December 9, 1998
RT0290
Human-Computer Interaction 8 pages

Research Report

Automatic Allograph Categorization Based on Stroke Clustering for On-Line Handwritten Japanese Character Recognition

Kazutaka Yamasaki

IBM Research, Tokyo Research Laboratory
IBM Japan, Ltd.
1623-14 Shimotsuruma, Yamato
Kanagawa 242-8502, Japan

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

Automatic Allograph Categorization Based on Stroke Clustering for On-Line Handwritten Japanese Character Recognition

Kazutaka Yamasaki
Tokyo Research Laboratory, IBM Japan,
1623–14 Shimotsuruma, Yamato, Kanagawa 242, Japan
yamasakk@trl.ibm.co.jp

Key words: on-line handwritten Japanese character recognition, allograph, clustering, prototype stroke

Preferred conference track: Character recognition

Abstract: In order to build a recognition dictionary so that the dictionary includes various writing styles, an automatic method for categorizing writing styles of characters (allographs) is proposed. In the first step of allograph categorization, we categorize handwritten strokes in training data by using a clustering algorithm that is also proposed in this paper. After execution of the algorithm, the centroid of a cluster is referred to as the prototype stroke. By using prototype strokes, we categorize handwritten characters to obtain allographs. In this approach, allographs share common prototype strokes. This allows us to reduce the dictionary size and computational cost of recognition. Furthermore, we can compare two allographs to determine where the stroke order is different and which strokes are connected. In the stroke clustering, the number of clusters are automatically determined on the basis of one parameter Δ that we give before the clustering procedure. The parameter Δ is the maximum error in a stroke cluster. Allograph dictionaries for 2321 categories were experimentally made by using handwritten characters produced by 121 writers. Recognition experiment by using these dictionaries were carried out, so that the relations between the parameter Δ, the number of prototype strokes, the number of allographs, and recognition accuracy were obtained.

1 Introduction

Recent software and hardware technologies have made it possible for personal computers to incorporate natural user interfaces, including speech synthesis, speech recognition, and character recognition. Products with these interfaces are available on the market, and are reasonably priced for end users. Although current recognition technologies provide better accuracy than before, a considerable amount of error corrections is needed for entry of real-world data such as reports, business correspondence, and personal messages. In the field of on-line Japanese handwritten character recognition, errors are mainly caused by personal variations in writing style, including stroke connections and stroke order permutations. Furthermore, it is easily observed that the same person often writes characters differently, with shape distortions and connected strokes, when his writing speed increases or when his physical or mental condition changes.

For correct handling of shape distortion in handwritten characters, shape and size normalization methods have been proposed in the areas of off-line and on-line character recognition [9][7]. In order to cope with stroke connections and stroke order permutations, methods for finding the stroke correspondence between a reference character and an unknown one have also been proposed [3][8]. These approaches, which can be regarded as methods for reducing style variations, allow us to reasonably compare an unknown character with a reference even when a simple stroke-based matching algorithm does not work.

Another approach is to keep variations in a recognition dictionary, so that an unknown character can be compared with a reference in the dictionary. This approach includes statistical methods such as hidden Markov modeling. The model can learn the distributions of feature vectors from training data only when a suitable model structure such as a suitable number of states and reasonable connections between the states is given before a training procedure. In order to give the model structure, we need to know various styles of characters, which are termed "allographs." For algorithms that search the large space of stroke correspondence combinations [3][8], a knowledge of allographs is necessary to reduce the computational cost to a reasonable level by neglecting unreal correspondences.

The problem of allograph categorization for on-line English words has been discussed in the context of style recognition [1], systematic naming schemes for allographs [6], segmentation of cursive script into letters [4], and so on. This paper addresses the problem for on-line handwritten Japanese character recognition. Our approach to allograph categorization is based on stroke categorization, because Japanese characters consist of several strokes. In the first step of allograph categorization, we categorize handwritten strokes in training data by using a clustering algorithm that is also proposed in this paper. After execution of the algorithm, the centroid of a cluster is referred to as the prototype stroke. By using prototype strokes, we can categorize handwritten characters to obtain allographs. In this approach, allographs share common prototype strokes. This allows us to reduce the dictionary size and computational cost of recognition. Furthermore, we can compare two allographs to determine where the stroke order is different and which strokes are connected.

