PAPER

High Speed and High Accuracy Pre-Classification Method for OCR: Margin Added Hashing

Yutaka KATSUYAMA[†a)], **Yoshinobu HOTTA**[†], ***Members***, **Masako OMACHI**[††], ***and*** **Shinichiro OMACHI**[†††], ***Senior Members***

SUMMARY Reducing the time complexity of character matching is critical to the development of efficient Japanese Optical Character Recognition (OCR) systems. To shorten the processing time, recognition is usually split into separate pre-classification and precise recognition stages. For high overall recognition performance, the pre-classification stage must both have very high classification accuracy and return only a small number of putative character categories for further processing. Furthermore, for any practical system, the speed of the pre-classification stage is also critical. The associative matching (AM) method has often been used for fast pre-classification because of its use of a hash table and reliance on just logical bit operations to select categories, both of which make it highly efficient. However, a certain level of redundancy exists in the hash table because it is constructed using only the minimum and maximum values of the data on each axis and therefore does not take account of the distribution of the data. We propose a novel method based on the AM method that satisfies the performance criteria described above but in a fraction of the time by modifying the hash table to reduce the range of each category of training characters. Furthermore, we show that our approach outperforms pre-classification by VQ clustering, ANN, LSH and AM in terms of classification accuracy, reducing the number of candidate categories and total processing time across an evaluation test set comprising 116,528 Japanese character images.

key words: *pre-classification, associative matching method, OCR, clustering, Hash table*

1. Introduction

Recently, the speed of document scanners has been steadily increasing, allowing more documents to be processed in a shorter time. Corporations often use such scanners to register documents in Document Management Systems (DMS), for example as provided by enterprise content management software. In this scenario, not only is the speed at which new documents can be registered in the system critical, but also often just as important is the ability to search the often large corpus of stored documents efficiently to allow the quick retrieval of data, for example, when auditing accounts. Therefore, a fast OCR system with high recognition rate is desirable to facilitate this kind of search. For Japanese documents, the design of fast OCR systems is particularly challenging, because there are almost 5,000 character categories in written Japanese. Reducing the time complexity of the matching process is one of the most important challenges in the development of efficient Japanese OCR systems. In order to achieve fast character recognition, a Coarse-to-Fine strategy [20] is effective in attaining high recognition accuracy and a short processing time. With this strategy, total recognition processing consists of two steps as described in Fig. 1. In step one, pre-classification chooses candidate categories out of all the categories in the dictionary. Step two is precise recognition which selects one category as the recognition result from the categories that were chosen in step one.

By pre-classification, the number of candidates can quickly be reduced to a small set of categories that resemble the input query. It is highly probable that the correct match is in one of these selected categories. In this way, pre-classification serves an important role in reducing time consumption and increasing accuracy. When considering Japanese OCR, three essential requirements listed as follows are placed on the pre-classification stage.

1. Highly accurate classification.
   Because pre-classification performs the preprocessing for the precise recognition stage, it is necessary to choose a correct category with almost 100% accuracy from candidate categories.
2. High-speed.
   The total recognition processing time is composed of pre-classification processing time and precise recognition processing time. Pre-classification must be executed quickly for the total recognition processing time.
3. Reduced number of candidate categories.
   Without drastically reducing the size of the candidate pool, the precise recognition processing will spend a

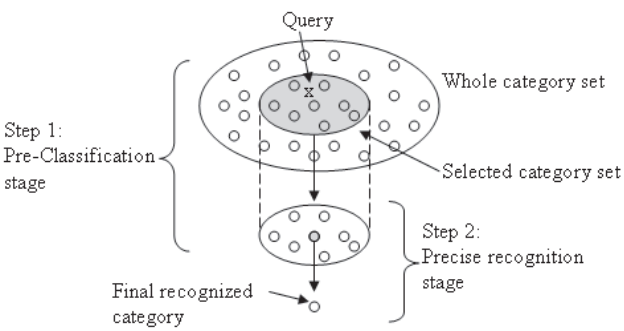

Fig. 1 Coarse-to-Fine strategy for recognition.

Manuscript received December 25, 2012.
Manuscript revised April 11, 2013.

[†]The authors are with the FUJITSU LABORATORIES LTD., Kawasaki-shi, 211–8588 Japan.
[††]The author is with the Sendai National College of Technology, Natori-shi, 981–1239 Japan.
[†††]The author is with the Tohoku University, Sendai-shi, 980–8579 Japan.
a) E-mail: katsuyama@jp.fujitsu.com

DOI: 10.1587/transinf.E96.D.2087

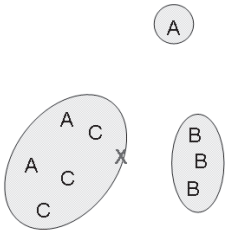

Fig. 2 VQ Clustering.

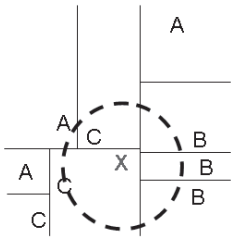

Fig. 3 ANN.

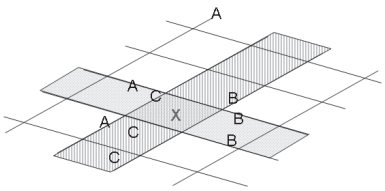

Fig. 4 LSH.

significant amount of time attempting to match the input character against a large number of candidates.

