WATCHING PATTERN DISTRIBUTION VIA MASSIVE CHARACTER RECOGNITION

Seiichi Uchida, Wenjie Cai, Akira Yoshida, Yaokai Feng

Kyushu University
Faculty of Information Science and Electrical Engineering
744 Motooka, Nishi-ku, Fukuoka-shi, 819-0395, Japan

ABSTRACT

The purpose of this paper is to analyze how image patterns distribute inside their feature space. For this purpose, 832,612 manually ground-truthed handwritten digit patterns are used. Use of character patterns instead of general visual object patterns is very essential to our purpose. First, since there are only 10 classes for digits, it is possible to have an enough number of patterns per class. Second, since the feature space of small binary character images is rather compact, it is easier to observe the precise pattern distribution with a fixed number of patterns. Third, the classes of character patterns can be defined far more clearly than visual objects. Through nearest neighbor analysis on 832,612 patterns, their distribution in the 32×32 binary feature space is observed quantitatively and qualitatively. For example, the visual similarity of nearest neighbors and the existence of outliers, which are surrounded by patterns from different classes, are observed.

Index Terms— character recognition, massive pattern recognition, nearest neighbor

1. INTRODUCTION

Massive pattern recognition is a recent trend where a huge number of training patterns are used. There are three important reasons for this trend. The first reason is that computer hardware becomes less expensive and more powerful. For example, we can buy 1TB HDD for less than 100USD. The second reason is that a huge number of patterns are easily available from various resources. The Internet is the most important information resource providing enormous patterns, such as digital images [1, 2, 3, 4] and videos [5]. The third reason is "crowdsourcing" for attaching a reliable class label to each sample pattern manually. For example, 11 million images of ImageNet [2] were labeled by 25,000 Amazon Mechanical Turk workers. Semi-supervised learning also helps to increase the labeled training patterns.

A fascinating example of the research project on massive pattern recognition is "80 million tiny images" by Torralba et al [1]. In their project, a huge dataset with 80 million images gathered from the Internet and resized into 32×32 pixels. In [1], the impact of the dataset size was carefully investigated through various recognition experiments. An impressive result is that high recognition accuracy was achieved not by a complex and sophisticated recognition method but just by the simplest 1-NN rule with millions of prototypes.

It is important that if we have such a huge dataset of patterns, it is a good chance to find the answer to several classical questions. For example, how are patterns distributed? Are they really Gaussian? Are there any outliers? Are there any "holes" inside the distribution? How about the overlap and proximity between classes? The huge dataset will help to have reliable answers to those questions.

It is also important that we can realize an ultimate recognition system by a huge dataset which can "fill up" the feature space completely. Although we know that its realization is impossible, even for 32×32 images, we can have an optimistic dream that we can approximate this ultimate situation by a fixed but an enormous number of patterns.

The purpose of this paper is to analyze the distribution of 832,612 labeled handwritten isolated digit patterns ("0",. . ., "9"). Those patterns were extracted and isolated manually and then stored into the dataset with their class label, i.e., ground-truth, which is also given manually. Each pattern is preprocessed to be a 32×32 binary image.

For the analysis of the distribution, we will use nearest neighbor (NN) analysis and then observe its result qualitatively and quantitatively. Specifically, we will observe (i) recognition accuracy by NN while changing the size of prototype dataset, (ii) 1-NN distances to the correct class and the closest incorrect class, and (iii) visual similarity and distance to k-NN patterns. We will not use visualization techniques based on lower-dimensionality representations, such as multidimensional scaling; this is because those techniques cannot avoid various losses during mapping into low-dimensional space. In contrast, NN realizes a simpler and lossless analysis.

Use of handwritten digit patterns instead of general visual object patterns is very essential to our purpose.

- Since there are only 10 classes for digits, it is possible to have an enough number of patterns per class for understanding the precise distribution of each class. In fact, although the total size of our dataset is 1/100

of [1], the number of patterns per class is far larger. Specifically, there are about 80,000 patterns for each of 10 classes in our case, whereas there are about 1,000 patterns on average for each of 75,000 classes in [1].