The notion of a prototype stroke has been used in the structural analysis approach to recognize Japanese, Chinese, and Korean characters [2]. In this approach, a dictionary has a hierarchical structure in which a Kanji character consists of radicals, which in turn consist of prototype strokes. These structures and prototype strokes are manually determined on the basis of observations of handwritten characters and empirical knowledge of recognition. A semi-automatic training algorithm is proposed, to include various writing styles in a structured dictionary [5]. Because the training procedure includes a manual process, it is not feasible to use a large database for building a dictionary. In this paper, in contrast, allographs are automatically categorized so that a large database can be used to build a dictionary. Furthermore, the relation between the number of allographs and the recognition accuracy is experimentally shown.

2 Stroke categorization

We obtain stroke clusters such that the maximum error from the centroid, namely, the radius of a cluster, is less than a constant Δ. The number of clusters is locally minimized by using the following two clustering algorithms, so that we do not need to determine the number of clusters. The outline of the procedure is shown in Figure 1. A centroid – that is, an average stroke of a cluster, is called a prototype stroke after a categorization procedure is completed.

2.1 Feature extraction

The size of handwritten characters are normalized so that the circumscribed box is a square with sides length of L. The origin of the coordinate system is placed at the left bottom of the square. The x-axis is along the bottom edge of the square, while the y-axis is along the left side of the square. On each stroke, six equally spaced points that include the beginning and the end of the stroke are determined. By using the six points, a 12-dimensional feature vector $s = (x_1, y_1, \cdots, x_6, y_6)$ is obtained for each stroke.

2.2 Categorization based on number of strokes and stroke generation order

In general, there are two approaches to categorization. One is top-down splitting of clusters, and the other is bottom-up merging of clusters. Both approaches are used in the proposed algorithm, which is based on the

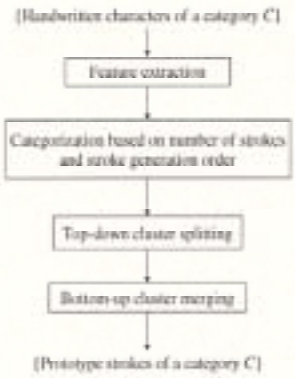

Figure 1: Procedure of stroke categorization.

Figure 2: Average strokes of category 田.

properties of handwritten Japanese characters. When we look at handwritten characters in a single category C that consists of N strokes, the n-th ($1 \leq n \leq N$) strokes are similar to each other except when the character has stroke order permutation or connected strokes. Because these variations are not randomly produced, the set of the n-th strokes is used as an initial cluster in the following procedure. For each (C, N, n), handwritten strokes are collected to make a stroke set S. Let S_0 be the initial cluster, that is, let $S_0 = S$.

Average strokes of the initial clusters S_0 when $C =$ 田 is shown in Figure 2. In the figure, a small circle at the head of each stroke shows the beginning of the stroke. The number that appears next to each stroke is the stroke generation order n. Handwritten characters were collected from 67 people. Only one character consisted of four strokes, while the rest consisted of of five strokes. The figure for $N = 5$ shows that the third and fourth strokes are tilted abnormally. This is because the character 田 has more than one kind of stroke order. Some people write the vertical stroke before the horizontal one, while others write the horizontal one before the vertical one. Another possible cause of tilted strokes is writer-dependent shape distortion. Such strokes can be automatically identified, as explained in the next section.

2.3 Top-down cluster splitting

For each (C, N, n), we split the initial cluster S_0. Let $P(s)$ be the cluster to which a stroke s belongs. Let $p(s)$ be the average stroke of a cluster $P(s)$. Let D be a partition of S_0, that is, the set of clusters. We find a partition D such that the following condition holds:

$$max_{s \in S} \|s - p(s)\| \leq \Delta. \tag{1}$$

If the above condition (1) does not hold, categorization is not completed. The number of clusters is incrementally increased one by one until condition (1) holds. A partition that satisfies condition (1) is not always unique. A criterion for the goodness of partition D is $\sum_{s \in S} \|s - p(s)\|^2$. An algorithm based on the above strategy is shown below:

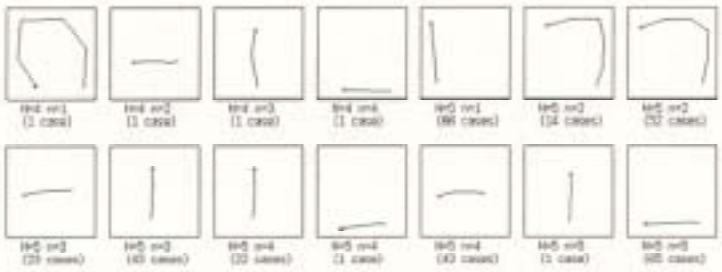

Figure 3: Average strokes of category 田 after cluster splitting.