Since there is a trade-off between accuracy and the number of selected candidates, several techniques have been applied for pre-classification to address this issue. The Nearest Neighbor (NN) search method can be regarded as a pre-classification method because it selects candidate categories close to the input query. Other conventional pre-classification methods such as VQ Clustering, ANN, LSH, and Associative Matching method have also been proposed. In standard k-NN [2], the distance between a query vector and each category's vector in the dictionary must be computed. The k nearest vectors then becomes the candidate categories. Clustering methods such as k-means clustering [3] and LVQ based clustering [1] can be used to reduce the number of distances which must be calculated. Clustering can be performed offline, where multiple categories which are close to each other in the feature space are assigned to the same cluster center, as shown in Fig. 2. Here, training data of categories A, B, and C are clustered as shown in Fig. 2. The cluster whose center is the closest to the input query X is marked and the categories in the cluster are selected as output of pre-classification. However, although this method is faster than standard k-NN, it is still slower than the hashing methods described below, since the distances between the query vector and each of the representative cluster vectors must still be calculated.

Another approach to clustering involves tree based methods such as kd-tree [2], [4], [22]–[24], R-tree [5], [22], [23], SS-tree [6], [22], SR-tree [7], [22], Sp-tree [8], MPA [21], and ANN [9], [10], [22]–[24]. The ANN method uses kd-tree structure as a dictionary that divides feature space into a lot of cells. Then the categories in the cells that exist in a constant distance from the query vector are selected (Fig. 3). In Fig. 3, the input query X is dropped in a cell of category C. Then, the distance between X and the training vector of C in the cell is calculated. The categories of the cells which overlap with the circle which centers on X and makes the distance a radius are selected as output of pre-classification. Concerning the three requirements for effective pre-classification, since the cells completely cover feature space, ANN can select the correct category to a very high degree of accuracy. However, because this method must make comparisons by calculating discrimination against many nodes in kd-tree, the processing time exceeds that of hashing methods and is subsequently outperformed by them.

Locality-Sensitive Hashing (LSH) method [11]–[15], [23]–[25] projects feature vectors into a lower-dimensional space by using many hash functions that are generated at random. Hash tables are built in the subspaces. When a query vector is input, hash functions are used to project the input vector onto the hash table. As a result of pre-classification, LSH selects categories which appear in at least one of hash tables. In Fig. 4, two random axes and buckets on each axis are shown. The bucket that includes the input query X on each axis is marked as shown by dark color. The categories in the marked buckets are selected as output of pre-classification. Concerning the three pre-classification requirements, we see that this method is faster than the methods mentioned above, because it does not depend on distance calculations to select candidate categories, relying only on projection and OR operations instead. This method is also highly accurate because it uses multiple redundant axes that are not orthogonal to each other and are generated randomly. However, this redundancy produces a larger number of selected candidate categories than other methods.

Associative matching (AM) method [16] uses a hash table that is built by projecting feature vectors on top of each orthogonal axis, and is described by bit array. Each category's range on an axis is defined by a minimum and maximum value on the axis, and the range is expanded by adding a margin. In the classification phase, the query vector is classified as all the categories whose range it falls inside (Fig. 5). In Fig. 5, there are three boxes that show ranges of category A, B, and C. Each range is built by minimum and maximum value which are expanded by a margin on each axis of training sample of each category. The categories of boxes in which input query X dropped are selected as output of pre-classification. Regarding the pre-classification requirements, this method achieves desirable accuracy and is faster than the methods mentioned above. This is because the hash dictionary on each axis can be represented as a bit

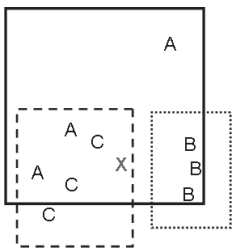

Fig. 5 AM.

Table 1 Parameters.

Notation	Meanings
k	Axis.
ϕ_k	Eigen vector of k-th axis.
ω	Category, $\omega = 1, 2, \cdots, max\_category$.
x_ω^j	The j-th training vector of category ω
N_ω	Number of training vector of category ω
i	Index to bit array on an axis, $i = 1, 2, \cdots, max\_index$.
max_index	Maximum index value that is the same on every axes.
max_axis	Dimensionality of a feature vector.

array. This allows for logical AND operations to be used for classification which is more efficient, because bit AND operations run faster than more complex operations such as distance calculations and discriminate processing. However, this method selects a large amount of candidate categories, because the distribution range, which is decided by the min-max values on an axis, grows larger than the real distribution of a category.

To summarize these conventional methods, hashing outperforms VQ clustering and the ANN methods in terms of processing speed. This is because the hashing method relies on simpler calculations. Moreover, because the AM method uses only orthogonal axes and the hash table is designed to reflect the distribution of categories, the AM method is superior to the LSH method. However, as shown above, the conventional methods can not satisfy three requirements of pre-classification simultaneously. To solve this problem, we propose a novel pre-classification method named Margin Added Hashing in this paper. In this technique, there are features of the hash table which can reduce the range of each category in feature space, and of the high-speed operation by the hash method using a bit operation. We confirm its effectiveness by an experiment. The idea of Margin Added Hashing has been published in previous work [19]. In this paper, LSH and ANN using a detailed parameter were newly added as a candidate for comparison of the proposal technique, and detailed consideration was performed.

2. Margin Added Hashing (MAH)

Considering these conventional methods and the given requirements, the Associative Matching method is superior, since it performs with high degree of accuracy while only using simple bit operations. However, for overall recognition processing it takes a relatively long amount of time, because the AM method suffers from the inability to select a small number of candidates as described above. By considering the range of each category in feature space, we propose a novel approach for pre-classification based on the AM method for an overall improvement in recognition speed while maintaining high accuracy. The MAH method uses hash tables on axes in feature space for classification as well as AM method. Differently from using a min-max range to define category boundaries on an axis in AM method, each hash table in the MAH method is calculated by projecting each category's training vector and additional margin to allow for the variation in input characters to be absorbed. Since the range of each category is reduced more than the AM method in the hash table, the MAH method can select the correct category in small set of candidates. Additionally, because the MAH method uses orthogonal axes and a bit representation in the hash table, it can execute in a very short time. Figure 6 shows an example of the hash table on one axis. In this paper, the margin is set to a constant value. The $\boldsymbol{bit1}_{\omega k}$ array for category ω on the k-th axis in the hash table dictionary is defined by the formula 1, 2, 3, and Table 1.

