

Learning Vector Quantization with Local Subspace Classifier

Seiji Hotta

Tokyo University of Agriculture and Technology
s-hotta@cc.tuat.ac.jp

Abstract

In this paper, *generalized learning vector quantization (GLVQ) with local subspace classifier (LSC)* is proposed for achieving high accuracy with a small memory requirement. In a training phase, the k -closest prototypes to an input training sample are moved by the same update rule of GLVQ for reducing the number of misclassification on training samples. In a test phase, a test sample is classified by LSC with trained prototypes. Experimental results on a handwritten digit show that the proposed learning rule outperforms other classifiers such as the original GLVQ algorithm.

1. Introduction

In pattern recognition, neighbor-based classifiers such as the *nearest neighbor rule* (NN rule) are adopted to real life problems. Particularly, several classifiers that use the k -closest training samples to a test sample were proposed recently [1, 2, 3]. For example, *local mean-based classifier* (LMC) [1] uses a centroid of k -neighbor training samples as the most similar prototype from a class, and a test sample is classified into the class to which the nearest centroid belongs. LMC is robust against curse of dimensionality and can give smooth nonlinear decision boundaries with small k .

The idea of making use of k -neighbor training samples of each class first appeared in *local subspace classifier* (LSC) [4, 5]. In LSC, first k -neighbor training samples to a test sample are selected from each class by Euclidean distance. Next, distances from a test sample to affine subspaces spanned by the selected training samples are measured, and then a test sample is classified into the class to which the nearest affine subspace belongs. By this scheme, LSC can give nonlinear decision boundaries and expand a representation capacity of available training samples, thus LSC tends to outperform the NN rule and LMC in many cases. However, LSC requires a large amount of memory because it is a

kind of neighbor-based classifiers.

On the other hand, *generalized learning vector quantization* (GLVQ) [6] that includes the original LVQ [7] algorithm was used for keeping accuracy of the NN rule with a small memory requirement. In GLVQ, prototypes called reference (codebook) vectors are updated iteratively for reducing misclassification on training samples. After training, a test sample is classified by the NN rule with trained prototypes. When GLVQ is applied to class distribution that has complex decision boundaries, a large number of prototypes are required. However, too many prototypes might result in poor performance [8], so it is difficult to improve accuracy by only increasing the number of prototypes.

In this paper, a learning rule for LSC called *learning LSC* (LLSC) is proposed for keeping accuracy of LSC with a small number of prototypes. In a training phase, all k -closest prototypes to an input training sample are moved by the same update rule of GLVQ for reducing the number of misclassification on training samples. In a test phase, a test sample is classified by LSC with trained prototypes. Experimental results on the USPS handwritten digit dataset [9] show that LLSC outperforms other classifiers such as GLVQ.

2. LSC

Let us begin with a brief review of LSC. In LSC, the k -closest training samples to a test sample are selected from each class, and a $(k - 1)$ -dimensional affine subspace is spanned by them in each class. In a test phase, a test sample is classified based on the shortest distance from a test sample to these affine subspaces.

Now a procedural formulation of LSC is shown. Let $\mathbf{x}_i^j = (x_{i1}^j \cdots x_{id}^j)^\top \in \mathbb{R}^d$ ($i = 1, \dots, n_j$) be the i th training sample belonging to class j ($j = 1, \dots, C$), where C and n_j are the numbers of classes and training samples belonging to class j , respectively.

Given a test sample $\mathbf{q} = (q_1 \cdots q_d)^\top \in \mathbb{R}^d$, first the k -closest training samples to it are selected from each class using Euclidean distance, and denote a set

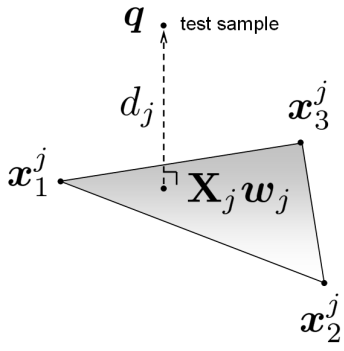


Figure 1. LSC concept in class j . Each of the corners of the triangle represents the pure three-closest training samples to a test sample \mathbf{q} , whereas a gray hyperplane in between represents linear combinations of them. These intermediate samples are as good representative samples of the class as pure training ones.