- Characters can form a compact feature space. For example, the character patterns in our dataset are 32×32 pixels and thus the dimension of their feature space is 1024, which is far smaller than that of ordinary visual object images. If each character is represented as a binary image patterns, the feature space becomes smaller. For example, the number of all possible 32×32 binary patterns is 2^{1024}, which is smaller significantly than that of 100 × 100 24-bit color images. By this compactness, it is easier to observe the precise distribution with a fixed number of patterns.
- The classes of character patterns can be defined more clearly and simply than visual objects. This is because all characters are artificial patterns designed for human communications and thus their classes must be less ambiguous. Accordingly, we can expect very accurate and firm ground-truth. Furthermore, it is not necessary for character classes to introduce hierarchical category tree like ImageNet.

In past character recognition researches, rather smaller datasets have been used compared to the above recent image datasets, such as the tiny image dataset and ImageNet. The well-known MNIST dataset includes only 70,000 handwritten digit patterns (60,000 for training and 10,000 for testing). The trial by Smith et al. [6] will be one of the largest-scale researches, where they have used about 223,000 handwritten isolated digit patterns in their research. We can differentiate this paper from [6] not only by our quadruple-sized dataset but also, more importantly, by new analysis results. The CASIA-HWDB2.0-2.2 [7] is a large handwritten Chinese character dataset which contains 3,895,135 patterns. Since the classes of Chinese characters are around 7,000, patterns for each class are reduced to about 550 on average.

2. OUR DATASET

Our handwritten digit image dataset is comprised of 832,612 patterns. Figure 1 shows several patterns from the dataset. All of the digit patterns were first isolated from their original scanned images. Then the ground-truth, i.e., correct class label ("0",. . ., "9"), was attached to each pattern carefully by manual inspections by several operators. (These operators are expert of this labeling task and they will be more reliable than Amazon Mechanical Turk workers.) The patterns will cover large variations of handwritten digit patterns because they were written by thousands of Japanese writers. Each pattern is a binary image (black and white) rescaled to 32 × 32 pixels.

Table 1. The number of patterns of each class.

class	#patterns
0	190,377
1	65,604
2	86,377
3	74,092
4	67,214
5	67,362
6	65,196
7	63,687
8	82,248
9	70,455
Total	832,612

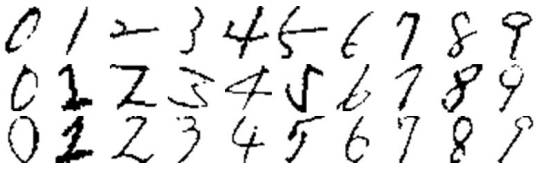

Fig. 1. Handwritten digit images from our dataset.

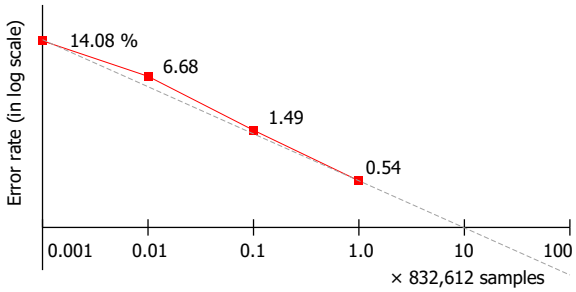

Fig. 2. Error rates under different numbers of prototypes. The plot is in log-log axis.

Table 1 shows the numbers of patterns of each digit class. Since the patterns were extracted from real form images (including money amount), the number of "0" class was larger than others. Although it was possible to use the same number of patterns for every class, we used different numbers. This is because it represents a trend of real form images; more specifically, the difference represents the difference of the prior probability of each class. In addition, we did not want to discard any patterns.

