Pattern Recognition Using Average Patterns of Categorical \(k\)-Nearest Neighbors

Seiji Hotta, Senya Kiyasu, and Sueharu Miyahara
Department of Computer and Information Sciences, Nagasaki University, Japan
{hotta, kiyasu, miyahara}@cis.nagasaki-u.ac.jp

Abstract

The typical nonparametric method of pattern recognition \("k\)-nearest neighbor rule (kNN)\" is carried out by counting the labels of \(k\)-nearest training samples to a test sample. This method collects the \(k\)-nearest neighbors without taking into account a class, and it outputs the class of the test sample by using only the labels of neighborhoods. This paper presents a classifier that outputs the class of a test sample by measuring the distance between the test sample and the average patterns, which are calculated using the \(k\)-nearest neighbors belonging to individual classes. A kernel method can be applied to this classifier for improving recognition rates. The performance of the proposed method is verified by experiments with benchmark data sets.

1. Introduction

The nonparametric method of pattern recognition \(k\)-nearest neighbor rule (kNN) has been implemented on pattern recognition systems because of its good performance and simple algorithm. In kNN, test samples are classified by counting the labels of \(k\)-closest training samples \([1, 2]\). This approach includes the following features: 1) It has been proved that the error rate of kNN approaches the Bayes error when both the number of training samples and the value of \(k\) are infinite. 2) We can design the classifier by kNN even if training samples are few. 3) We can implement kNN when classes are overlapped with each other. 4) kNN can be implemented easily due to its simple algorithm. The main drawback to kNN is that recognition rates deteriorate when the dimensionality of feature vectors is large \([3]\). For example, Figure 1 shows the example of a test sample from the MNIST dataset and the five nearest training samples, which are evaluated using the Euclidean distance. Because the selected five training samples include the three samples of class 8, so the test sample has been misclassified to ‘8’.

For reducing this type of misclassification, it is effective to use the classification method based on comparison between the test sample and the global data distribution of individual classes such as Linear Subspace Method (LSM) \([4]\). In LSM, the data distribution of each class is represented by subspaces. The class of a test sample is determined by computing the norm of the projected test sample on the subspace. The weakness of this method is that it cannot represent the local distribution of patterns, so recognition rates decrease when the data distribution is not normal distribution.

This paper presents an alternative approach similar to kNN that classifies a test sample by measuring the distance between the test sample and the average patterns, which are calculated using the \(k\)-nearest neighbors of each class. This approach can be easily implemented due to its simple algorithm and can overcome the difficulty of kNN that recognition rates deteriorate when the dimensionality of feature vectors increases. In addition, we show how to apply kernel methods to the proposed method. The performance of the proposed method is verified by experiments with handwritten digit patterns and the benchmark data sets of binary classification problems.

2. Classification using average patterns of categorical \(k\)-nearest neighbors

In this section, we observe the nature of the \(k\)-nearest neighbors of a test sample for overcoming the difficulties found in kNN and LSM. Figure 2 illustrates the five nearest training samples of each class (only classes 3, 5 and 8 are shown). They consist of various size and line-thickness images. Note that the training samples of the classes 3 and 8 contain the patterns that are not similar to the test sample. To evaluate the rela-
2.1. Formulation

Let \( x_i^j = [x_{i1}^j, ..., x_{id}^j]^T \) \((i = 1, ..., n_j)\) be the \( d \)-dimensional training sample belonging to class \( \omega_j \), where \( n_j \) is the number of the training samples belonging to the class \( \omega_j \). When a test sample \( q = [q_1, ..., q_d]^T \) is given, the class of the test sample (denoted by \( \omega \)) is determined by

\[
    \omega = \arg \min_j \left\{ \left\| \frac{1}{k} \sum_{i \in X_j} x_i^j - q \right\|^2 \right\}, \tag{1}
\]

where \( X_j \) is the set of the \( k \)-nearest samples which belong to the class \( \omega_j \). The following relationship is established between the individual samples of \( X_j \):

\[
    \|x_i^j - q\|^2 \leq \|x_j^j - q\|^2 \leq \|x_k^j - q\|^2. \tag{2}
\]

This classification approach employs \( k \) as a parameter. In this paper, we call this method CAP (classification using Categorical Average Patterns). When \( k = 1 \), CAP coincides with the nearest neighbor rule (1-NN).

