ELECTRONICS DIVISION

EUROPEAN WORKSHOP ON

HANDWRITING ANALYSIS AND RECOGNITION:

A EUROPEAN PERSPECTIVE

Hotel Albert Premier
20 Place Rogier
Brussels

12-13 July 1994

Application of suboptimal Bayesian classification to handwritten numerals recognition

Jean-Luc Voz, Philippe Thissen*, Michel Verleysen** and Jean-Didier Legat

Université Catholique de Louvain, Microelectronics Laboratory
3 Place du Levant, B-1348 Louvain-La-Neuve, Belgium
tel:+ 32 (10) 472551, fax:+32 (10) 478667, E-mail:voz@dice.ucl.ac.be

Abstract. Non-parametric estimation of probability densities provides a useful way to realise Bayesian classifiers that may be used for example in OCR problems. The complexity of conventional kernel estimators is however far beyond the acceptable limits for performant systems. We present in this paper a novel Learning Vector Quantization technique (IRVQ) which allows to strongly decrease the complexity of kernel estimators. We apply this original technique to the recognition of handwritten numerals and we prove its interest through high recognition rates coupled to low memory and computational requirements.

1 Introduction

Classification of high-dimensional data is a challenging task in engineering science. The Bayes theory provides the mathematical tools to study classification problems; however, to minimize the number of misclassifications in a problem according to the Bayes law, the knowledge of the probability densities in each class is necessary. Estimations of these probability densities may be obtained through kernel estimators, or Parzen windows [1, 2]; however, such methods involve tremendous high numbers of computations, and are thus inefficient in most practical applications.

We present here a novel vector quantization technique (IRVQ) to drastically reduce the number of operations involved in probability densities estimation, by selecting a limited number of points representing the training distribution, and estimating the densities from these points only. This method is applied to the recognition of writer-independent handwritten numerals, and is tested on a database of handwritten digits. The preprocessing used before the classifier is a classical extraction of topological characteristics; the test results show very good performances of the classifier itself. The advantages of our method comes from the adaptive training as in neural networks, and also from the limited numbers of computations as required in real-time OCR systems.

2 Bayesian classification

Assume the problem consists of classifying an observed vector u of $\mathbb{R}^d$ among c classes denoted ω_j. Assume that u is random and that its d components admit a joint density $p_x(u|\omega_j)$ in class ω_j. If all wrong decisions are given the same penalty, the Bayes law may be expressed as:

$$P(\omega_i|u) = \frac{p_x(u|\omega_i)P(\omega_i)}{\sum_{j=1}^{c} p_x(u|\omega_j)P(\omega_j)}, \tag{1}$$

Part of this work has been funded by the ESPRIT project ELENA-Nerves II, supported by the CEC.

* Fellow of the Belgian IRSIA.

** Senior Research Assistant of Belgian National Fund for Scientific Research (FNRS).

where $P(\omega_j)$ is the a priori probability of class ω_j, and $P(\omega_i|u)$ the probability that vector u belongs to class ω_i. The Bayesian decision to select the most probable class will thus be:

$$Decide\ u \in \omega_s \Leftrightarrow s = Arg \max_{1 \leq i \leq c} \{P_i\ p_x(u|\omega_i)\}. \tag{2}$$

It is quite obvious that such an ideal Bayesian solution can be used only if distributions $p_x(u|\omega_i)$, and the c a priori probabilities P_i are known. In the problems we are interested in, it is seldom the case. We rather have at disposal a set of patterns, $A_N = \{x(n), \omega_{x(n)}, 1 \leq n \leq N\}$, where each pattern $x(n)$ belongs to a known class $\omega_{x(n)}$. Denote N_i the number of available patterns in class $\omega_i, 1 \leq i \leq c$, $\sum_{i=1}^{C} N_i = N$, and $A_{N_i} = \{x(n) \mid \omega_{x(n)} = \omega_i,\ 1 \leq n \leq N\}$.

