Local Fisher Discriminant Component Hashing for Fast Nearest Neighbor Classification

Tomoyuki Shibata and Osamu Yamaguchi

Corporate Research and Development Center, Toshiba Corporation,
1, Komukai-Toshiba-cho, Saiwai-ku, Kawasaki, 212-8582, Japan
{tomoyuki1.shibata,osamu1.yamaguchi}@toshiba.co.jp

Abstract. This paper presents a novel approximate nearest neighbor classification scheme, *Local Fisher Discriminant Component Hashing* (LFDCH). Nearest neighbor (NN) classification is a popular technique in the field of pattern recognition but has poor classification speed particularly in high-dimensional space. To achieve fast NN classification, Principal Component Hashing (PCH) has been proposed, which searches the NN patterns in low-dimensional eigenspace using a hash algorithm. It is, however, difficult to achieve accuracy and computational efficiency simultaneously because the eigenspace is not necessarily the optimal subspace for classification. Our scheme, LFDCH, introduces Local Fisher Discriminant Analysis (LFDA) for constructing a discriminative subspace for achieving both accuracy and computational efficiency in NN classification. Through experiments, we confirmed that LFDCH achieved faster and more accurate classification than classification methods using PCH or ordinary NN.

Keywords: approximate nearest neighbor classification, high-dimensional space, hash, dimensionality reduction.

1 Introduction

Nearest neighbor (NN) classification [1] is a popular technique in the field of pattern recognition. Given a query pattern, a NN classifier classifies the pattern by simply finding the nearest pattern among training ones. However, the NN classification is not regarded as a practical, because of its inefficient memory use and poor classification speed.

To accelerate classification speed, fast NN search algorithms have been proposed. These search algorithms introduce the following two ideas:

– Reducing the number of comparisons: The number of distance calculations between a query and each training patterns is minimized by using the triangle inequality. [10][11][12][13]
– Aborting the number of distance calculation: Distance calculation are not necessary for those training patterns that can not possibly be the nearest neighbor of the query pattern. [2]

However, the previous methods have the limitation that search cost does not reduce in high-dimensional space (more than a few tens of dimensions). In this case, NN search is no better than brute force. Thus, recently, approximate NN search methods have been gathering interest. These approximate methods do not guarantee to find the nearest pattern among training patterns.

Principal Component Hashing (PCH) [3] is one of the approximate NN search methods. PCH projects training patterns into an eigenspace obtained using principal component analysis (PCA) [4]. The eigenspace is generated by the maximizing the variance of training patterns in the lower-dimensional space. We refer to this eigenspace as PCA space. In the experiments of [3], PCH is a faster and more accurate search method than normal fast NN search methods and previous approximate methods.

However, PCH has the following barriers to achieving faster classification speed while keeping accuracy of classification.

(1) PCA space is not necessarily the optimal subspace for classification.
(2) PCH has a significant trade-off relationship between classification speed and classification accuracy.

To solve these problems, we propose a novel approximate nearest neighbor classification scheme, *Local Fisher Discriminant Component Hashing* (LFDCH). For problem (1), above, we introduce Local Fisher Discriminant Analysis (LFDA) [7], which is a supervised dimensionality reduction method, for obtaining a discriminative subspace by maximizing between-class and minimizing the within-class variance. We refer to this discriminative subspace as LFDA space. LFDCH in low dimensional LFDA space is a more accurate NN classifier than PCH in low dimensional PCA space. For problem (2), above, we recognize that the ultimate goal of NN classification is to classify the class label of the query pattern, rather than simply to find the NN pattern. Thus, LFDCH can abort the classification process when all training patterns have the same class label. LFDA space helps to increase the probability of this event compared with PCA space. As a result, classification speed on LFDCH is further accelerated, compared with the classification method based on PCH.

The remainder of this paper is organized as follows: Section 2 and Section 3 explain PCH and LFDA respectively. We propose LFDCH in Section 4 and show experimental results in Section 5. Finally Section 6 presents conclusions.

2 Principal Component Hashing

Principal Component Hashing (PCH) [3] is a search method using the hash function in one of the approximate NN search algorithms. Other approximate NN search algorithms (e.g. Locality Sensitive Hashing [5][6]) have the following drawbacks.