2.2. Kernel CAP

In recent years much research has been conducted on kernel methods (e.g. [5, 6]), to which CAP described above can be applied. When we apply the kernel method to CAP, the class of the test sample is determined by

\[
    \omega = \arg \min_j \left\{ \left\| \frac{1}{k} \sum_{i \in X_j} \Phi(x_i^j) - \Phi(q) \right\|^2 \right\}, \tag{3}
\]

where \( \Phi(\cdot) \) is a mapping function that maps samples from an input space to a high-dimensional space. We represent an inner product in the high-dimensional space \( \langle \Phi(x_i), \Phi(x_j) \rangle \) by an appropriate Mercer kernel \( K(x_i, x_j) \). Hence, the square of the Euclidean distance between the test sample \( q \) and the training sample \( x_i^j \) in the high-dimensional space is written as

\[
    \|\Phi(x_i^j) - \Phi(q)\|^2 = \langle \Phi(x_i^j), \Phi(x_i^j) \rangle - 2\langle \Phi(x_i^j), \Phi(q) \rangle + \langle \Phi(q), \Phi(q) \rangle
    = K(x_i^j, x_i^j) - 2K(x_i^j, q) + K(q, q). \tag{4}
\]

In the same way, the equation (3) can be expanded as

\[
    \frac{1}{k^2} \sum_{i, j \in X_j} K(x_i^j, x_j^j) - \frac{2}{k} \sum_{i \in X_j} K(x_i^j, q) + K(q, q).
\]

In this equation and the equation (4), the factor \( K(q, q) \) can be ignored, because it is the common term in all classes. In short, CAP that uses kernel methods is conducted in the following manner: First,
patterns of each class in a high-dimensional space: the distance between the test sample and the average class of the test sample is determined by measuring test error of CAP decreases while training and 10,000 test images. For feature extraction, we tested the proposed method on the handwritten digit datasets MNIST and USPS. Firstly, the proper selection of this paper, we call this method KCAP (Kernel CAP). Throughout this paper we use the Gaussian kernel with width parameter \( \alpha \). Table 1 lists the lowest error rates and average query-time (i.e., execution time per one test sample) of each method: \( k \)NN, Distance-Weighted \( k \)NN (D\( k \)NN), the basic LSM method CLAFIC [4], CAP and KCAP. For CLAFIC, the \( k \) indicates the dimensionality of subspaces. For KCAP, \( \alpha = 70 \) was used for the parameter of the Gaussian kernel. The average query-time was obtained using a Pentium 3.06 GHz with 1.0 GB of RAM memory. From the above table, we conclude that the computational costs of CAP and KCAP are high, but the error rates of them significantly are lower than those of other methods.

### 3.2. Influence of dimensionality on error rates

Next, the relationship between the dimensionality of features and error rates was examined. In this experiment, dimension reduction was applied to the training set (60000) using the Karhunen-Loève expansion technique. The variation in error rates was examined with the dimensionality ranging from 8 to 256. Figure 5 shows the results. As shown in the figure, CAP achieved lower error rates than \( k \)NN across all the range. Also note that the test error rate of \( k \)NN reaches its minimum when the number of dimensions is 32, while that of CAP decreases after it. This empirical analysis showed that CAP is effective for processing high-dimensional patterns.

### 3.3. Experimental results on USPS

Secondly, we tested the proposed method on the USPS dataset. This dataset is more difficult to recog-
nize than MNIST. The USPS dataset consists of 7,291 training and 2,007 test images. Table 2 lists the lowest error rates and average query-time of each method. For KCAP, $\alpha = 130$ was used for the parameter of the Gaussian kernel. The result showed that the proposed method outperformed all the other investigated techniques. Furthermore, the errors of KCAP were lower than those of CAP, i.e., CAP with kernel methods improved its classification abilities.