3. NEAREST NEIGHBOR ANALYSIS

As noted above, we use the simple NN analysis for observing the distribution of character patterns. Since the improvement of recognition accuracy is not our purpose, we adhere to use the original simple feature, i.e., black and white pixel values. In this case, each pattern is represented as a 1024-dimensional binary vector. In other words, each pattern corresponds to a

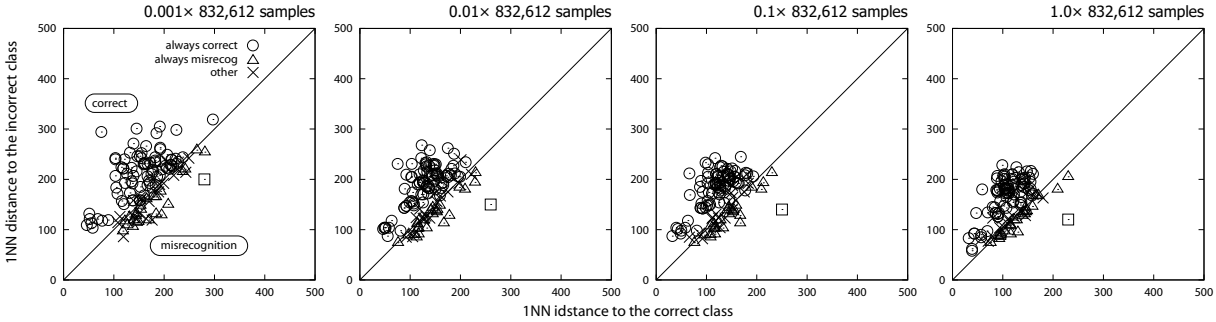

Fig. 3. 1-NN distances to the correct class and the closest incorrect class.

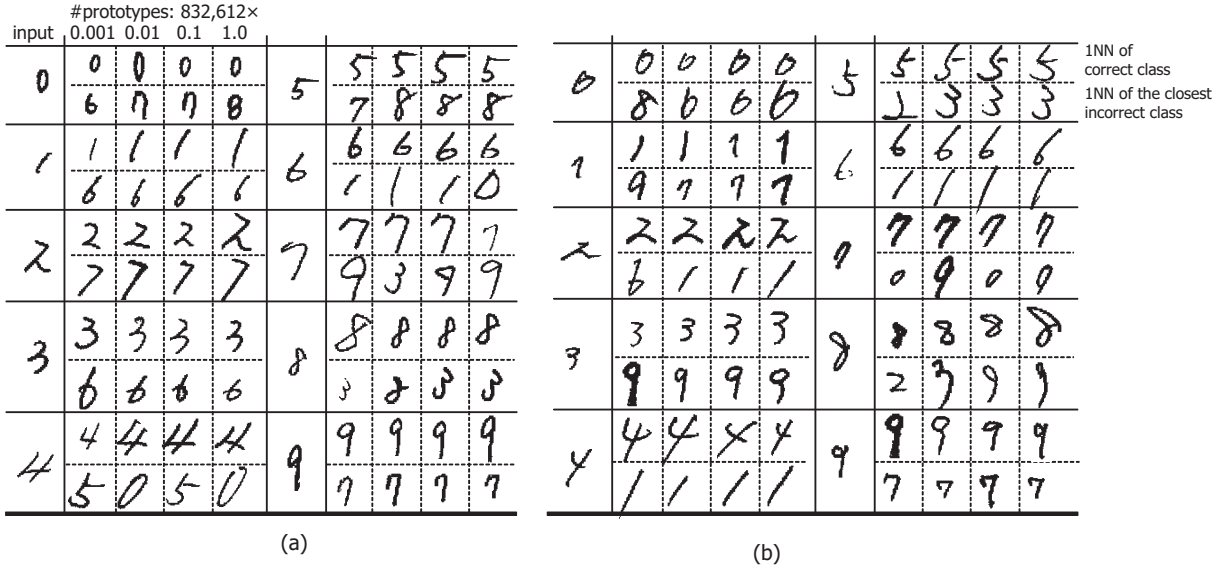

Fig. 4. Example of input patterns and their nearest prototypes under different numbers of prototypes (i.e., 0.001, 0.01, 0.1, and 1.0 × 832,612 prototypes). The input patterns in (a) were correctly recognized with all the prototypes, whereas those in (b) were still misrecognized even with all the prototypes.

corner of the 1024-dimensional hypercube.

