98.2
	Testing Set	8.04	99.5	13.31	97.7

(obtained through comparing them with an input pattern). On the other hand, Algorithm-II does not need to sort the candidates, but the number of candidates has a wide variable range, for example, from several tens for some input patterns to nearly 1,100 for others.

Here, we also propose a hybrid algorithm to use the advantages of both the algorithms. At first, we use Algorithm-II for its high speed and reduce the number of candidates, and then use Algorithm-I to further reduce their number. Of course, for some categories of input patterns, if the output of Algorithm-II (the number of candidates for the fine classification) is less than m of Algorithm-I, Algorithm-I is skipped.

Hybrid candidate selection is applicable to the SSS stage and the coarse classification stage in the diagram shown in Fig. 2.

4.1 Experiments for SSS

We employ the values of the multiplying coefficient m for Algorithm-II and the constant l for the number of candidates for Algorithm-I obtained through the experiments in section 3.3.

Since the objective of the following experiments is to test the effect of hybrid candidate selection on acceleration, we do not need to test all the possible divisions of the prototypes into a different number of clusters. We only choose the case as the base experiment, in which the prototypes are divided into 400 clusters

At the points with original recognition rate, after we adopted the “1.75” and “130” in the hybrid method, the coarse classification time was further reduced to about 8.06 msec. (62% of original 13 msec.) and whole recognition time was further reduced to about 12.44 msec. (61.9% of original 20.1 msec.). Both the coarse classification and recognition rates are kept to the original recognition rate.

This is because all the necessary prototypes were included within the candidates after using the multiplying coefficient method. When we later used the fixed candidate number method, the necessary prototypes were still included within the final candidates.

From the above experiments and analysis, we can affirm that the hybrid candidate selection is an effective method to accelerate the recognition while keeping the original recognition rate. For this case the coarse classification time was reduced to about 62%; the recognition time was reduced to about 61.9% of original time.

4.2 Experiments for coarse classification

Actually, in the coarse classification, it is also needed a candidate selection approach too. The original method only adopted fixed candidate number method. After adopting the hybrid method with original fixed candidate number 40 and multiplying coefficients 1.8, the coarse classification time was reduced to 7.28 milliseconds (56% of originial time); the whole recognition time was reduced to 10.5 milliseconds (52.2% of originial time) without sacrificing the coarse classification rate.

5. Conclusion

This paper presented “structuring search space” (SSS)

method in order to accelerate recognition of large character sets. We also considered two ways of candidate selection and finally combined them into the hybrid candidate selection method. We applied the SSS method to a practical off-line Japanese character recognizer and verified that the coarse classification time and the whole recognition time were reduced to 61.4% and 66%, respectively from the original system without sacrificing coarse classification and recognition rates. The hybrid candidate selection further accelerated the coarse classification and whole recognition time to 56% and 52% from the original system, respectively. If we are allowed to sacrifice recognition rate, we can further speed up the recognition process.

Acknowledgement

Thanks are due to Professor Yamamoto and other people for providing us with the database ETL-9 and HP-JEITA.

6. References

[1] S. Mori, K. Yamamoto, M. Yasuda: Research on machine recognition of handprinted characters, *IEEE PAMI*, Vol.6, No.4, 386-405, 1984.

[2] T. H. Hildebrandt, W. Liu: Optical recognition of handwritten Chinese characters: advances since 1980, *Pattern Recognition*, Vol.26, No.2, 205-225, 1993.

[3] T. Wakabayashi, Y. Deng, S. Tsuruoka, F. Kimura, Y. Miyake: Accuracy Improvement by Nonlinear Normalization and Feature Compression in Handwritten Chinese Character Recognition (in Japanese), *IEICE PRU*, 95-1(95-05).

[4] N. Sun, M. Abe, Y, Nemoto: A Handwritten Character Recognition System by Using improved Directional Element Feature and Subspace Method (in Japanese), *IEICE* Vol.J78-D-II, No6, pp.922-930, 1995-6.

[5] C. -L. Liu and M. Nakagawa: Precise candidate selection for large character set recognition by confidence evaluation, *IEEE PAMI*, Vol. 22, No. 6, 636-642, 2000.

[6] T. Kumamoto, et al: On speeding candidate selection in handprinted Chinese character recognition, *Pattern Recognition*, Vol.24, No. 8, 793-799, 1991.

[7] C. -H. Tung, H. -J. Lee, J. -Y. Tsai: Multi-stage pre-candidate selection in handwritten Chinese character recognition systems, *Pattern Recognition*, Vol.27, No.8, 1093-1102, 1994.

[8] Y. Yang, O. Velek, M. Nakagawa: Acclerating Large Character Set Recognition Using Pivots, *Proc. 7th ICDAR*, Vol, 4C, 262-267, 2003.

[9] Y. -H. Tseng, C. -C. Kuo, H. -J. Lee: Speeding up Chinese character recognition in an automatic document reading system, *Pattern Recognition*, Vol.31, No.11, 1601-1612, 1998.

[10] F. Fujimoto, H. kamada, K. Kurokawa: Fast, precise pre-classification method using projections of feature regions (in Japanese), PRMU97-220 (1998-02).

[11] J. Tsukumo, H. Tanaka: Classification of handprinted Chinese characters using non-linear normalization and correlation methods, *Proc. 9th ICPR*, Roma, Italy, 168-171, 1988.

[12] F. Kimura: Modified quadratic discriminant function and the application to Chinese characters, *IEEE PAMI* Vol.9, No.1, 149-153, 1987.

[13] Y. Linde, A. Buzo, and R. M. Gray: An algorithm for vector quantization design, *IEEE Trans. on Communications*, Vol. COM-28, 84-95, 1980.

Yiping Yang received the B.S. degrees in Electrical Engineering from Guizhou University of China in 1996 and M.Sc. degree from Tokyo University of Agriculture and Technology (TUAT) of Japan in 2003. He is currently a Ph.D. student in the Graduate School of Engineering at TUAT. He is working on off-line and on-line character recognition of a large character set.

Bilan Zhu received M.Sc. degree from Tokyo University of Agriculture and Technology, Tokyo (TUAT), Japan in 2004. She is currently a Ph.D. student in the Department of computer, information and communication sciences at TUAT. She is working of forms processing, on-line, off-line handwriting recognition and context analysis of Japanese, Chinese and English text.

Masaki Nakagawa was born on 31 October 1954 in Japan. He received B. Sc. and M. Sc. degrees from the University of Tokyo in 1977 and 1979, respectively. During the academic year 1977/78, he followed Computer Science course at Essex University in England, and received M.Sc. with distinction in Computer Studies in July 1979. He received Ph.D. in Information Science from the University of Tokyo in December 1988. From April 1979, he has been working at Tokyo University of Agriculture and Technology. Currently, he is a Professor of Media Interaction in Department of Computer, Information and Communication Sciences.