$$\boldsymbol{bit1}_{\omega k}^{j} = (\{b_{\omega k}^{j}\}_1, \{b_{\omega k}^{j}\}_2, \cdots, \{b_{\omega k}^{j}\}_{max\_index}) \in \boldsymbol{B}^{max\_index} \tag{1}$$

$$\{b_{\omega k}^{j}\}_i = \begin{cases} 1 & for \ \left|\lfloor (x_\omega^j \cdot \phi_k) \rfloor - i\right| \le margin \\ 0 & otherwise \end{cases} \tag{2}$$

$$\boldsymbol{bit1}_{\omega k} = \boldsymbol{bit1}_{\omega k}^{1} \vee \boldsymbol{bit1}_{\omega k}^{2} \vee \cdots \vee \boldsymbol{bit1}_{\omega k}^{N_\omega} \tag{3}$$

where,

$$\begin{aligned} &\boldsymbol{B} = \{0, 1\} \\ &\boldsymbol{u} = (u_1, u_2, \cdots, u_n) \in \boldsymbol{B}^n, \\ &\boldsymbol{v} = (v_1, v_2, \cdots, v_n) \in \boldsymbol{B}^n \end{aligned} \tag{4}$$

$$\begin{aligned} &\boldsymbol{u} \vee \boldsymbol{v} = (u_1 \vee v_1, u_2 \vee v_2, \cdots, u_n \vee v_n) \\ &\boldsymbol{u} \wedge \boldsymbol{v} = (u_1 \wedge v_1, u_2 \wedge v_2, \cdots, u_n \wedge v_n) \end{aligned} \tag{5}$$

Here, for bool vector $\boldsymbol{u}$ and $\boldsymbol{v}$ shown in formula 4 in which the element is 0 or 1, we define two operations as formula 5. That is, the operation '$\vee$' shows bit OR operation of two bool vectors. In this operation, bit OR operation is executed on each corresponding element of vectors. The operation '$\wedge$' shows bit AND operation of two bool vectors. In this operation, bit AND operation is executed on each corresponding element of vectors. We regard the bool vector as bit array.

The $\boldsymbol{bit1}_{\omega k}^{j}$ shows the bit array of the j-th training vector x_ω^j in category ω on the k-th axis. When a training vector x_ω^j in category ω is input, it is projected onto a value $\lfloor (x_\omega^j \cdot \phi_k) \rfloor$ in the k-th axis. Then, all bits in a bit array $\boldsymbol{bit1}_{\omega k}^{j}$ are set to '1' in the range from $\lfloor (x_\omega^j \cdot \phi_k) \rfloor - margin$ to $\lfloor (x_\omega^j \cdot \phi_k) \rfloor + margin$. Other bits out of the range in the bit array are set to '0'. Then, each bit array of each training vector on category ω is merged by logical OR operation to make a bit array $\boldsymbol{bit1}_{\omega k}$ on category ω in the hash table dictionary on the k-th axis. For example, the bit array $\boldsymbol{bit1}$ on category a, b, and c are described in Fig. 6. The thick line

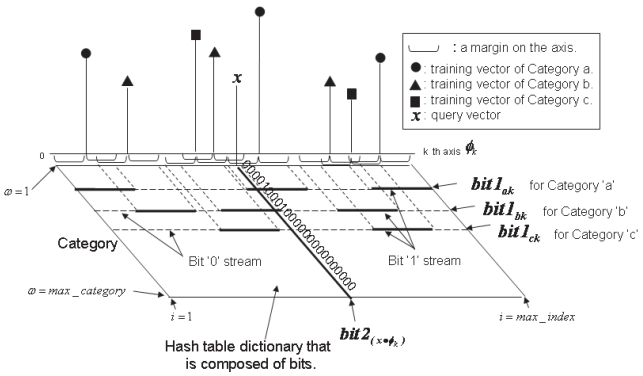

Fig. 6 Dictionary of margin added hashing method.

in a bit array shows bit '1' stream, and dotted line shows bit '0' stream. The bit '1' stream is set by each training vector on a category, so the MAH method can reduce the range of each category in feature space more than the AM method in which the range is set by only minimum and maximum value on a category.

Pre-classification is realized in the same way as AM method. That is, when a query vector x is input, it is projected onto a value $\lfloor(x \cdot \phi_k)\rfloor$ in the k-th axis. Categories including this value on the k-th axis are expressed as $\boldsymbol{bit2}_{\lfloor(x\cdot\phi_k)\rfloor}$ which is a bit array where each bit corresponds to one category in the hash table dictionary (see Fig. 6). Here, $\{b_{\lfloor(x\cdot\phi_k)\rfloor}\}_i$ shows the i-th category on the $\lfloor(x\cdot\phi_k)\rfloor$ point on which query vector is projected in the k-th axis. The final candidate category set is selected by a logical AND operation among the $\boldsymbol{bit2}_{\lfloor(x\cdot\phi_k)\rfloor}$ arrays for all axes according to the formula 6 and 7. The categories which are bit '1' in *Output_bit* array are final pre-classification output of this method.

$$\boldsymbol{bit2}_{\lfloor(x\cdot\phi_k)\rfloor} = (\{b_{\lfloor(x\cdot\phi_k)\rfloor}\}_1, \{b_{\lfloor(x\cdot\phi_k)\rfloor}\}_2, \cdots, \{b_{\lfloor(x\cdot\phi_k)\rfloor}\}_{max\_category}) \in \boldsymbol{B}^{max\_category} \quad (6)$$

$$\boldsymbol{Output\_bit} = \boldsymbol{bit2}_{\lfloor(x\cdot\phi_1)\rfloor} \wedge \boldsymbol{bit2}_{\lfloor(x\cdot\phi_2)\rfloor} \wedge \cdots \wedge \boldsymbol{bit2}_{\lfloor(x\cdot\phi_{max\_axis})\rfloor} \quad (7)$$