Step 1: Initialize partition.

$D = \{S_0\}$

Step 2: Get the maximum error.

$\max\_error = \max_{s \in S} \|s - p(s)\|$

$s_1 = \mathrm{argmax}_{s \in S} \|s - p(s)\|$

Step 3: If $\max\_error \leq \Delta$ then

stop the algorithm,

else

go to Step 4.

Step 4: Get the most different stroke s_2 from s_1.

$s_2 = \mathrm{argmax}_{s \in P(s_1)} \|s - s_1\|$

Step 5: Split $P(s_1)$ into S_1 and S_2 on the basis of the distances from the two strokes s_1 and s_2.

Step 6: Remove $P(s_1)$ from D, and add S_1 and S_2 to D.

Step 7: Do k-means clustering to decrease $\sum_{s \in S} \|s - p(s)\|^2$.

Step 8: Go to Step 2.

The above algorithm provided average strokes of category 田 shown in Figure 3, where $\Delta = 0.6L$. When $N = 4$, the clusters are not split because there is only one case. When $N = 5$, the set of first strokes ($n = 1$) is not split. This result is consistent with the observation that all first strokes are vertical strokes in the left of the normalized box. The set of the second strokes ($n = 2$) is split into two sets, but the two average strokes are similar to each other. The set of third strokes ($n = 3$) is split into two sets, one consisting of vertical strokes and the other of horizontal ones. The set of fourth strokes ($n = 4$) is split into three sets, one consisting of vertical strokes and two of horizontal ones. The set of fifth strokes ($n = 5$) is split into two sets, one consisting of vertical strokes and the other of horizontal ones. Among the five average strokes for $n = 4$ and $n = 5$ in the Figure 3, two are consist of only one case. This is caused by a stroke order permutation that was found only in one handwritten character. This algorithm does not require many examples in order to identify a stroke order permutation.

2.4 Bottom-up cluster merging

Looking at average strokes of category 田 in Figure 3, we can easily find similar average strokes in different (N, n). This is because stroke order permutation and stroke connection do not change the shapes of all strokes. Many of the strokes remain unchanged. Merging these stroke sets allows allographs to share prototype strokes, so that the size of a recognition dictionary is reduced. We merge stroke sets under the condition (1) used for the top-down cluster splitting in Section 2.3. Let $p(S)$ be the average stroke of a cluster S.

Figure 4: Average strokes of category 田 after cluster merging.

Step 1: Initialize a set of clusters.

D = (The set of all clusters of a category C)

Step 2: Get candidates for merging.

E = (The set of all the cluster pairs to which the following Step 5 has not been applied)

Step 3: If E is empty set, then

stop the algorithm,

else

go to Step 4.

Step 4: Get the most similar cluster pair (S_1, S_2) in the set E.

$(S_1, S_2) = \mathrm{argmin}_{(S_1,S_2)\in E}\|p(S_1) - p(S_2)\|$

Step 5: Merge the two clusters S_1 and S_2.

$S_3 = S_1 \cup S_2$ (Union of S_1 and S_2)

Step 6: Get the maximum error of the new cluster S_3.

$\mathrm{max\_error} = \max_{s\in S_3} \|s - p(s)\|$

Step 7: If $\mathrm{max\_error} \leq \Delta$ then

remove S_1, S_2 from D, and add S_3 to D.

Step 8: Go to Step 2.

The average strokes of category 田 obtained by the above algorithm are shown in Figure 4. In the figure, Px ($x = 1, \cdots, 7$) denotes the x-th average stroke. In Section 2.3, fourteen average strokes were obtained from 67 handwritten characters by using the cluster splitting algorithm. The cluster merging algorithm reduced the number of clusters to 7. The merged strokes are vertical and horizontal ones in Figure 3. This is consistent with the observation that these strokes are similar to each other.

3 Allograph categorization

3.1 Allograph and prototype stroke

Each stroke in the training data always belongs to a cluster obtained in the previous section and does not belong to more than one cluster. The above stroke categorization directly leads to categorization of handwritten characters in the training set. Replacing the strokes of the handwritten characters by the prototype strokes, we obtain categorized allographs.