One way of performing Bayesian classification is to compute the best estimate of each density $p_x(u|\omega_i)$ in the Mean Square sense with the help of the N_i patterns available. Kernel estimators [1, 3] provide a useful way to estimate probability densities. The kernel estimate of the density $p_x(u|\omega_i)$ of a random variable x takes the following general form:

$$\hat{p}_x(N_i, u|\omega_i) = \frac{1}{N_i} \sum_{n=1}^{N_i} K\left(\frac{u - x(n)}{h(n)}\right) \tag{3}$$

where $\{x(n), 1 \leq n \leq N_i\}$ denote the available patterns in a given class ω_i and $K(\cdot)$ a kernel function. The parameter $h(n)$ is called the *width factor* of the kernel. If $h(n)$ is not allowed to depend on index n, the kernel is referred to as *fixed*, whereas it is referred to as *variable* when the width factor may be different for each $x(n)$. Better estimates are always obtained with variable kernel width factors, but an important problem is to obtain their optimal values. Several types of radial kernels $K(\cdot)$ may be used, the most classical one being a Gaussian function:

$$K\left(\frac{u - x(n)}{h(n)}\right) = \frac{1}{(h(n)\sqrt{2\pi})^d} exp(-\frac{1}{2}\left(\frac{||u - x(n)||}{h(n)}\right)^2), \tag{4}$$

where d is the dimension of u and $x(n)$.

In the following, we will see how to decrease the number of samples N_i in each class, in the case of large training sets, to obtain acceptable number of computations in practical applications.

3 IRVQ classifier

The problem with conventional kernel estimators resides in the number of operations involved in the computation of equation (3); the number N of prototypes in the training database is usually large, and may lead to unacceptable computation times for on-line applications like OCR. The solution to this problem is to build the kernel classifier from a reduced design set, chosen to keep the performances as near as possible from those of the classifier built on the entire original learning set.

The method we propose in this paper is to apply vector quantization techniques to build the reduced design set, using the statistical information of the original one during the quantization process [4, 5].

3.1 Vector quantization

Vector quantization provides a useful way to reduce the number of high-dimensional vectors in a set, while keeping their distribution unchanged [6]. The Generalized Lloyd Algorithm

(GLA) [7] is the most widely used one in various industrial problems such as image compression. A neural network "on-line" method ("competitive learning", or "Kohonen Learning Algorithm (KLA)") which provides an interesting computationally efficient substitute to the GLA algorithm for comparable overall performances may be found in [8]. A convergence analysis of this algorithm may be found in [9].

The aim of the Kohonen Learning Algorithm is to approximate the sets of patterns A_{N_i} by sets of so-called centroids $B_{M_i} = \{c(m),\ \omega_{c(m)} = \omega_i,\ 1 \leq m \leq M_i\}$, where $M_i << N_i$, and roughly keeping the same probability density of vectors in the space for the sets A_{N_i} and B_{M_i}. The principle of the KLA method is then the following in each class ω_i. First, the M_i centroids $c(m)$ are randomly initialized to any of the N_i patterns, keeping the same a priori probabilities of classes for both sets A_{N_i} and B_{M_i}. Then, each of the N_i patterns $x(n)$ is presented to the set B_{M_i}; the centroid $c(a)$ closest from the presented pattern $x(n)$ is then selected and moved in the direction of $x(n)$:

$$c(a) = c(a) + \alpha(x(n) - c(a)) \tag{5}$$

where α is an adaptation factor ($0 \leq \alpha \leq 1$) which must decrease with time during the learning to ensure the convergence of the algorithm. After several presentations of the whole set of patterns A_{N_i}, the distribution of centroids $c(m)$ in B_{M_i} will reflect this of the pattern set A_{N_i}.

The purpose of this vector quantization technique in our problem of estimation of probability densities for Bayesian classification is then to use the reduced set B_{M_i} instead of the original set A_{M_i} for the estimation of probability densities (equation 3), decreasing by the way the number of computations involved in this estimation. The width factors $h(n)$ must however still be estimated in an adequate manner.