- Hash may fail to find the corresponding bucket when the query lies in a low-density area of training patterns. The PCA space is divided into equal-sized buckets by hashing, for each axis.

– The number of NN candidates will not decrease drastically when the query is given at a high-density area of training patterns.

Those problems are caused by the hash function which produces a disjoint decomposition of the space by equal-sized buckets.

For solving these problems, PCH searches by dividing the space into finite buckets that each contain the same expected number of training patterns. We describe below how to decompose the space into buckets in this way.

2.1 Bucket Decomposition

Previous approximate NN search methods using hash functions, such as Locality Sensitive Hashing, divided PCA space into equal intervals, with each interval referred to as a bucket, and made training patterns within buckets to hash to the same index. These methods construct the hash function without any relation to training pattern distribution, or do not allocate buckets to cover the whole pattern space. As a result, if a query falls into a high-density area of training patterns, NN search becomes less efficient. Conversely, if a query falls into an area with no training patterns, there may be no training patterns that hash to the same index as the query hash index.

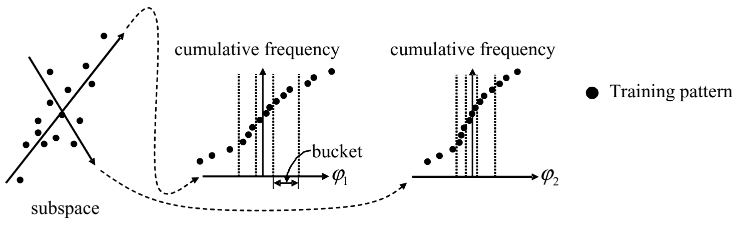

Fig. 1. Each axis is divided depending on the cumulative frequency of training patterns in PCA space. The divided interval is called a bucket, and training patterns in the same bucket hash to the same index.

The hash function constructed for PCH depends on the training pattern distribution, and each bucket contains a fixed number of training patterns as shown in Fig. 1.

Let $x_i \in \Re^d$ $(i = 1, 2, \ldots, n)$ be d-dimensional training patterns and $\varphi^T(x - \bar{x})$ be the training patterns projected into the subspace, where matrix φ is made up of the A components of the subspace, forming an orthonormal basis. Let ^{T} denotes the transpose of a matrix or vector, and $\bar{x}$ is an average vector of all training patterns. The bucket is decomposed to contain a fixed number of training patterns on φ_i. This process, which is called bucket decomposition, represents threshold processing on φ_i, and the bucket that contained the query can be determined by binary search.

2.2 Query Search

The PCH's search algorithm steps are shown in the following. **Step1.** and **Step2.** are preprocessing.

Step1. Obtain the PCA space for training patterns using PCA.
Step2. Let A be the number of dimensions of the PCA space, each bucket is decomposed, and training patterns are ascribed to the corresponding buckets.
Step3. On each φ obtained in preprocessing, the bucket that contained query q is determined.
Step4. For each training pattern, let it's *overlapped number* be the number of occasions when the patterns and the query pattern fall into the same backet. Let candidate NN patterns $C(q)$ be the training patterns that correspond to the highest $b\%$ of the calculated overlapped numbers, where $b\%$ is called *cutoff rate.*
Step5. After the approximate NN pattern of the query is determined by distance calculation from among $C(q)$, the process finishes.

Because complex processing to refer to neighboring buckets was necessary for **Step4.** in the original PCH, **Step4.** was simplified as shown above.

2.3 Refinement by Aborting the Distance Calculation

Since the approximate NN pattern of a query is determined from among $C(q)$ as shown above, the training pattern that has the most overlapped number is x_{max}, and the distance from query q to x_{max} is $D(q, x_{max})$. This distance is called provisional distance z, which means that

$$z = D(q, x_{max}), \quad where\ x_{max} \in C(q) \tag{1}$$

The distance calculation is able to be aborted using this provisional distance z. In the following discussions, this paper uses L_p distance $D_p(x_1, x_2)$. L_p distance is given by

$$D_p(x_1, x_2) = \sqrt[p]{\sum_{i=1}^{M} (x_{1i} - x_{2i})^p} \tag{2}$$

in M dimensional space.