of selected k -neighbor training samples from class j as a matrix $\mathbf{X}_j = (\mathbf{x}_1^j | \mathbf{x}_2^j | \dots | \mathbf{x}_k^j) \in \mathbb{R}^{d \times k}$. Next, optimal weights for the k -neighbors from class j ($\mathbf{w}_j = (w_1^j \dots w_k^j)^\top \in \mathbb{R}^k$) for reconstructing a test sample from its neighbors by solving the following optimization problem [5]:

$$\begin{aligned} \min_{\mathbf{w}_j} \quad & \|\mathbf{q} - \mathbf{X}_j \mathbf{w}_j\|^2 \\ \text{s.t.} \quad & \mathbf{1}_k^\top \mathbf{w}_j = 1, \end{aligned} \quad (1)$$

where $\mathbf{1}_k = (1 \dots 1)^\top \in \mathbb{R}^k$ is a vector of which all elements are 1. By using Lagrange multipliers, the optimal weights subject to sum-to-one are given as follows:

$$\mathbf{w}_j = \frac{\mathbf{C}_j^{-1} \mathbf{1}_k}{\mathbf{1}_k^\top \mathbf{C}_j^{-1} \mathbf{1}_k}, \quad (2)$$

where $\mathbf{C}_j = (\mathbf{Q} - \mathbf{X}_j)^\top (\mathbf{Q} - \mathbf{X}_j) \in \mathbb{R}^{k \times k}$, $\mathbf{Q} = (\mathbf{q} | \dots | \mathbf{q}) \in \mathbb{R}^{d \times k}$. When $k \geq d$, regularization is applied to \mathbf{C}_j before inversion such as $\mathbf{C}_j + r \mathbf{I}_k$, where $r > 0$ and $\mathbf{I}_k \in \mathbb{R}^{k \times k}$ are a regularization parameter and an identity matrix, respectively. After this optimization, the shortest distance between a test sample \mathbf{q} and a reconstructed sample $\mathbf{X}_j \mathbf{w}_j$ is measured by

$$d_j = \|\mathbf{q} - \mathbf{X}_j \mathbf{w}_j\|^2, \quad (3)$$

and \mathbf{q} is classified into the class to which the nearest reconstructed sample belongs, i.e., test sample's class (denoted by ω) is determined by $\omega = \arg \min_j d_j$.

The LSC rule employs k as a hyper-parameter. When $k > 1$, LSC can interpolate the sparsity between neighbor training samples in a high-dimensional space. In fact, substitute the constraint $\mathbf{1}_k^\top \mathbf{w}_j = 1$ for d_j , we

get $d_j = \|\mathbf{q} - (\mathbf{x}_1^j + \sum_{i=2}^k (\mathbf{x}_i^j - \mathbf{x}_1^j) w_i^j)\|^2$. This equation means the minimum distance from a test sample to a $(k-1)$ -dimensional affine subspace (cf. Fig. 1). Obviously, d_j is equal to or smaller than to the nearest training sample, i.e., $d_j \leq \|\mathbf{q} - \mathbf{x}_1^j\|^2$ in any case because d_j can be calculated as $d_j = \|\mathbf{q} - \mathbf{x}_1^j\|^2 - \|\mathbf{x}_1^j - \mathbf{X}_j \mathbf{w}_j\|^2$. When $k = 1$, LSC is equivalent to the NN rule, and if all elements of \mathbf{w}_j are equal to $1/k$, LSC is equivalent to LMC [1]. Furthermore, if k -neighbors are selected not for a test sample but for each training one, LSC is equivalent to local manifold matching [10]. Hence LSC can be regarded as a general-classifier that includes most of neighbor-based classifiers.

3. Learning LSC

In neighbor-based classifiers such as the NN rule, memory requirements tend to be large, so GLVQ [6] was proposed for reducing memory requirement without accuracy deterioration. In GLVQ, prototypes called reference (codebook) vectors are updated by a steepest descent method that minimizes a cost function defined with a training error criterion [6]. In this paper, *learning LSC* (LLSC) is derived by applying the same learning rule of GLVQ to LSC for reducing the number of prototypes without accuracy deterioration.

3.1. Formulation

Let $\mathbf{x}_i^j = (x_{i1}^j \dots x_{id}^j)^\top \in \mathbb{R}^d$ be the i th prototype belonging to class j . When an input training sample \mathbf{x} is given, the k -closest prototypes to \mathbf{x} are selected from each class, and denote a set of the selected k -neighbor prototypes from class j by a matrix $\mathbf{X}_j = (\mathbf{x}_1^j | \mathbf{x}_2^j | \dots | \mathbf{x}_k^j) \in \mathbb{R}^{d \times k}$.