By using the MAH method, the pre-classification captures only the categories of the range which an input query vector falls into in feature space. Accordingly, the MAH method selects fewer candidate categories than the AM method with keeping high accuracy and the same processing speed. The dictionary for this method in 2-dimensions is shown in Fig. 7. In Fig. 7, there are boxes that shows range of each training data of category A, B, and C. First, a range on each axis is determined by integrated range of each individual range whose center is projected point of each training vector. Then, the final range is built by intersection of each range on each axis. The categories of boxes in which input query X dropped are selected as output of pre-classification. Comparing this with the AM method shown in Fig. 5, this method covers the entire distribution of each category with smaller sections of rectangles. In this method, the query vector 'X' whose category is 'C', is only inside the 'C' rectangle, so only the correct category is selected as a candidate. Even if the difference of ranges in Fig. 7 and Fig. 5 is small, when we consider the high dimension, the difference becomes large. In this case, the proposed method can select smaller number of categories than the AM method in the almost same accuracy.

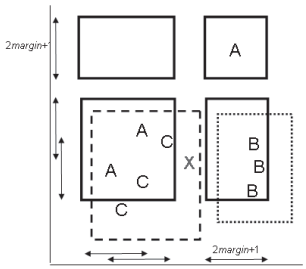

Fig. 7 Margin Added Hashing.

3. Evaluation Experiment

To confirm effectiveness of the MAH method, we tested on Japanese printed character images. A character image is first normalized and then a contour image of character image is extracted. The contour pixels are taken in four directions starting from, 0, 45, 90, and 135 degrees from the horizontal. Next, they are divided into several cells. The original feature vector is extracted by aligning the number of contour pixels taken from each direction in each cell respectively [18]. Based on the original vector, we get a 16-dimensional feature vector for pre-classification. Since one element of this vector is expressed by 1 byte, the range of the element is 1 to 256. However, the value of the element is divided by a fixed value to reduce dynamic range. We set the fixed value to 2 for each axis in our proposed method and the AM method. Then, the range on one axis which can be taken is 1 to 128. The *max_index* of Table 1 is set to 128.

The training character images are from 22 to 124 font types in one character category. We used one character image per font type for training. The total number of training character images was 445518. Since some character categories have a complex distribution in feature space, a clustering method is used for non-Kanji characters in order to make up to three clusters for each original category. We adopt the clustering algorithm in [17], as it was shown to be effective in achieving a high recognition rate in our preliminary experiment. The total of the cluster made from all the training characters was 5225. We define these clusters as categories. So, finally, we used 5225 Japanese character categories for training.

The reason why only non-Kanji character has three clusters or less is as follows. Since the degree of freedom of the stroke of non-Kanji character is larger than that of the Kanji-character, the deformation of non-Kanji character is larger in general than the Kanji character. So, the distribution of non-Kanji character in feature space becomes more complex than that of Kanji character. Actually, it turned out that the recognition accuracy of non-Kanji character was

worse than that of the Kanji character if every class has one cluster by the exploratory experiment. The recognition accuracy has improved by increasing the clusters of non-Kanji character. To make the level of distribution similar, a clustering method is used for non-Kanji characters.

We evaluated the pre-classification performance with the following three items;

1. Error rate of pre-classification
   When the selected category set has a category of an input character image, it is considered that the input character was selected correctly. The falsely selected rate ER in all the input characters is calculated as follows;
   $$ER = 1 - \frac{Ncs}{Nci} \tag{8}$$
   where Ncs is the numer of characters that are selected correctly and Nci is the numer of characters that are inputted.
2. Processing time for pre-classification and total recognition
   We evaluated the processing time taken by the pre-classification stage and total character recognition time which is defined as the pre-classification time plus the precise recognition time. In this test, precise recognition is performed by calculating the city block distance. We tested this evaluation on a PC with a Core 2 Duo E8500 CPU, 2 GB of memory, and Windows XP.
3. Ability to select a small number of categories
   When one character is inputted, some categories are selected. The average value of the number of selected categories when all the test characters are inputted is calculated. Then it is divided by the total number of categories to calculate the selection rate SR.
   $$SR = \frac{Nas}{Ncd} \tag{9}$$
   where Nas is the average number of selected categories and Ncd is the number of categories in dictionary that is fixed to 5225.

We tested the MAH method, against the AM, VQ Clustering, ANN, and LSH methods for comparison. We used a k-means method for VQ Clustering, because of its generality and speed in building clusters. As seeds for the initial condition, we used 20, 60, and 100 vectors that were determined randomly in a preliminary experiment. As for candidate selection, the top N_c clusters near the query vector were selected. According to the results of the preliminary experiment, N_c was set to 5. For the MAH method as well as the AM method, we searched a margin value for the best performance.

For ANN, we used k-nearest neighbor search in the library from [10] for evaluation with a changing number of selected candidate categories k and an eps value that represents an approximation factor. Since there are many training vectors for creating a kd-tree dictionary in the same category, the number of unique categories in the output of ANN becomes very small. For a precise recognition after pre-classification, it is necessary to select unique categories by sorting the output categories from ANN. The sorting time is included in the total ANN processing time. For LSH, we used E2LSH (Exact Euclidean LSH) [13], [23]–[25] with several parameters L, k, and w used to find the best performance. L is the number of hash tables, k is the number of the dimensions in the transferred vector, and w represents the hash size of each axes.