To be based on Parseval's theorem, L_p distance can be calculated between projected vector and discretional orthonormal basis, and let L_p distance in subspace be

$$D_p(x_1, x_2) = \sqrt[p]{\sum_{i=1}^{M} \left|\varphi_i^T x_{1i} - \varphi_i^T x_{2i}\right|^p}. \tag{3}$$

Consequently, candidate x that satisfies the following conditions is not able to be NN pattern.

$$\sum_{i=1}^{m} \left|\varphi_i^T q - \varphi_i^T qx\right|^p > z^p, \quad where\ m \leq M. \tag{4}$$

Based on this property, **Step5.** (see 2.2) is modified using refinement by aborting the distance calculation by the following steps. Let A be the number of dimensions in PCA space, N be the number of NN pattern candidates, provisional NN pattern be x_t, and provisional distance be z^p.

Step5-1. Let x_{max} be the pattern that has maximum overlapped number among $C(q)$, and calculate the provisional distance z^p.
Step5-2. $n := 1$
Step5-3. Select x_n among $C(q)$ without x_t. If $n = N$, NN search is finished as NN pattern is x_t.
Step5-4. $m := 1$
Step5-5. If x_n satisfies Eqs.4, x_n is deleted, $n = n + 1$, and go to **Step5-3.**. If $m = A$, go to **Step5-7.**.
Step5-6. $m = m + 1$ and go to **Step5-5.**.
Step5-7. Calculate $D_p(q, x_n)$. If $D_p^p(q, x_n) < z^p$, $x_t = x_n$ and $z^p = D_p^p(q, x_n)$. Go to **Step5-3.**.

These steps are processed with φ vectors in decreasing order of training pattern variance. Therefore, aborting the number of distance calculation becomes efficient because more training patterns satisfy Eqn.(4). This ensured that the true NN pattern is not rejected by above steps.

3 Local Fisher Discriminant Analysis

Fisher Discriminant Analysis (FDA) [8] is a popular method for linear supervised dimensionality reduction. FDA seeks an embedding transformation such that the between-class scatter is maximized and the within-class scatter is minimized.

Let W and B be the within-class scatter matrix and the between-class scatter matrix and $\{\varphi_k\}_{k=1}^d$ be the generalized eigenvectors associated with the generalized eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$ of the following generalized eigenvalue problem:

$$B\varphi = \lambda W\varphi. \tag{5}$$

The FDA transformation matrix T_{FDA} is given by

$$T_{FDA} = (\varphi_1 \mid \varphi_2 \mid \cdots \mid \varphi_r), \tag{6}$$

where r is rank(T_{FDA}). The between-class scatter matrix B has at most rank $c-1$. This implies that the multiplicity of $lambda = 0$ is at least $d-c+1$. Therefore, FDA can find at most $c - 1$ meaningful features; the remaining features found by FDA are arbitrary. This is an essential limitation of FDA for dimensionality reduction and is very restrictive in practice. Additionally, FDA can perform poorly if patterns in a class form several separate clusters (i.e., multimodal).

To overcome these problems, Local Fisher Discriminant Analysis (LFDA) evaluates the levels of the between-class scatter and the within-class scatter in a local manner. Namely, according to the affinity matrix defined by similarity from pattern pairs, LFDA weights the values for the pattern pairs in the same class. This means that far apart pattern pairs in the same class have less influence on the within-class scatter matrix W and the between-class scatter matrix B. This allows LFDA to attain between-class separation and within-class local structure preservation simultaneously. Additionally, because the between-class scatter matrix B in LFDA is always full rank for various data sets, thanks to the affinity matrix, LFDA can find at most d meaningful features.

4 Local Fisher Discriminant Component Hashing

For faster and more accurate classification, Local Fisher Discriminant Component Hashing (LFDCH) introduces the following two ideas into PCH.

- LFDA obtains the subspace that maximizes between-class and minimizes the within-class variance.
- NN classification does not necessarily need to find the NN pattern, but only needs to classify its class label.

LFDCH projects training patterns into LFDA space and searches by hash function, which produces a disjoint decomposition of the space by finite buckets that each contain the same numbers of training patterns. LFDCH aborts the process when all training patterns have the same class label.