After neighbor selection, a distance between \mathbf{x} and an affine subspace of class j is measured by Eq. 3 with the optimized weights. Let \mathbf{X}_1 be the set of k -neighbor prototypes selected from the same class as \mathbf{x} . In contrast, let \mathbf{X}_2 be the set of k -neighbor prototypes selected from the nearest different class from \mathbf{x} . Let us consider the relative distance difference $\mu(\mathbf{x})$ defined as follows:

$$\mu(\mathbf{x}) = \frac{d_1 - d_2}{d_1 + d_2}, \quad (4)$$

where d_1 and d_2 represent distances from \mathbf{x} to $\mathbf{X}_1 \mathbf{w}_1$ and $\mathbf{X}_2 \mathbf{w}_2$, respectively. The above $\mu(\mathbf{x})$ satisfies $-1 < \mu(\mathbf{x}) < 1$. If $\mu(\mathbf{x})$ is negative, \mathbf{x} is classified correctly; otherwise, \mathbf{x} is misclassified. For improving accuracy, minimize the following cost function:

$$S = \sum_{i=1}^N f(\mu(\mathbf{x}_i)), \quad (5)$$

where N is the total number of input training samples, and $f(\mu)$ is a monotonically increasing function. To minimize S , a steepest descent method with a small positive constant γ ($0 < \gamma < 1$) is adopted to \mathbf{X}_j :

$$\mathbf{X}_j \leftarrow \mathbf{X}_j - \gamma \frac{\partial S}{\partial \mathbf{X}_j}, j = 1, 2. \quad (6)$$

Now $\partial S / \partial \mathbf{X}_j$ is derived as

$$\begin{aligned} \frac{\partial S}{\partial \mathbf{X}_j} &= \frac{\partial S}{\partial \mu} \frac{\partial \mu}{\partial d_j} \frac{\partial d_j}{\partial \mathbf{X}_j} \\ &= \mp \frac{\partial f}{\partial \mu} \frac{4d_{3-j}}{(d_1 + d_2)^2} (\mathbf{x} - \mathbf{X}_j \mathbf{w}_j) \mathbf{w}_j^\top \quad (j = 1, 2), \end{aligned} \quad (7)$$

Consequently, the update rule of LLSC can be written as follows:

$$\mathbf{X}_j \leftarrow \mathbf{X}_j + \gamma_j \frac{\partial f}{\partial \mu} \frac{d_{3-j}}{(d_1 + d_2)} (\mathbf{x} - \mathbf{X}_j \mathbf{w}_j) \mathbf{w}_j^\top, \quad (8)$$

where $\gamma_2 = -\gamma_1$. In the above rule, correction $d_j / (d_1 + d_2)$ dose not affect the convergence condition [6], and $f(\mu, t) \{1 - f(\mu, t)\}$ is used for $\partial f / \partial \mu$, where t is learning time and $f(\mu, t)$ is a sigmoid function $1 / (1 + e^{-\mu t})$. In this case, $\partial f / \partial \mu$ has a single peak at $\mu = 0$, and the peak width becomes narrower as t increases. After the above training, a test sample is classified by LSC with trained prototypes. When $k = 1$, LLSC is equivalent to GLVQ, so LLSC can be regarded as a natural extension of GLVQ. In addition, if all elements of \mathbf{w}_j are fixed to $1/\sqrt{k}$, the above process can be regarded as a learning rule for LMC. Hence, we call LLSC that uses fixed weights $\mathbf{w}_j = (1/\sqrt{k} \cdots 1/\sqrt{k})^\top$ *learning LMC* (LLMC).

3.2. Accuracy improvement

We encounter different types of geometric transformations in image classification. Hence, it is important to incorporate transform-invariance into classification rules for achieving high accuracy. Distance-based classifiers such as the NN rule often rely on simple distances such as Euclidean distance, thus they suffer a high sensitivity to geometric transformations of images such as shifts, scaling and others. In LSC, the distance is measured based on a square error, so it is also not robust against geometric transformations. Hence, combining *tangent distance* (TD) [12] and LLSC is presented here for accuracy improvement. Henceforth, a combination of LLSC and TD is denoted as LLSC+TD for short.