3.2 Width factors

During the adaptation process (5), it is possible to keep a trace of the mean distance between a pattern $x(n)$ and its closest prototype $c(a)$, by affixing an *inertia* coefficient $i(m)$ to each centroid $c(m)$, $1 \leq m \leq M_i$. This inertia coefficient is randomly initialized to a small value and then adapted at the same time as the centroid locations (equation 5) according to:

$$i(a) = i(a) + \alpha(\|x(n) - c(a)\|^2 - i(a)) \tag{6}$$

This equation means that the inertia coefficient $i(a)$ is adapted at each presentation of a pattern $x(n)$ to a convex combination between the actual value of $i(a)$ and the norm of the distance between $x(n)$ and the closest centroid $c(a)$.

After learning, parameters $i(m)$, $1 \leq m \leq M_i$, will converge to the average inertia of points in the clusters associated to $c(m)$. We will use this inertia coefficient $i(m)$ for the computation of the optimal variable width factor associated to the kernel centered on $c(m)$ in the reduced classifier.

Let us make now the main hypothesis of this estimation of width factors: we consider that the size of the clusters is small enough to approximate the true density $p_x(u|\omega_i)$ in any class i by a constant over two consecutive clusters. Clusters are here defined as the sets of points nearest from their associated centroid $c(a)$ than from any other centroid $c(m), 1 \leq m \leq M_i$; consecutive clusters are clusters sharing a common border. This assumption is of course only an approximation, but is not too restrictive when the true densities are smooth.

The purpose will then be to fix the width factors in order to have a constant estimate of the probability densities $\hat{p}_x(M_i, u/\omega_i)$ over two consecutive clusters too. Let us first examine the problem in dimension 1, with an estimate computed by the sum of two kernels A and B (figure 1). Using equation 4 for both kernels, the estimate of the probability density at

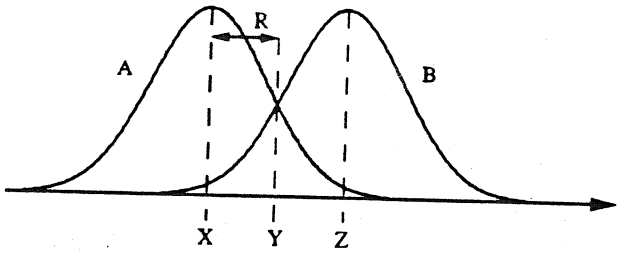

Fig. 1. Illustration of two Gaussian functions in 1 dimension, and related notations.

location X is given by

$$\hat{p}_x(\{A,B\},X) = \frac{1}{h(n)\sqrt{2\pi}} \tag{7}$$

if the contribution of kernel B is neglected at X (this is a second hypothesis aimed to reduce the complexity of the computations). In the same way, the estimate of the probability density at location Y is given by

$$\hat{p}_x(\{A,B\},Y) = \frac{2}{h(n)\sqrt{2\pi}}\, e^{\frac{-R^2}{2\,h(n)^2}} \tag{8}$$

where R is the distance between X and Y ($2R$ is the distance between two consecutive centroids). Making the estimates 7 and 8 equal to have an equal approximation of probability density at points X, Y and Z leads then to

$$R = \sqrt{2\ln 2}\, h(N) \tag{9}$$

A similar development may be done in dimension 2. In this case, we can consider the approximation of probability density due to four Gaussian functions A, B, C and D centered on the four vertices of a square:

- at the location X of a vertex of the square,
- at the center Y of the square, and
- at the midpoint Z of an edge of the square.