Apart from character images for training, we used 116528 Japanese printed character images for experiment that were extracted by manually cropping from 70 different kinds of real documents. Because processing time in precise recognition stage is proportional to the number of categories that are selected in the pre-classification stage, we defined the processing time for precise recognition as a coefficient multiplied by the number of category. The coefficient chosen was 0.067 microseconds per category, according to our findings in the preliminary experiment using simple recognition method with city-block distance for precise recognition processing stage.

The results are shown in Fig. 8. In each method, there are some parameters described in Fig. 8 to search for the best balance on the requirements of pre-classification. In the MAH method and the AM method, the parameter is margin value that is described as m in Fig. 8. In VQ clustering method, the parameter is the number of cluster n. For ANN method, the parameters are eps and k. The eps represents the approximation factor and the k represents the number of nearest neighbors to find in k-nearest neighbor search in the library from [10]. In LSH, the parameters are L, k, and w. The L is the number of hash tables, the k is the number of the dimensions in the transferred vector, and the w represents the hash size of each axes. Figure 8 shows the relationship between the total recognition processing time in microseconds and the error rate. As shown in Fig. 8, the best method is one that satisfies both small error rate and small processing time coincidentally. In this figure, we see that

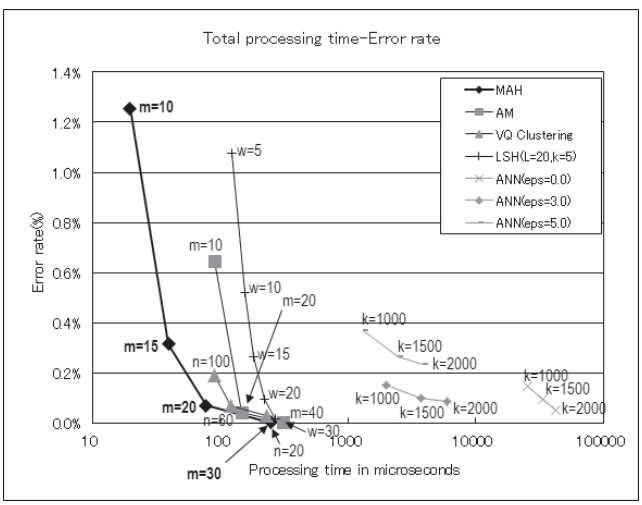

Fig. 8 Result on total processing time-Error rate.

the MAH method finds the best balance between the total recognition processing time and the error rate.

4. Discussion

In analyzing the results of Fig. 8, we examined the relationships between error rate and selection rate, and between error rate and pre-classification processing time. The results are shown in Fig. 9 and Fig. 10, respectively. We can see the trade-off between total processing time and error rate in Fig. 8. The trade-off between selection rate and the error rate for all methods is shown in Fig. 9. In Fig. 9, we can see that in terms of high accuracy, ANN is superior in selecting a small number of categories. The reason is considered as follows. That is, ANN makes cells in the kd-tree dictionary by partitioning the entire feature space without leaving any vacancy. So when an outlying vector is input as query, it must be in a large cell in the kd-tree. It is high probability that ANN selects correct category, because the distance between query vector and leaf vector is large in this case. LSH has a problem with selection accuracy. Since LSH uses random axes for hash tables, LSH tends to select a large number of candidate categories.

However, in terms of pre-classification processing time, ANN performs the worst, as shown in Fig. 10. Compared to the other methods, ANN takes almost 100 times longer to finish. Since a kd-tree dictionary is built with categories made from a large number of training vectors, the kd-tree will grow to include many layers. This is compounded by the problem that ANN also uses more complex calculations than other methods to traverse the kd-tree, so it can take a long time to reach a leaf cell with ANN. Additionally, in ANN, the number of unique categories in the selected vectors can become quite small. To address this, ANN must choose a large number of training vectors that are near the query vector so that more unique categories will be selected in order to ensure that the correct category is included in them. From these reasons it is clear that ANN's pre-classification time is much longer than the other methods. As shown in Fig. 10, LSH's pre-classification time is also quite poor. This is because LSH uses random axes for the hash tables which cause redundant processing.

Concerning the selection rate as illustrated in Fig. 9, the MAH method performs better than both the AM method and the VQ clustering method. With the VQ clustering method, there is a trade-off between the number of categories in one cluster and the error rate, so it is difficult to select a small number of categories with high accuracy. Additionally, for the AM method, when category's distribution is wide or sparse in feature space, as shown in Fig. 5, the range of this category becomes quite large which may include other categories that will also be selected as category candidates. However, Fig. 10 shows that the pre-classification processing time of these three methods is shorter than ANN and LSH. This is because the VQ clustering method uses fewer Euclid distance calculations than ANN, where as AM and the MAH method rely on bit operations. Additionally, AM and the MAH method use orthogonal axes which are simpler than the multiple random axes used by LSH.

Regarding the total character recognition performance shown in Fig. 8, we can see that the MAH method is superior. Additional confirmation from Fig. 9 and Fig. 10 shows that the MAH method is able to select a relatively small number of categories with a high degree of accuracy and faster pre-classification processing time. For example, when we allow an error rate around 0.07% as in Fig. 8, the MAH method can recognize one character less than 100 microseconds. However, when we allow similar the error rate, the other methods take over 100 microseconds.