In LLSC+TD, two-sided TD and linearly transformed images are used for selecting neighbor training samples and updating prototypes. Let $\mathbf{x} \in \mathbb{R}^d$ and $\mathbf{T} \in \mathbb{R}^{d \times r}$ be an input training image and its tangent

vectors, where r is the number of geometric transformations. The distance between the linear combination $\mathbf{x} + \mathbf{T}\boldsymbol{\alpha}$ and $\mathbf{X}_j \mathbf{w}_j$ is given by

$$\begin{aligned} \min_{\boldsymbol{\alpha}, \mathbf{w}_j} \quad & \|\mathbf{x} + \mathbf{T}\boldsymbol{\alpha} - \mathbf{X}_j \mathbf{w}_j\|^2 \\ \text{s.t.} \quad & \mathbf{1}_k^\top \mathbf{w}_j = 1. \end{aligned} \quad (9)$$

By using Lagrange multipliers, the optimal weights $\boldsymbol{\alpha}$ and \mathbf{w}_j are given as follows:

$$\begin{aligned} \mathbf{w}_j &= (\mathbf{X}_j^\top (\mathbf{I}_d - \mathbf{A}) \mathbf{X}_j)^{-1} \{ \mathbf{X}_j^\top (\mathbf{I}_d - \mathbf{A}) \mathbf{x} + \beta \mathbf{1}_k \}, \\ \boldsymbol{\alpha} &= (\mathbf{T}^\top \mathbf{T})^{-1} \mathbf{T}^\top (\mathbf{X}_j \mathbf{w}_j - \mathbf{x}), \end{aligned} \quad (10)$$

where \mathbf{A} and β are

$$\mathbf{A} = \mathbf{T} (\mathbf{T}^\top \mathbf{T})^{-1} \mathbf{T}^\top, \quad (11)$$

$$\beta = \frac{1 - \mathbf{1}_k^\top (\mathbf{X}_j^\top (\mathbf{I}_d - \mathbf{A}) \mathbf{X}_j)^{-1} \mathbf{X}_j (\mathbf{I}_d - \mathbf{A}) \mathbf{x}}{\mathbf{1}_k^\top (\mathbf{X}_j^\top (\mathbf{I}_d - \mathbf{A}) \mathbf{X}_j)^{-1} \mathbf{1}_k}. \quad (12)$$

By the above optimal weights, the distance between $\mathbf{x} + \mathbf{T}\boldsymbol{\alpha}$ and $\mathbf{X}_j \mathbf{w}_j$ can be defined as $d_j^{TD} = \|\mathbf{x} + \mathbf{T}\boldsymbol{\alpha} - \mathbf{X}_j \mathbf{w}_j\|^2$. Hence, the learning rule for LLSC+TD can be derived by the same manner as that for deriving LLSC:

$$\mathbf{X}_j \leftarrow \mathbf{X}_j - \delta_j (\mathbf{x} + \mathbf{T}\boldsymbol{\alpha} - \mathbf{X}_j \mathbf{w}_j) \mathbf{w}_j^\top, \quad (13)$$

where $\delta_j = (-1)^j \gamma \frac{\partial f}{\partial \mu} \frac{d_{3-j}^{TD}}{(d_1^{TD} + d_2^{TD})}$ ($j = 1, 2$). Consequently, the algorithm of LLSC+TD is summarized as follows: First, find the k -closest training samples to a test sample from class j according to d_j^{TD} , and denote them as \mathbf{X}_j . Next, update \mathbf{X}_j according to Eq. 13.

4. Experiments

Experimental results on the handwritten digit image dataset USPS [9] are shown. In general, handwritten patterns include various geometric-transformation such as rotation, so it is difficult to reduce the number of prototypes. The USPS dataset consists of 7,291 training and 2,007 test images. The size of images is 16×16 pixels. The effectiveness of LLSC was verified with comparing to *other classifiers*: SVM, GLVQ, LLMC, and *averaged learning subspace method* (ALSM) [11]. They were implemented with MATLAB on a standard PC that has Pentium 1.86GHz CPU and 2GB RAM.