Keeping the distance between two centroids on an edge of the square being equal to $2R$, and neglecting the influence of kernels at a distance greater or equal to $2R$, we have respectively:

$$\hat{p}_x(\{A,B,C,D\},X) = \frac{1}{\left(\sqrt{2\pi}\, h(N)\right)^2} \tag{10}$$

$$\hat{p}_x(\{A,B,C,D\},Y) = \frac{4}{\left(\sqrt{2\pi}\, h(N)\right)^2}\, e^{\frac{-R^2}{2\,h(n)^2}} \tag{11}$$

$$\hat{p}_x(\{A,B,C,D\},Z) = \frac{2}{\left(\sqrt{2\pi}\, h(N)\right)^2}\, e^{\frac{-R^2}{2\,h(n)^2}} \tag{12}$$

It is possible to make the estimations 10, 11 and 12 equal, by setting the width factor $h(n)$ according to equation 9, which gives thus the same result as in dimension 1. An identical development can be made in dimension 3, by considering the influence of 8 Gaussian kernels located on the vertices of a cube, respectively at the locations of these vertices, of the midpoint of any edge, of the center of a face, and of the center of the cube. Again, the estimations of probability densities will be equal if equation 9 is respected; it will also be the case in dimension d greater than 3.

The last step is now to set the relation between the estimated inertia in each cluster $h(n)$ and the size of the cluster, fixed by R. For this purpose, we make the supplementary hypothesis that within a short volume in the d-dimensional space, all centroids are equally spaced on a regular isotropic grid; this hypothesis is similar to the one of a constant true density over consecutive clusters, since a vector quantization applied to a constant distribution will lead to centroids on such a regular isotropic grid. We can thus consider that a cluster associated to a particular centroid will be a d-dimensional hypercube with edges of length $2R$, denoted by V. The inertia in such a cluster is thus

$$i(m) = \frac{1}{(2R)^d} \int_V \|x(n) - c(m)\|^2 dV = \frac{dR^2}{3} \tag{13}$$

Combining equations 9 and 13 then leads to a width factor $h(n)$ given, in dimension d, by

$$h(n) = \sqrt{\frac{3\,i(n)}{2\,d\ln 2}} \tag{14}$$

Finally, the estimation of probability density in each class will be calculated through equation 3, applied on a set of centroids fixed by 5, and the width of the kernels being fixed by 14. Bayesian classification is then realized through equation 2, where the probability densities are replaced by the above estimates, and the a priori probabilities by percentage of occurrence of prototypes $x(n)$ in each class. This constitutes the IRVQ (Inertia-Rated Vector Quantization) method.

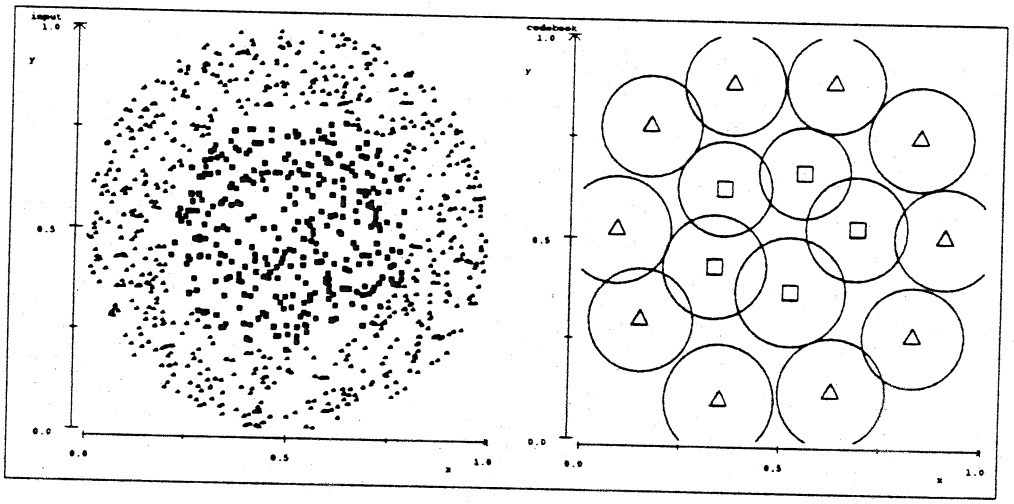

Fig. 2. Centroids obtained by the IRVQ algorithm on a two-class bidimensional database.