As the distance for determining the NN pattern, we employ the Hamming distance. This is because we have to process many binary patterns and thus the Hamming distance for binary patterns are suitable for reducing computation time and memory space. Another merit of the Hamming distance is that it is easy to understand its value; if the Hamming distance between two 32 × 32 binary patterns is 100, those patterns have different black/white values at 100 pixels (about 10% among 1024 pixels) and the same black/white values at the remaining 924 pixels.

The analysis was done by the leave-one-out manner, where every pattern of 832,612 patterns was treated as an input pattern and then its NN pattern (or the k-NN patterns) were selected from the remaining 832,611 patterns. Note that in the following sections, several experiments were done while changing the dataset size. In this case, a certain number of patterns (e.g., $0.01 \times 832,612$) were randomly selected from all the patterns to form the smaller dataset.

4. OBSERVING RECOGNITION RATE

recognition accuracy by the simple 1-NN classification was evaluated. The recognition rate was evaluated by the above leave-one-out manner; that is, for considering a pattern of 832,612 as an input pattern, its NN pattern is selected from the prototype set which is comprised of all the remaining 832,611 patterns. The class of the input pattern is determined as the class of the 1-NN pattern. When a smaller dataset was used, the NN pattern was selected from the smaller dataset (i.e., the smaller prototype set). The final recognition accuracy was calculated by repeating this procedure 832,612 times.

Figure 2 plots the recognition accuracy under different numbers of prototypes. The lowest error rate of 0.54% was achieved when all the patterns were employed as prototypes. A more important fact is that, we can establish a parametric relation between the number of prototypes and the recognition accuracy. As shown in Fig. 2, it is possible to approximate the

relation roughly by

$$\log(\text{error rate}) \sim -0.4(\log(\#\text{prototypes})) + 1.27.$$

Accordingly, we can estimate that, if the number of prototypes increases 10 times, the error rate decreases to 40%. For example, if we increase the number of prototypes to 100 times, i.e., if we have 83 million of prototypes, the error rate may become about 0.11%. Note that this logarithmic property coincides with the observation by Torralba [1], where visual similarity of nearest neighbor patterns can be improved obviously by the increase their dataset size logarithmically.

5. OBSERVING 1-NN DISTANCE

Figure 3 plots the distributions of 1-NN distance to correct and incorrect classes under different numbers of prototypes. A point represents a single input pattern. For better visibility, the number of plotted points was limited. Clearly, the patterns plotted upon the diagonal line are correctly recognized because their 1-NN distance to correct class is less than that to incorrect class. Each point is depicted as a circle, triangle or cross. A circle (triangle) point corresponds to a pattern which is always recognized correctly (incorrectly) regardless of the change in the numbers of prototypes. Otherwise, the pattern is depicted as a cross mark. (The square point will be noted later.)

From Fig. 3, we can observe that the distribution of 1-NN distance moves to the lower-left corner gradually by increasing prototypes. That is, all the distances become smaller. Again, when the Hamming distance is 100, black and white are interchanged at 100 pixels among 1024 pixels, that is, about 10% of all the pixels. The patterns plotted in Fig. 3, the average distance to the 1-NN from the correct class ("correct NN") is converged around 100, and that to the 1-NN from the closest incorrect class ("incorrect NN") is converged around 150∼200, when all the prototypes were used. Consequently, their difference is around 50∼100 and thus, roughly speaking, a pattern is correctly recognized by the positive effect of only 5∼10% of all pixels. (What a narrow escape it is!)

By a further inspection of Fig. 3, it is revealed that the horizontal movement is larger than the vertical movement. This means that the distance to the correct class is more largely affected by increasing prototypes than that to the incorrect class. An exception from this trend is the pattern plotted as a square point; the distance to its incorrect NN decreased largely by the increase of prototypes. This pattern was "9", which will be shown later in Fig. 5 (b).

The patterns plotted by a cross-mark are unstable patterns because they are recognized correctly in some cases and incorrectly in the other cases. Thus, they seem to lie on a boundary of two (or more) classes. An important fact is that they are *always* on the class boundary. More specifically, their distances to the correct class and the incorrect class are always almost the same. This fact indicates that the class boundary does not change drastically by the change of the dataset size.