In contrast to PCH aimed at NN search, LFDCH is aimed at NN classification and does not need to store the vector of training patterns in the original space. LFDCH only stores the projected vectors to memory. As a result, LFDCH can use memory more efficiently than PCH can.

The LFDCH's classification algorithm steps are shown in the following. **Step1.**, **Step2.**, and **Step3.** are preprocessing.

Step1. Obtain the LFDA space for training patterns using LFDA.
Step2. Let A be the number of dimensions of the LFDA space. All training patterns are projected to the LFDA space. LFDCH only stores training patterns after dimensionality reduction.
Step3. For A components φ, each bucket is decomposed, and training patterns are ascribed to corresponding bucket as well as PCH.
Step4. On each φ obtained in preprocessing, the bucket that contained query q is determined.
Step5. Let the candidate NN patterns $C(q)$ be the patterns corresponding to the highest b% of overlapped numbers.

Step6. Refine $C(q)$ by aborting the distance calculation where possible, the same as with PCH. LFDCH can also abort the process and classify *query* when all $C(q)$ have the same class label.

5 Experimental Results

In the experiment, we compared the following methods with Correct Match Rate (CMR) and in terms of classification time:

LFDCH: Proposed algorithm.
PCH: Query is classified into the same class with NN pattern searched by PCH.
NN: Query is classified into the same class with NN pattern searched by naive NN search engine.

The classification time is the elapsed time taken to classify one query pattern. All experimental results described here are obtained on Intel Xeon PC, 3.0 GHz CPU, 16 GB memory using Microsoft Visual Studio 2005. We use the following settings as default for LFDCH and PCH:

Number of PCA and LFDA special dimensions: 20
Number of buckets per axis: 500
cutoff rate: 20%

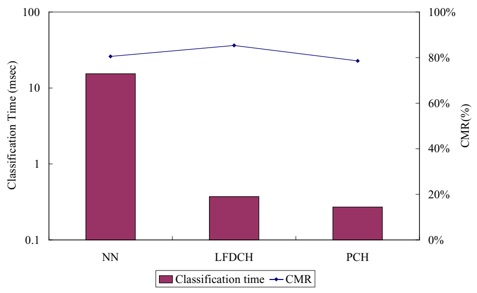

Fig. 2. This figure shows the classification time per query pattern and the CMR. The left vertical axis shows a logarithmic scale of classification time and the right vertical axis shows the CMR. The bar graph shows the classification time, and the line graph shows the CMR, for each method.

5.1 Gender Classification Using Face Image

We use multiple face image data sets and illustrate how LFDCH classifies a two-class classification problem. We classify gender using a normalizetion method that uses a generic 3D face model [9]. We describe the in-depth data as follows.

Distance: Euclidean Distance
Number of classes: 2 (male, female)

Number of training patterns: 18239 (11453 male, 6786 female)
Number of query patterns: 21573 (15991 male, 5582 female)
Dimension of patterns: 1024

Fig. 2 shows the classification time per query pattern and the corresponding CMR. LFDCH and PCH are much faster than NN. LFDCH is about 0.1 msec slower than PCH, but the CMR of LFDCH is about 7 points higher than that of PCH. Additionally, the CMR of LFDCH is about 5 points higher than that of NN. Shown on the left in fig. 3 are the classification times per query pattern and the CMR for LFDCH and PCH, changing the number of both LFDA and PCA spatial dimensions from 1 to 100. The CMR of LFDCH is always higher than that of PCH in each dimension. CMR of PCH becomes higher by adding the PCA spatial dimensions, but that of LFDCH becomes maximum in low LFDA spatial dimensions. As a result, LFDCH can achieve faster more and accurate classification than PCH. For example, LFDCH needs only 2 LFDA spatial dimensions to achieve the same CMR as NN, but PCH needs 100 PCA spatial dimensions. In this case, LFDCH is about 100 times faster than PCH, and 50 times smaller than PCH in terms of memory use. Shown on the right in fig. 3 are the classification times per query pattern and the CMR for LFDCH and PCH, changing the number of bucket decompositions from 50 to 1000. The CMR of LFDCH is more independent fo the number of bucket decompositions than PCH. Because both methods become fast by increasing the number of bucket decompositions, LFDCH speed can be further imprved while maintaing a good CMR.