For illustration purposes, the IRVQ algorithm has been applied on an artificial database. Figure 2 shows the results of the IRVQ algorithm on a two-class problem with uniform concentric circular distributions; centroids are represented with a circle of radius $h(n)$.

4 Application to Optical Character Recognition

4.1 Database and preprocessing

A number of simulations have been carried out on a small real-world dataset provided by the ATT Bell Laboratories [10, 11]. It consists of 1200 handwritten numerals, containing 120

examples of each digit, written by 12 differents people. Each person has written 10 times the same succession of numerals from 0 to 9. Figure 3 presents one of these successions. Each data is a centered bitmap normalized to the size 16x16 pixels. Among the 10 examples written by each person, the first 5 examples were placed in the training set, and the remaining 5 were placed in the test set. The problems encountered by a handwritten recognition system are

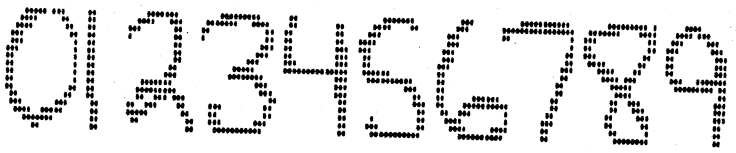

Fig. 3. Examples of handwritten digits from the database.

important : variation of sizes, presence of defaced characters, similar shapes, ... Topological features are well adapted to size variation and makes possible the recognition of handwritten characters with very few constraints. Topological recognition techniques trying to represent the general "concept of the character" based on a general description of its structure have been investigated by many researchers [12].

Each character is described after preprocessing by a feature vector F=(f1, f2, f3, ..f15) whose elements are topological information about the character. The features have been designed to minimize their number. Minimizing the number of features greatly reduces the memory size needed by the classification.

Topological Features are extracted from the binary image. The character is described in term of "bars" and "holes". Furthermore, the approximate position of these bars and holes is also used. The bitmap image is scanned in four directions (top to bottom, bottom to top, left to right and right to left) and projections, outlines and stroke densities vectors are calculated. These are typical low-level features that can be calculated along various directions [13]. For a typical 16 x 16 image, 12 features vectors with a total of 192 elements are generated. These basic features are then combined to extract the bars and holes present in the image of the character.

The topological feature vector contains 15 elements and has a fixed size :

- f0 represents the aspect ratio of the character,
- f1 to f3 represent upper holes (left, middle, right),
- f4 to f6 represent lower holes (left, middle, right),
- f7 to f9 represent left holes (top, middle-up, middle-down),
- f10 to f12 represent right holes (top, middle-up, middle-down),
- f13 to f14 represent horizontal bars (top, bottom).

4.2 Recognition results

The training of the IRVQ algorithm was performed by presenting 20 times the 600 preprocessed training samples of the Bell database, using 5 centroids per class, the α adaptation factor linearly decreasing from 0.3 to 0 during the learning.

In these conditions the global recognition rate of our classifier is 98.7%. Table 1 shows the confusion matrix on the 600 samples of the test set. Each line of the confusion matrix represents a class; the percentage of elements from this class attributed to any class is given in each of the elements of the line.

In order to compare the performances of our classifier with those reported in [10] for the same problem, the estimate of the required memory size has been computed by the same

Class	0	1	2	3	4	5	6	7	8	9
0	96.6	0.0	0.0	0.0	0.0	0.0	1.7	0.0	1.7	0.0
1	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	1.7	98.3	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	95.0	0.0	0.0	0.0	3.3	1.7
5	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	1.7	0.0	98.3	0.0	0.0
8	1.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	98.3	0.0
9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0

Table 1. Confusion matrix on the test set of the Bell database.

method. To store the topological feature vector associated with each centroid, we need 120 bits (there are 15 features which may be coded on 8 bits); this value must then be multiplied by the number of centroids (50) and summed with the memory size needed to store the width factors of each centroid (50x8 bits) to obtain the estimated classifier size: 6400 bits.