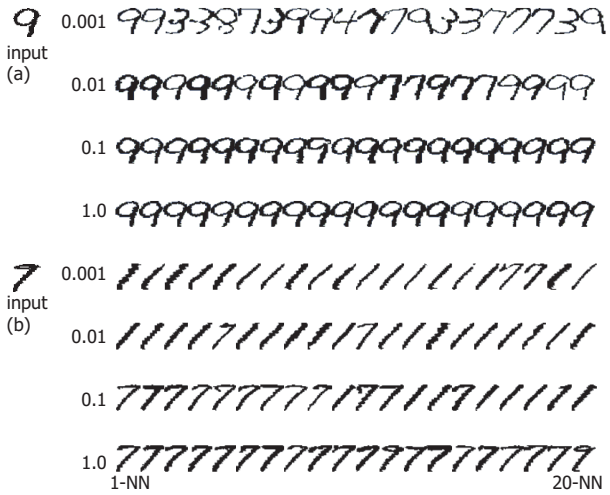

Fig. 5. 20-NN patterns for two patterns of "9" under 0.001, 0.01, 0.1, and 1.0 × 832,612 prototype.

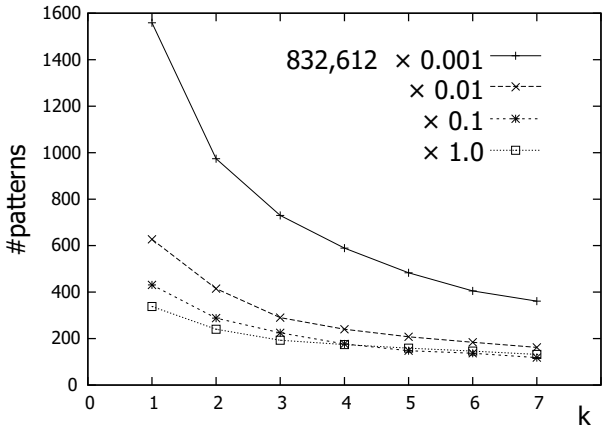

Fig. 6. The number of "outlier patterns" whose k nearest neighbors come only from incorrect classes.

Fig. 7. Examples of "outlier patterns" whose 7-nearest neighbors come only from incorrect classes.

6. OBSERVING NEAREST NEIGHBOR PATTERNS

6.1. How the nearest neighbor changes by prototype set size

Figure 4 shows the 1-NN patterns for several input patterns selected randomly. For each input pattern, its correct NN and incorrect NN under different prototype set sizes are shown.