In experiments, intensities of images were directly used as feature vectors. The set of training images was randomly split into two sets of equal size (i.e., about 3600 images in each set). One of them was used for initializing prototypes and the other was used as input training samples for learning. This random splitting was performed independently for ten repetitions of training

Table 1. Error rates [%] and standard deviations on USPS.

| n_j | LSC | SVM | GLVQ | LLMC | LLSC | LLSC+TD | ALSM |
|-------|------------|------------|-----------|-----------|------------------|------------------|-----------|
| 10 | 16.4 ± 1.5 | 22.1 ± 1.3 | 7.1 ± 0.4 | 8.3 ± 0.4 | 5.6 ± 0.2 | 4.1 ± 0.2 | 7.7 ± 0.5 |
| 50 | 8.2 ± 0.4 | 11.6 ± 0.9 | 7.4 ± 0.3 | 6.6 ± 0.5 | 5.0 ± 0.2 | 3.4 ± 0.1 | 6.3 ± 0.4 |
| 100 | 6.5 ± 0.4 | 8.6 ± 0.5 | 6.8 ± 0.3 | 6.2 ± 0.2 | 4.8 ± 0.2 | 3.0 ± 0.2 | 6.0 ± 0.3 |
| all | 4.2 | 4.2 | 5.5 | 4.7 | 4.2 | 2.2 | 5.1 |

and testing to estimate error rate and its standard deviation. The number of prototypes of class j (denoted as n_j) was changed as 10, 50, and 100. The fixed learning rate $\alpha = 0.1$ and 100 learning time were introduced for GLVQ, LLMC, and LLSC. The parameters of individual classifiers were tuned on input training samples in each step. For SVM, we used the RBF kernel for nonlinear mapping. For LSC and SVM, initial prototypes were directly used as training samples for classification. For LLSC+TD, seven tangent vectors (i.e., $r = 7$) were used for approximating geometric transformations (cf. [12]).

Table 1 shows error rates with different prototype sizes. As shown in Table 1, LLSC outperformed other classifiers in all prototype sizes. Particularly, LLSC+TD with 10 prototypes performed as well or better than other classifiers with all training samples because LLSC can expand a representation capacity of available prototypes. ALSM also uses subspaces, but decision boundaries of it would be poor for complex distributed patterns.

5. Conclusion

In this paper, a *learning local subspace classifier* (LLSC) was proposed for achieving high accuracy with small number of prototypes. In LLSC, all k -neighbor prototypes were updated with the same update manner as that of *generalized learning vector quantization* (GLVQ). It was verified with experiments on handwritten digit recognition that LLSC achieved lower error rates than other classifiers such as GLVQ.

The advantages of LLSC are summarised as follows: 1) LLSC can achieve high accuracy even if dimensionalities of feature vectors are large because LSC can extend a representation capacity of prototypes. 2) LLSC can achieve high accuracy for pattern classes that have nonlinear decision boundaries because LLSC uses k -neighbor prototypes for classification. 3) LLSC can reduce the number of prototypes without accuracy deterioration. 4) LLSC is suitable for multi-class problems because it is not a binary classifier such as SVM. However, training cost for LLSC is high. Future work will be

dedicated to speed up a training phase and adopt LLSC to other classification tasks.

References

- [1] Y. Mitani and Y. Hamamoto. A local mean-based nonparametric classifier. *Patt. Recog. Lett.*, 27(10):1151–1159, 2006.
- [2] H. Zhang *et al.* SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. *CVPR*, 2126–2136, 2006.
- [3] L. Cazzanti and M.R. Gupta. Local similarity discriminant analysis. *ICML*, 2007.
- [4] J. Laaksonen. *Subspace classifiers in recognition of handwritten digits*. PhD thesis, Helsinki Univ. of Tech., 1997.
- [5] P. Vincent and Y. Bengio. K-local hyperplane and convex distance nearest neighbor algorithms. *Proc. of NIPS*, 2001.
- [6] A. Sato and K. Yamada. Generalized learning vector quantization. *Prop. of NIPS*, 7:423–429, 1995.
- [7] T. Kohonen. *Self-organizing maps*. 2nd Ed. Springer-Verlag, Heidelberg., 1995.
- [8] K. Crammer, R.G. Bachrach, and A. Navot. Margin analysis of the LVQ algorithm. *NIPS*, 2003.
- [9] Y. LeCun, *et al.* Backpropagation applied to handwritten zip code recognition. *Neur. Comp.*, 1(4):541–551, 1989.
- [10] W. Liu *et al.* Local manifold matching for face recognition. *ICIP*, 2:926–929, 2005.
- [11] E. Oja. *Subspace methods of pattern recognition*. Research Studies Press, 1983.
- [12] P.Y. Simard, Y. LeCun, and J.S. Denker. Efficient pattern recognition using a new transformation distance. *NIPS*, 5:50–58, 1993.