### 3.4. Experimental results on other benchmarks

Finally, we tested the proposed method on the benchmark data of binary classification problems (see \[9, 10\] for more details). Table 3 lists the lowest average test error rates and its standard deviations. Due to lack of space, we showed the results of $k$NN, CAP and KCAP only (for comparison to other methods cf. \[9, 10\]). The best values of each set are depicted in boldface type. This table showed that the results of CAP and KCAP are better than those of $k$NN in many cases. In addition, the error rates of KCAP were lower than those of CAP, i.e., the use of kernel methods helped improve the recognition performance of CAP.

<table>
<thead>
<tr>
<th>dataset</th>
<th>$k$NN</th>
<th>CAP</th>
<th>KCAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Banana</td>
<td>11.3 ± 0.6</td>
<td>11.8 ± 0.5</td>
<td>10.7 ± 0.5</td>
</tr>
<tr>
<td>B.Cancer</td>
<td>25.3 ± 4.0</td>
<td>36.5 ± 4.5</td>
<td>25.9 ± 4.4</td>
</tr>
<tr>
<td>Diabetes</td>
<td>25.1 ± 1.7</td>
<td>34.5 ± 1.8</td>
<td>23.7 ± 1.9</td>
</tr>
<tr>
<td>German</td>
<td>25.2 ± 2.3</td>
<td>24.6 ± 2.3</td>
<td>24.4 ± 2.5</td>
</tr>
<tr>
<td>Heart</td>
<td>15.7 ± 3.3</td>
<td>15.9 ± 3.4</td>
<td>16.1 ± 3.5</td>
</tr>
<tr>
<td>Image</td>
<td>3.4 ± 0.5</td>
<td>3.3 ± 0.6</td>
<td>3.3 ± 0.6</td>
</tr>
<tr>
<td>Ringnorm</td>
<td>35.0 ± 1.4</td>
<td>12.0 ± 0.8</td>
<td>1.4 ± 0.1</td>
</tr>
<tr>
<td>F.Sonar</td>
<td>34.8 ± 1.9</td>
<td>34.4 ± 1.7</td>
<td>34.4 ± 1.7</td>
</tr>
<tr>
<td>Splice</td>
<td>26.2 ± 2.1</td>
<td>43.5 ± 0.8</td>
<td>12.9 ± 0.7</td>
</tr>
<tr>
<td>Thyroid</td>
<td>4.4 ± 2.2</td>
<td>4.4 ± 2.2</td>
<td>4.2 ± 2.1</td>
</tr>
<tr>
<td>Titanic</td>
<td>22.8 ± 1.1</td>
<td>23.1 ± 1.9</td>
<td>22.8 ± 1.5</td>
</tr>
<tr>
<td>Twonorm</td>
<td>2.5 ± 0.2</td>
<td>2.4 ± 0.1</td>
<td>2.4 ± 0.1</td>
</tr>
<tr>
<td>Waveform</td>
<td>10.7 ± 1.0</td>
<td>10.2 ± 0.5</td>
<td>9.9 ± 0.6</td>
</tr>
</tbody>
</table>

### 4. Conclusions

This paper has presented an algorithm that outputs the class of a test sample by measuring the distance between the test sample and the average patterns of the $k$-nearest neighbors of each class. It was verified by the experiments using benchmark data sets that the proposed method achieved higher recognition rates than other nonparametric method such as $k$NN and LSM.

The computational cost of the proposed method is high as well as that of $k$NN. For instance, the computational costs of $k$NN and CAP are approximately $O((cnj)^2d)$ and $O(cn^2d)$ respectively, where $c$ is the number of classes. However, the proposed method can compute the distance between test samples and the average patterns of individual classes independently. Hence, it is able to reduce the number of candidate classes by performing rough classification. In addition, when training samples are added, LSM and neural networks such as Support Vector Machine require recalculating the subspace and re-learning support vectors respectively, but the proposed method only needs to add them. In other words, there is no need to reconstruct systems when training samples are added.

In short, the proposed method includes the following advantages: 1) Our methods can achieve lower error rates than other nonparametric methods such as $k$NN and LSM. 2) The proposed method can achieve low error rates even if the dimensionality of feature vectors is large. Hence, it is possible to improve recognition rates by employing kernel methods to CAP. 3) We can implement CAP and KCAP easily because of its simple algorithms. 4) There is no need to reconstruct systems when samples are added.

### Acknowledgments

This research was supported by the Telecommunications Advancement Foundation.

### References