Allographs of category 田 are shown in Figure 5. Note that the number that appears next to each stroke is the stroke generation order n. The names of the prototype strokes are shown in Figure 6 so that it can be clearly seen which strokes are shared by two or more allographs. Sixty-seven handwritten characters are classified into six allographs. 田 2 and 田 3 consist of the same prototype strokes, but with mutually different stroke orders. 田 4, 田 5, and 田 6 also have mutually different stroke orders. If we compare 田 1 and 田 2, we can see that the first stroke P1 of 田 1 may be a connected stroke consisting of the first two strokes P5 and P6 of 田 2. The connected stroke may consist of the first two strokes P5 and P7 of 田 4.

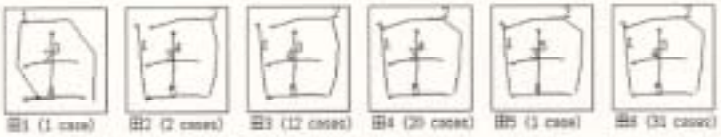

Figure 5: Allographs of category 田

Figure 6: Allographs of category 田 as a sequence of the names of the prototype strokes.

Table 1: Data used for experiments.

	Training data	Test data
Number of categories including alphanumeric, Kana, and Kanji characters	2321	2321
Number of handwritten character examples	193661	29389
Number of handwritten character examples per category	68–242	10–40
Number of writers	121	20

3.2 Allographs and the maximum error Δ in a stroke cluster

The relations between the number of prototype strokes, the number of allographs, and the maximum error Δ in a cluster, which is the only parameter for allograph categorization, are shown experimentally. The data shown in Table 1 were used for the following experiment. The value of the parameter Δ is common for all categories. Varying the value Δ from $0.2L$ to $0.8L$, where L is the length of a side of the normalized circumscribed square, fourteen allograph dictionaries were made. For each value of Δ, two dictionaries were made. One was made after executing the top-down cluster splitting algorithm in Section 2.3 and the other after executing the bottom-up cluster merging algorithm in Section 2.4.

Averages of the numbers of prototype strokes per category are shown in Table 2. Average of the number of handwritten stroke examples per category in the training data, which was independent of Δ, is also shown at the second column of the table. The average of the value of N (the number of strokes in a character) was 8.1, which was about half the number of prototype strokes after bottom-up cluster merging when $\Delta = 0.5L$. The number of prototype strokes after top-down cluster splitting is 7% – 21% of the number of handwritten stroke examples. The number of prototype strokes after bottom-up cluster merging is 16% – 63% of the one after top-down cluster splitting.

Table 2: Average of the number of prototype strokes per category.

Parameter Δ	Number of handwritten stroke examples	Number of prototype strokes	
		After top-down cluster splitting	After bottom-up cluster merging
$0.2L$	678.5	139.8	88.5
$0.3L$	678.5	88.9	42.7
$0.4L$	678.5	66.2	24.8
$0.5L$	678.5	55.2	16.4
$0.6L$	678.5	49.9	11.9
$0.7L$	678.5	46.8	9.1
$0.8L$	678.5	44.8	7.1

Table 3: Average of the number of allographs per category.

Parameter Δ	Number of handwritten character examples	Number of allographs	
		After top-down cluster splitting	After bottom-up cluster merging
$0.2L$	83.4	75.4	74.7
$0.3L$	83.4	66.4	63.8
$0.4L$	83.4	45.2	40.2
$0.5L$	83.4	21.6	18.7
$0.6L$	83.4	12.9	11.7
$0.7L$	83.4	9.6	9.1
$0.8L$	83.4	8.0	7.0

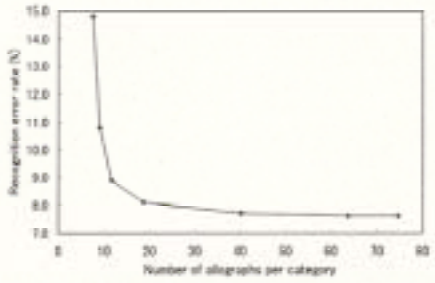

Figure 7: The relation between the recognition error rate and the number of allographs.

After the merging algorithm has been executed, a prototype stroke sometimes appears more than once in an allograph. For example, a prototype sequence P1, P1, P2, P3 including two instances of P1 can be found in the dictionaries. It is not usual for the same stroke to be written twice in the real world. This over-merging reduces the number of allographs, as shown in Table 3. In the table, the average of the number of handwritten character examples per category in the training data, which is independent of Δ, is also shown at the second column. The numbers of allographs after bottom-up cluster merging is 9% – 90% of the number of handwritten character examples.