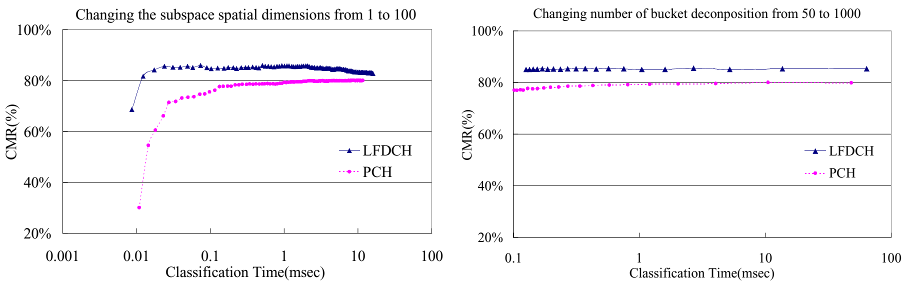

Fig. 3. The left figure shows the classification time per query pattern and the CMR on LFDCH and PCH, changing both LFDA and PCA spatial dimensions from 1 to 100. The right figure shows the classification time per query pattern and the CMR of LFDCH and PCH, changing the number of bucket decompositions from 50 to 1000. In both figures, the horizontal axis shows logarithmic scale of classification time and the vertical axis shows the CMR.

5.2 Handwriting Character Recognition

We use orientation component density of 256 dimensional features for uniquely collected handwritten characters and illustrate how LFDCH classifies the multi-class classification problem. We describe the in-depth data as follows.

Distance: Euclidean Distance
Number of classes: 62 (case-sensitive alphabet (52), digit 0 to 9 (10)
Number of training patterns: 20150 (325 patterns per character)
Number of query patterns: 20150 (325 patterns per character)

Shown on the left in fig. 5 are the classification times per query pattern and the CMR for LFDCH and PCH, changing both LFDA and PCA spatial dimensions from 1 to 100.

Fig. 4 shows the classification time per query pattern and the CMR. LFDCH and PCH are much faster than NN. LFDCH is about 0.1 msec slower than PCH, but the CMR of LFDCH is about 7 points higher than that of PCH. Additionally, the CMR of LFDCH is about 1 point higher than that of NN.

The CMR of LFDCH is always higher than that of PCH in every case. The CMR of PCH becomes higher by increasing the number of spatial dimensions,

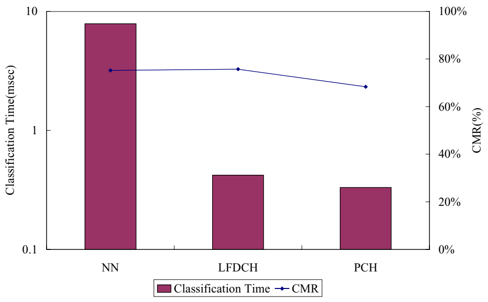

Fig. 4. This figure shows the classification time per query pattern and the CMR. The left vertical axis shows a logarithmic scale of classification time and the right vertical axis shows the CMR. The bar graph shows the classification time, and the line graph shows the CMR, for each method.

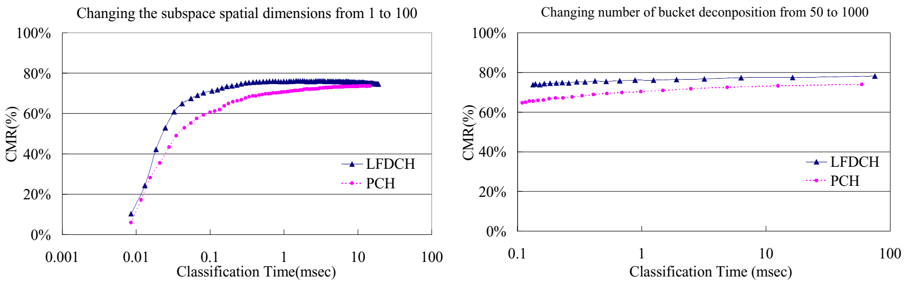

Fig. 5. The left figure shows the classification time per query pattern and the CMR on LFDCH and PCH, changing both LFDA and PCA spatial dimensions from 1 to 100. The right figure shows the classification time per query pattern and the CMR on LFDCH and PCH, changing the number of bucket decomposition from 50 to 1000. In both figures, the horizontal axis shows a logarithmic scale of classification time and the vertical axis shows the CMR.