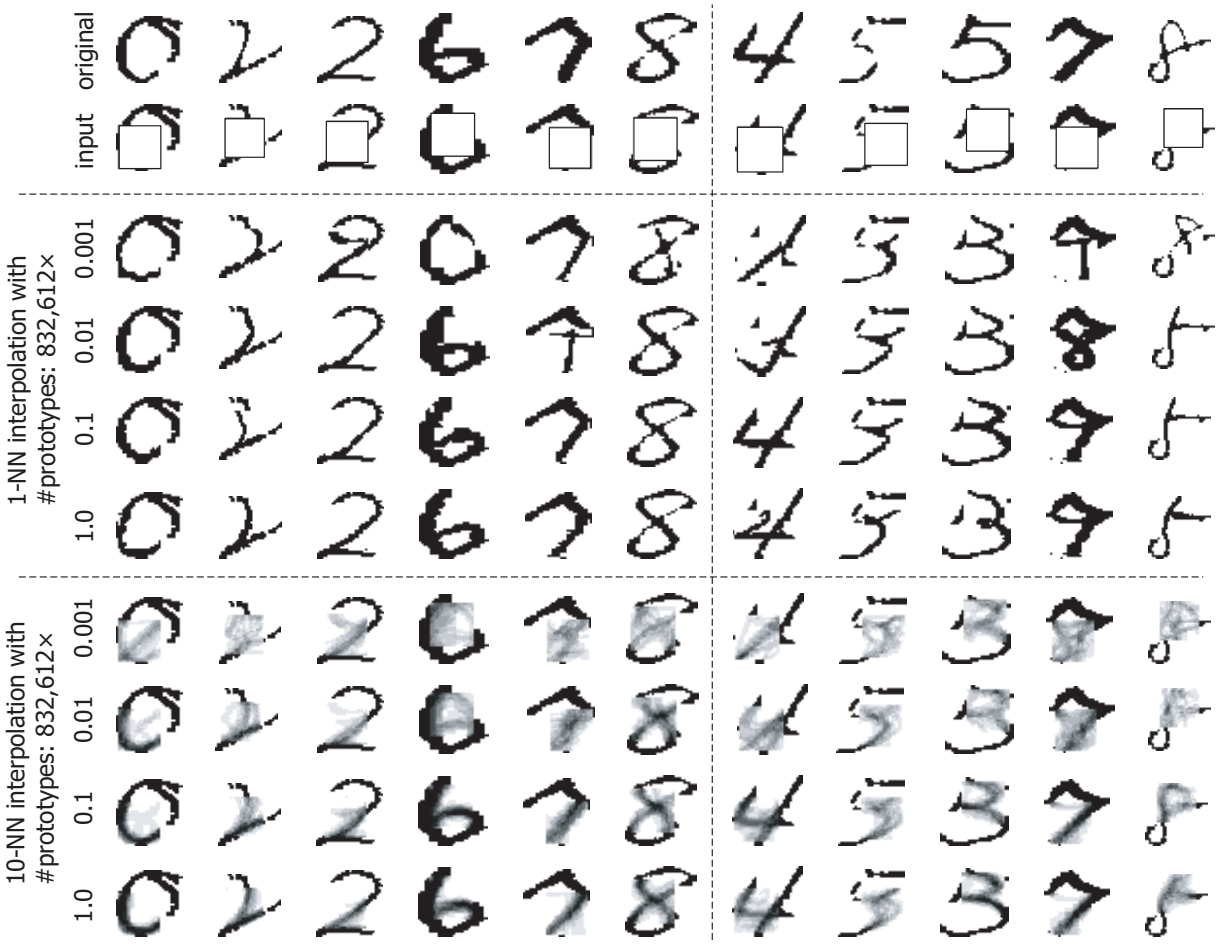

Fig. 8. Interpolation of missing part by 1-NN or 10-NNs.

The input patterns in (a) and (b) were correctly recognized and misrecognized with all the prototypes, respectively.

It is possible to observe that more prototypes often provide more visually similar correct NN prototypes to the input pattern. For example, for "3" of Fig. 4 (a), we can observe that the *visual* similarity of the correct NN prototypes was improved drastically by increasing the prototypes.

As a side-effect, the incorrect NN prototypes also become similar to the input pattern. For example, for "7" of Fig. 4 (b), we can observe that a pattern "9" which resembles the input pattern "7" became the NN pattern with all the prototypes and then lead a misrecognition.

It is interesting to note that there was sometimes a fluctuation of the NN class along with the increase of prototypes. For example, the class of the incorrect NN of "7" of Fig. 4(b) oscillated between "0" and "9." This result explicitly shows the existence of overlap in the distributions of different classes; the overlapping area is unstable and thus the NN class will easily change even by a small change of prototypes.

Figure 5 shows top 20 NNs for two patterns of "9". For (a), its NNs are occupied by the correct class "9" by the increase of prototypes. Especially, when all the patterns were used, its 20 NNs are very similar to the input pattern. This input pattern will be an "inlier", which lies inside the main cluster of "9". In contrast, for (b), which is also a pattern of "9", its NNs come mostly from incorrect classes even though all the prototypes were used. This input pattern will be an "outlier pattern", which lies in a class boundary or inside a cluster of a different class.

6.2. Outliers

Figure 6 plots the number of "outlier patterns", whose k-NNs come only from incorrect classes. One simple fact is that such outliers decrease according to the increase of prototypes. A more important fact is that the number of the outliers does not become less than 200 even with the increase of the prototypes. Figure 7 shows several example of those outliers. In addition, when all the prototypes were used, about a half of the misrecognized patterns by 1-NN ($k = 1$ in Fig. 6) are still outliers at $k = 7$. In other words, when all the prototypes were used, most misrecognized patterns are outliers which

are surrounded by other classes and thus very difficult to be recognized correctly by the NN classifier.