A recognition experiment was carried out to determine the quality of allograph dictionary, using the test data shown in Table 1. Let us assume that the difference between a handwritten character and a reference is $\sum_{n=1}^{N} \|s_n - p_n\|^2$, where s_n and p_n are n-th strokes of an unknown character and of an allograph, respectively.

By using the seven dictionaries made after bottom-up cluster merging where $\Delta = 0.2L, \cdots 0.8L$, the relation between the recognition error rate and the number of allographs was obtained and is shown in Figure 7. In the figure, the error rate is monotonically decreased when the number of allographs is increased to 63.8. When it is more than 63.8, no improvement of the error rate is seen. When the error rate is less than 8.0%, we can reduce the number of allographs at the expense of small degradation of the accuracy.

The relation between the recognition error rate and the number of prototype strokes was also obtained and is shown in Figure 8. When the error rate is 8.0%, the number of prototype strokes after bottom-up cluster merging is one third of the one after top-down cluster splitting. In other words, the bottom-up cluster merging algorithm reduces prototype strokes, while keeping the accuracy.

4 Conclusion

A method of automatic allograph categorization has been proposed. Allographs are defined by using prototype strokes, which are obtained by a proposed method of stroke clustering. In the clustering procedure, both top-down cluster splitting algorithm and bottom-up cluster merging algorithm are used so that prototype strokes are shared by allographs of each category. Comparing these allographs allows us to find stroke connections

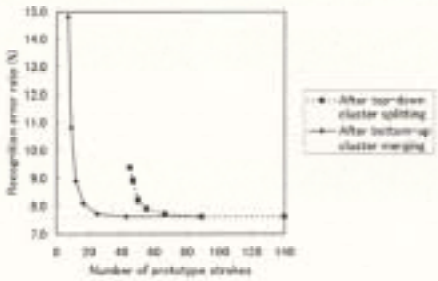

Figure 8: The relation between the recognition error rate and the number of prototype strokes.

and stroke order permutations. Allograph dictionaries for 2321 categories were experimentally made by using handwritten characters produced by 121 writers. Relations between the maximum error Δ in a stroke cluster, the number of prototype strokes, and the number of allographs were experimentally obtained. Recognition experiments were carried out by using the dictionaries and test data from 20 writers. It was found that the bottom-up cluster merging algorithm reduces prototype strokes, while keeping the accuracy.

References

[1] J. Crettez: "A set of handwritten families: style recognition," Proc. of the Third ICDAR, Aug. 14–16, 1995, Montreal, Canada, pp. 489–494.

[2] P. Kim and H. Kim: "On-line recognition of run-on Korean characters," Proc. of the Third ICDAR, Aug. 14-16, 1995, Montreal, Canada, pp. 54–57.

[3] J. Shin and H. Sakoe: "Cubic search algorithm for stroke-order and stroke-number free on-line character recognition," Technical Report of IEICE Japan, vol. PRU96-84, pp. 29–36, Nov. 1996.

[4] H. Teulings and L. Schomaker: "Unsupervised learning of prototype allographs in cursive script recognition, " From Pixels to Features III: Frontiers in Handwritten Recognition, eds. S. Impedovo and J. Simon, pp. 61–73.

[5] L. Tu and M. Nakagawa: "Structured learning of characters for on-line recognition of handwritten Japanese characters," Technical Report of IEICE Japan, vol. PRU95-165, pp. 43–48, Nov. 1995.

[6] L. Vuurpijl and L. Schomaker: "Finding structure in diversity: A hierarchical clustering method for the categorization of allographs in handwriting," Proc. of the Fourth ICDAR, Aug. 18-20, 1997, Ulm, Germany, pp. 387–393.

[7] T. Wakahara and K. Odaka: "Adaptive normalization of handwritten characters using global/local affine transformation," Proc. of the Fourth ICDAR, August 18-20, 1997, Ulm, Germany, pp. 28–33.

[8] T. Wakahara et al.: "Stroke-number and stroke-order free on-line Kanji character recognition as one-to-one stroke correspondence problem," IEICE Trans. Inf. & Syst., vol. E79-D, No. 5, pp. 529–534, May 1996.

[9] H. Yamada, K. Yamamoto, and T.Saito: "A nonlinear normalization method for handprinted Kanji character recognition – line density equalization," Pattern Recognition, vol. 23, pp. 1023–1029, 1990.