We analyzed error cases on each pre-classification method. To compare the performance of each method, the parameters for each method were chosen so that the total processing time would approach 90 microseconds. For ANN, the parameters were chosen considering both total processing time and error rate, because ANN takes very long

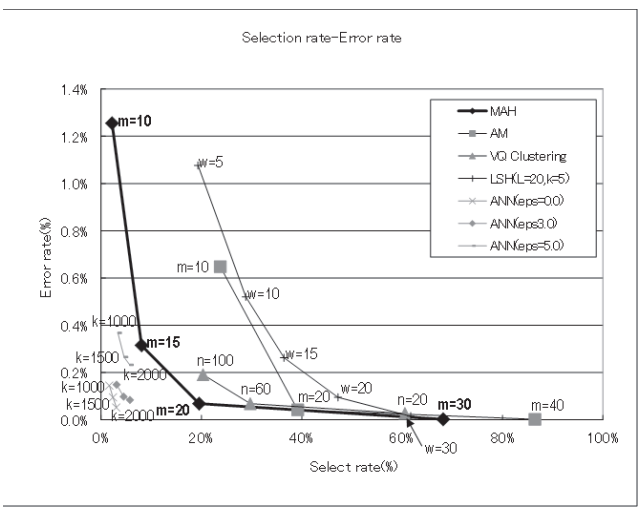

Fig. 9 Results for Selection rate-Error rate.

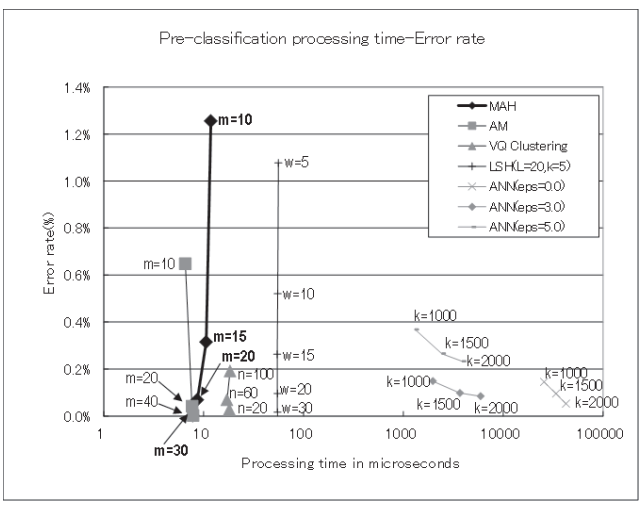

Fig. 10 Results for pre-classification processing time-Error rate.

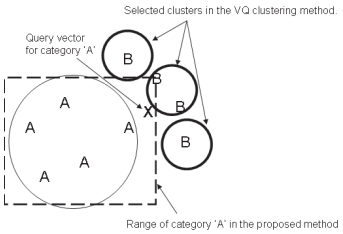

Fig. 11 An example where VQ clustering fails to identify 'X' and the MAH method succeeds.

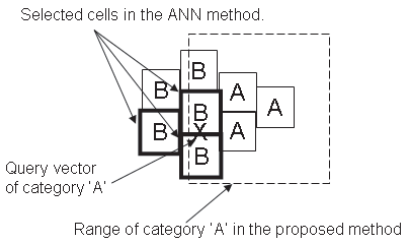

Fig. 12 An example where the ANN method fails to identify 'X' and the MAH method succeeds.

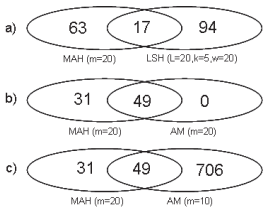

Fig. 13 The error character analysis result of LSH, AM, and the MAH method.

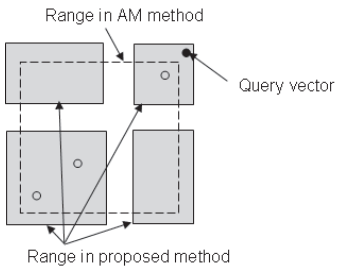

Fig. 14 An example that the MAH method succeeds and the AM method fails.

processing time. The selected parameters in each method are as follows. In AM method the value of a margin is 10, and it is 20 in the MAH method. In VQ Clustering, the number of clusters is 100. In ANN, *eps* is 3.0 and *k* is 1000. In LSH, *L* is 20, *k* is 5, and *w* is 20.

To better analyze the improvement of our proposal, we examined the cases where our method correctly recognizes the input but the comparative methods fail to do so. In VQ Clustering, clusters are selected according to the distance between the input query vector and the center of each cluster, so when there are clusters of different sizes, smaller sized clusters near the input query vector are selected. In this case, when the correct category is not included in the smaller clusters, VQ Clustering fails to select them. For example, in Fig. 11, when a query vector of category 'A' is input, clusters near the query vector are selected. However, the selected clusters only contain the 'B' category. In contrast, the correct category is selected by the MAH method because the range of the category is designed to include an additional margin.

In ANN, each cell in the kd-tree dictionary is decided by a single training vector, so the size of a cell becomes very small in the space where the density of training vectors is very high in feature space. In order to maintain a short processing time in ANN, the number of selected vectors must be reduced, however this results in there being only a few unique categories per selected vectors. This causes selection error. In Fig. 12, cells near the input query vectors all have category 'B'. However, in this case the MAH method will correctly select category 'A', because the query vector falls into the range of 'A' as defined by its training vector and margin value.

In LSH and the MAH method, the input query vector is projected onto each axis of each method to select the candidate categories. However, since LSH uses random axes of subspace of MAH, a direct comparison is impossible. The character which became an error by LSH is projected on the outside of the hash bucket at all the axes of LSH. However, sometimes that character is selected correctly by MAH. We tested LSH and MAH of 0.095% and 0.069% error rate respectively, to compare error characters. The parameters are *L*=20, *k*=5, and *w*=20 in LSH, *margin*=20 in MAH. The result of error characters analysis is shown in Fig. 13 (a). The number of error characters that is made only by MAH is 63. The number of error characters that is made only by the LSH is 94. The number of common error character is 17. In this test, we confirmed that all errors of 111 characters are caused by the projection onto the outside of the hash bucket at all the axes in LSH. Also, we confirmed that 94 characters in the 111 characters are selected correctly by MAH, because they are projected into the range in MAH dictionary.