but that of LFDCH exhibits a maximum at low LFDA spatial dimensions. As a result, LFDCH can achieve faster and more accurate classification than PCH which is the same as the situation described in Subsection 5.1. For example, LFDCH needs only 34 LFDA spatial dimensions to achieve the same CMR as NN, but PCH needs more than 100 PCA spatial dimensions. In 100 PCA spatial dimensions case, LFDCH is about 10 times faster than PCH, and 3 times smaller than PCH in terms of memory use. Shown on the right in fig. 5 are the classification times per query pattern and the CMR for LFDCH and PCH, changing the number of bucket decompositions from 50 to 1000. The CMR of LFDCH is more independent of the number of bucket decompositions than PCH. Because both methods become fast by increasing the number of bucket decompositions, LFDCH speed can be further improved while maintaining a good CMR.

6 Conclusions

In this paper, we proposed a fast and accurate NN classifier, LFDCH. The experimental results have confirmed the following properties.

- LFDCH is faster than NN search and classification methods using PCH and ordinary NN in high-dimensional space.
- The CMR of LFDCH exhibits a maximum in low-dimensional LFDA space.
- LFDCH can maintain a good CMR if the number of bucket decompositions is large.

LFDCH was able to achieve a faster classification speed than the classification method based on PCH while maintaining accuracy of classification. Future work is to automate the optimization of the number of dimensions of the LFDA space.

Acknowledgment

I am deeply grateful to Prof. Toshikazu Wada whose enormous support and insightful comments about PCH were invaluable.

References

1. Cover, B.K.B.T.M., Hart, P.E.: Nearest neighbor pattern classification. IEEE Transactions on Information Theory IT-13(1), 21–27 (1967)
2. Arya, S., Mount, D.M., Netanyahu, N.S., Silverman, R., Wu, A.Y.: An optimal algorithm for approximate nearest neighbor searching. Journal of the ACM 45, 891–923 (1998)
3. Matsushita, Y., Wada, T.: Principal Component Hashing. Meeting on Image Recognition and Understanding (MIRU), pp. 127–134 (July 2007) (in Japanese)
4. Hotelling, H.: Analysis of a complex of statistical variables into principal components. Journal of the American Statistical Association 24, 441 (1933)
5. Indyk, P., Motwani, R.: Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In: Proceedings of the 30th ACM Symposium on Theory of Computing (STOCf 1998), pp. 604–613 (May 1998)

6. Datar, M., Indyk, P., Immorlica, N., Mirrokni, V.: Locality-Sensitive Hashing Scheme Based on p-StableDistributions. In: Proceedings of the 20th Annual Symposium on Computational Geometry (SCG 2004) (June 2004)
7. Sugiyama, M.: Dimensionality reduction of multimodal labeled data by local fisher discriminant analysis. JMLR 8, 1027–1061 (2007)
8. Fisher, R.A.: The Use of Multiple Measurements in Taxonomic Problems. Annals of Eugenics 7, Part II, 179–188 (1936)
9. Kozakaya, T., Yamaguchi, O.: Face Recognition by Projection-based 3D Normalization and Shading Subspace Orthogonalization. FGR, pp. 163–168 (2006)
10. Vidal, R.: An algorithm for finding nearest neighbor in (approximately) constant average time. Pattern Recognition Letters 4, 145–158 (1986)
11. Mico, L., Oncina, J., Vidal, E.: A new version of the nearest-neighbor approximating and eliminating search algorithm (AESA) with linear preprocessing time and memory requirements. Pattern Recognition Letters 15, 9–17 (1994)
12. Brin, S.: Near neighbor search in large metric spaces. In: Proc. of 21st Conf. on very large database (VLDB), Zurich, Switzerland, pp. 574–584 (1995)
13. Yianilos, P.Y.: Data structures and algorithms for nearest neighbor search in general metric spaces. In: Proc. of the Fourth Annual ACM-SIAM Symp. on Discrete Algorithms, Austin, TX, pp. 311–321 (1993)