6.3. Interpolation of missing part by nearest neighbors

In order to show the power of a large dataset, an experiment has been conducted where a missing-part in a character pattern is interpolated by using its 1-NN pattern or 10-NN patterns. The NN patterns were selected by using the Hamming distance of the remaining-part; in other words, the missing-part was excluded from the distance evaluation for the NN selection.

Figure 8 shows the result on several patterns selected randomly. From an original pattern, an input pattern was created by removing a square region as the missing-part. The interpolation was simply done by replacing the missing-part by the corresponding part of the 1-NN pattern or the average of the corresponding parts of the 10-NN patterns.

From Fig. 8, it is observed that the 1-NN interpolation is often successful by more prototypes. These successful results indicate that we can guess the appearance of the missing-part accurately from the remaining-part if we have an enough number of instances (i.e., datasets). Especially, the first six patterns from the left are very successful and their 1-NN interpolation results with all of the 832,612 prototypes are very similar to their original patterns.

The remaining five patterns cannot have successful results by 1-NN interpolation although the patterns such as "4" and "7" could have improved results by 10-NN interpolation. The pattern "8" became an outlier by missing an important part for "8" and not only its 1-NN but also its 10-NNs contain mostly the patterns from the class "5".

7. CONCLUSION

This paper tried to analyze how enormous binary handwritten digit patterns distribute inside their feature space. For this purpose, 832,612 patterns (that is, about 80 thousands patterns for each of 10 classes on average) were collected and ground-truthed with careful manual inspection. Then, several experiments have been conducted and the following facts were revealed through the experimental results.

- By increasing prototypes 10 times, the error rate decreases to 40%. This means that a further increase of the prototypes will improve the recognition accuracy, although its effect becomes smaller.
- By increasing prototypes, the NN patterns to the input pattern become closer. This fact was confirmed by observing not only the images of NN patterns but also the distance to the NN patterns. The visual similarity to k-NN patterns was also observed.
- The existence of outliers, which were defined as patterns surrounded by patterns of different classes, was confirmed. It was difficult to change these outliers into inliers even with all of the 832,612 patterns. Considering the above first fact, an exponential increase of the prototypes is necessary for this change.
- With an enormous prototype set, it was possible to interpolate a missing-part on an character image just by finding the nearest neighbors of remaining-part. This fact indicates another possibility to make low resolution character image into higher resolution one by using a similar idea of face hallucination [8].

8. ACKNOWLEDGMENTS

The research was supported by O-RID Company of Japan. We sincerely thank O-RID Company for permitting the use of the digit image dataset for this research.

9. REFERENCES

[1] A. Torralba, R. Fergus, and W. T. Freeman, "80 Million Tiny Images: A Large Data Set for Nonparametric Object and Scene Recognition," IEEE Trans. PAMI, vol. 30, no. 11, pp. 1958-1970, 2008.

[2] J. Deng, W. Dong, R. Socher, L. -J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," Proc. CVPR, 2009.

[3] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, A. Torralba, "SUN Database: Large-scale Scene Recognition from Abbey to Zoo," Proc. CVPR, 2010.

[4] F. Schroff, A. Criminisi, and A. Zisserman, "Harvesting Image Databases from the Web," IEEE Trans. PAMI, vol. 33, no. 4, pp. 754-766, 2011.

[5] A. Karpenko and P. Aarabi, "Tiny Videos: A Large Data Set for Nonparametric Video Retrieval and Frame Classification," IEEE Trans. PAMI, vol. 33, no. 3, pp. 618-629, 2011.

[6] S. J. Smith, M. 0. Bourgoin, K. Sims, and H. L. Voorhees, "Handwritten Character Classification Using Nearest Neighbor in Large Databases," IEEE Trans. PAMI, vol. 16, no 9, pp. 915-919, 1994.

[7] `http://www.nlpr.ia.ac.cn/databases/handwriti /Home.html`

[8] C. Liu, H. -Y. Shum and W. T. Freeman, "Face Hallucination: Theory and Practice," Int. J. Comp. Vis., vol. 75, no. 1, pp. 115-134, 2007.