To approach 90 microseconds in total processing time in Fig. 8, the AM method uses 10 as a margin and the MAH method uses 20 as a margin. Therefore it is possible that a query vector will exist outside of the range of the AM method but inside the range of the MAH method as shown in Fig. 14. The lower error rate of the MAH method than AM method is caused by this. However, when the same value of a margin is used, the range of AM that covers in feature space is larger than that in the MAH method. So, the error rate of AM method becomes less than that of the MAH method. We tested this by the experiment which uses 10 and 20 as a margin in AM, and the experiment of MAH using 20 as a margin. The result of the erroneous character analysis of margin 20 in both methods is shown in Fig. 13 (b). The number of error characters that is made only by AM is zero. This shows that the range of a category of AM includes the range of the category of MAH completely. For a margin 10 in AM, the result of error characters analysis is shown

in Fig. 13 (c). The number of error characters that is made only by AM become 706. That is the 706 query vectors exist outside of the range of the AM method but inside the range of the MAH.

We also examined cases where our proposed method fails and the comparison methods succeed. There were 80 different images which our method could not recognize and 16 of these images could not be identified correctly by any method. These unrecognizable characters were from the digit, alphabet, and symbol categories. The reason that they could not be recognized was because their feature vectors lie very far away from their training data distributions in feature space. This was due to dislocation and deformations such as italic type facing, noise pattern addition, and faintness in the images. As part of a further investigation regarding the remaining 64 images out of the 80 misidentified ones, we augmented our proposed method with the different pre-classification comparison methods in order to see if any further gains in recognition could be made. The results of this are shown in Table 2. In this table, ANN and VQ Clustering perform better than LSH and AM. The reason being that since the cells in the kd-tree dictionary cover feature space entirely, ANN will be able to correctly select the outlying vectors. Additionally, by selecting clusters near the input query vector, the VQ Clustering method will be able to completely cover feature space. This will increase the number of correct selections with VQ Clustering. The MAH method as well as LSH and AM use a hash table with a range on axes. So, when a query vector exists outside of a category's range in feature space, these methods will not be able to make a correct selection.

By analyzing these results, it can be seen that hashing methods have problems with outliers. That is, if a hashing method can cover the entire feature space, then the error rate will decrease. However, this comes at the cost of an increased number of selected candidates. Concerning our proposed method, if we use a larger margin in an area of feature space where a few categories exist, then the error rate will be decreased with a little number of candidate categories maintained.

We compared the dictionary sizes of these methods. The result is shown in Table 3. In VQ Clustering method, since required information is only a representative vector of each cluster, and the categories which belong to the cluster, dictionary size can be small in comparison with other methods. Since LSH, AM, and the MAH method are realized by bit expression of the category which belongs to the value on each axis as a hash table, the dictionary sizes of these methods become similar even if the number of tables and the number of axes differ. In these methods, since dictionary information is described for every category, the size of dictionary is larger than VQ clustering method. The dictionary size of ANN method becomes the greatest in these methods because having the information of all training vectors and the boundary planes between cells in feature space. With respect to a relation with error rate, since VQ clustering can keep relatively high accuracy in spite of small dictionary size, VQ Clustering is advantageous in the environment where only small memory size can be used.

Table 2 The number of correctly selected images in comparison methods.

Methods	LSH	AM	VQ Clustering	ANN
# of correct selections	20	31	42	51

Table 3 The size of dictionary of each method.

Methods	LSH	AM	MAH	VQ Clustering	ANN
Size of dictionary in Mbytes	1.4	2.7	2.7	0.05	41.1

5. Conclusion and Future Work

We have proposed a new pre-classification method named MAH based on hashing method for OCR. By reducing the range of each category of training character images in a hash table dictionary, we have shown that the proposed pre-classification method can decrease the number of selected categories more than is possible with conventional AM method, while maintaining similar classification accuracy. Additionally the total time required for recognition processing which includes pre-classification and precise recognition time has been made shorter than conventional methods. We evaluated the performance of the MAH method, and compared it to the conventional AM, VQ clustering, ANN, and LSH methods with a barrage of 116528 Japanese character images. The MAH method outperforms the conventional methods in terms of classification accuracy, the number of selected categories, and total processing time.

In the MAH method, the margin value selected for each training character on each axis is important because it affects both the processing time and classification accuracy. We plan to investigate strategies for choosing the best margin for each training character by considering a category distribution in feature space for use in real applications.

Acknowledgements

This work was inspired by a helpful discussion with Prof. Nei Kato of the Graduate School of Information Sciences at Tohoku University.

References

[1] M. Blachnik and W. Duch, "Improving accuracy of LVQ algorithm by instance weighting," Lecture Notes in Computer Science, vol.6353, pp.256–265, 2010.

[2] Y.C. Liaw, M.L. Leou, and C.M. Wu, "Fast exact k nearest neighbors search using an orthogonal search tree," Pattern Recognition, vol.43, Issue 6, pp.2351–2358, 2010.

[3] A. Ahmad and L. Dey, "A k-mean clustering algorithm for mixed numeric and categorical data," Proc. Data & Knowledge Engineering, vol.63, Issue 2, pp.503–527, 2007.