Table 2 compares the performances and estimated memory sizes of the IRVQ classifier with the values obtained for the two best classifiers reported in [10], which are two 2 by 2 class separation neural network algorithms. This illustrates the excellent performances and low-cost properties of our method.

Method	Performance	Size
IRVQ	98.7	6 400
1	98.0	190 850
2	97.0	29 537

Table 2. Performances and sizes of the IRVQ classifier in comparison with the two best classifiers reported in [10].

5 Conclusion and future works

The use of vector quantization techniques allows to limit the sizes of databases to reasonable values. We showed in this paper how to extend these techniques to serve as preprocessing to Bayesian kernel classifiers using estimates of in-class probability densities. We applied this technique to the recognition of handwritten numerals, and found high recognition rates coupled to low requirements in terms of memory and computational resources. The results are encouraging for the use of the technique presented in this paper to more complex OCR problems, our aim being to apply this method to the problem of on-line cursive character recognition. A performant adaptive vector quantization method such as the one presented in [14] could also be used for this purpose. The classification results obtained with the IRVQ method on several databases lead us to study its implementation on a fully parallel mixed analog-digital architecture [15].

References

1. T. Cacoullos, "Estimation of a multivariate density", *Annals of Inst. Stat. Math.*, vol. 18, pp. 178–189, 1966.
2. P. Comon, J.L. Voz, and M. Verleysen, "Estimation of performance bounds in supervised classification", in *ESANN94-European Symposium on Artificial Neural Networks*, M. Verleysen, Ed., Brussels, Belgium, April 1994, pp. 37–42, D facto publications.

3. E. Parzen, "On the estimation of a probability density function and the mode", *Ann. Math. Stat.*, vol. 27, pp. 1065–1076, 1962.
4. P. Comon, "Classification bayésienne distribuée", *Revue Technique Thomson CSF*, vol. 22, no. 4, pp. 543–561, 1990.
5. Q. Xie, C. A. Laszlo, and R. K. Ward, "Vector quantization technique for nonparametric classifier design", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 12, pp. 1326–1330, december 1993.
6. A. Gersho, "Asymptotically optimal block quantization", *IEEE Transactions on Information Theory*, vol. IT-25, pp. 373–80, July 1979.
7. Y. Linde, A. Buzo, and R.M. Gray, "An algorithm for vector quantizer design", *IEEE Transactions on Communications*, vol. 28, pp. 84–95, January 1980.
8. T. Kohonen, *Self-Organization and Associative Memory*, Springer-Verlag, Berlin, 1984.
9. E. Yair, K. Zeger, and A. Gersho, "Competitive learning and soft competition for vector quantizer design", *IEEE Transctions on Signal Processing*, vol. 40, no. 2, pp. 294–309, 1992.
10. I. Guyon et al., "Comparing different neural architectures for classifying handwritten digits", in *IJCNN89*, Whashington, 1989.
11. L.D. Jackel et al., "An application of neural net chips : Handwritten digits", in *IEEE Int. Conf. on Neural Nets*, San Diego, July 1988, pp. 107–115.
12. C. Y. Suen, "Distinctive features in automatic recognition of handprinted characters", *Signal Processing*, vol. 4, pp. 193–207, 1982.
13. P. De Muelenaere, M. Dauw, and J.D. Legat, "Omnifont recognition using topological recognition techniques", in *11th IAPR , Int. Conf. on Pattern Recognition*, The Hague, Sepember 1992, pp. 410–413, vol. B.
14. P. Demartines and J. Hérault, "Representation of nonlinear data structures through a fast VQP neural networ3", in *NeuroNimes93 (Neural Networks and their applications)*, Paris, France, October 1994, pp. 411–424, EC2.
15. M. Verleysen, P. Thissen, J.L. Voz, and J. Madrenas, "An analog processor architecture for neural network classifier", *IEEE Micro*, vol. 14, no. 3, pp. 16–28, June 1994.