[4] M. Barrena, E. Jurado, P.M. Neila, and C. Pachon, "A flexible framework to ease nearest neighbor search in multidimensional data spaces," Proc. Data & Knowledge Engineering, vol.69, Issue 1,

pp.116–136, 2010.
[5] H. Jung, Y.D. Chung, and L. Liu, "Processing generalized k-nearest neighbor queries on a wireless broadcast stream," Information Sciences, vol.188, pp.64–79, 2012.
[6] M.J. Fonseca and J.A. Jorge, "Indexing high-dimensional data for content-based retrieval in large databases," Proc. Eighth International Conference on Database Systems for Advanced Applications (DASFAA 2003), pp.267–274, 2003.
[7] K.K. Reddy, J. Liu, and M. Shah, "Incremental action recognition using feature-tree," Proc. 12th International Conference on Computer Vision (ICCV2009), pp.1010–1017, 2009.
[8] T. Liu, A. Moore, A. Gray, K. Yang, "An Investigation of Practical Approximate Nearest Neighbor Algorithms," Proc. Neural Information Processing Systems (NIPS 2004), pp.825–832, 2004.
[9] M. Muja and D.G. Lowe, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration," Proc. International Conference on Computer Vision Theory and Applications (VISAPP'09), pp.331–340, 2009.
[10] D.M. Mount and S. Arya, "ANN: A library for approximate nearest neighbor searching," http://www.cs.umd.edu/˜mount/ANN/
[11] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," STOC, pp.604–613, 1998.
[12] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," Proc. 25th International Conference on Very Large Data Bases, pp.518–529, 1999.
[13] M. Datar, N. Immorlica, P. Indyk, and V.S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," Proc. 20th Annual Symposium on Computational Geometry, pp.253–262, 2004.
[14] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," Proc. Symposium on Foundations of Computer Science (FOCS'06), pp.1–10, 2006.
[15] M. Slaney and M. Casey, "Locality-sensitive hashing for finding nearest neighbors," IEEE Signal Processing Magazine, vol.25, Issue 2, pp.128–131, 2008.
[16] M. Toda, Y. Magome, and T. Kubota, "A high-speed rough classification method based on associative matching," Systems and Computers in Japan, vol.30, Issue 9, pp.34–43, 1999.
[17] F. Sun, S. Omachi, and H. Aso, "An algorithm for constructing a multi-template dictionary for character recognition considering distribution of feature vectors," Proc. 14th International Conference on Pattern Recognition (ICPR'98), vol.2, pp.1114–1116, 1998.
[18] N. Kato, M. Suzuki, S. Omachi, H. Aso, and Y. Nemoto, "A handwritten character recognition system using directional element feature and asymmetric mahalanobis distance," IEEE Trans. Pattern Analysis and Machine Intelligence, vol.21, no.3, pp.258–262, 1999.
[19] Y. Katsuyama, A. Minagawa, Y. Hotta, S. Omachi, and N. Kato, "A new pre-classification method based on associative matching method," Proc. DRR'2010. vol.7534SPIE, pp.1–10, 2010.
[20] M. Zhang, R. Alhajj, and J. Rokne, "Effectiveness of optimal incremental multi-step nearest neighbor search," Expert Systems with Applications, vol.37, Issue 8, pp.6018–6027, 2010.
[21] Y.L. Qiao, Z.M. Lu, J.S. Pan, and S.H. Sun, "Fast k-nearest neighbor search algorithm based on pyramid structure of wavelet transform and its application to texture classification," Proc. Digital Signal Processing, vol.20, Issue 3, pp.837–845, 2010.
[22] Y.S. Chen, Y.P. Hung, T.F. Yen, and C.S. Fuh, "Fast and versatile algorithm for nearest neighbor search based on a lower bound tree," Pattern Recognition, vol.40, Issue 2, pp.360–375, 2007.
[23] J. Toyama, M. Kudo, and H. Imai, "Probably correct k-nearest neighbor search," Pattern Recognition, vol.43, Issue 4, pp.1361–1372, 2010.
[24] J. Kybic and I. Vnucko, "Approximate all nearest neighbor search for high dimensional entropy estimation for image registration," Proc. Signal Processing, vol.92, Issue 5, pp.1302–1316, 2012.
[25] K. Tanaka and E. Kondo, "A scalable algorithm for monte carlo localization using an incremental E2LSH-database of high dimensional features," Proc. International Conference on Robotics and Automation (ICRA 2008), pp.2784–2791, 2008.

Yutaka Katsuyama received his B.E., and Master of Engineering degrees in Information Engineering from Tohoku University, Japan, in 1984, and 1986, respectively. From 1986 to 1992, he worked for the Fujitsu Ltd. Since 1992, he is working for the Fujitsu Laboratories Ltd. He received FIT2002 Award (Paper Award) in 2002.

Yoshinobu Hotta received his BE degree in mathematical engineering and information physics from the University of Tokyo in 1992. He joined Fujitsu Laboratories Ltd and is currently a senior researcher. His interest includes document image processing and pattern recognition.

Masako Omachi received her B.E., Master of Information Sciences, and Doctor of Engineering degrees from Tohoku University, Japan, in 1994, 1996, and 1999, respectively. From 1999 to 2010, she was with the Faculty of Science and Technology, Tohoku Bunka Gakuen University. Since 2010, she has been with the Advanced Course of Production System and Design Engineering, Sendai National College of Technology, where she is currently an associate professor. Her research interests include pattern recognition, character recognition, and image processing. She received the MIRU Nagao Award in 2007.

Shinichiro Omachi received his B.E., M.E., and Doctor of Engineering degrees in Information Engineering from Tohoku University, Japan, in 1988, 1990, and 1993, respectively. He worked as a research associate at the Education Center for Information Processing at Tohoku University from 1993 to 1996. Since 1996, he has been with the Graduate School of Engineering at Tohoku University, where he is currently a professor. From 2000 to 2001, he was a visiting associate professor at Brown University. His research interests include pattern recognition, computer vision, image processing, and parallel processing. He received the MIRU Nagao Award in 2007, IAPR/ICDAR Best Paper Award in 2007, Best Paper Method Award of 33rd Annual Conference of the GfKl in 2010, and the ICFHR Best Paper Award in 2010. Dr. Omachi is a member of the IEEE, the Information Processing Society of Japan, and the Japanese Society of Artificial Intelligence, among others.