A statistical neural network for high-dimensional vector classification

Michel Verleysen[*] and Jean-Luc Voz and Philippe Thissen[†] and Jean-Didier Legat
Laboratoire de Microélectronique - DICE, Université catholique de Louvain,
3, place du Levant, B-1348 Louvain-la-Neuve, Belgium
verleysen@dice.ucl.ac.be

ABSTRACT

The minimum number of misclassifications in a multi-class classifier is reached when the borders between classes are set according to the Bayes criterion. Unfortunately, this criterion necessitates the knowledge of the probability density function of each class of data, which is unknown in practical problems. The theory of kernel estimators (Parzen windows) provides a way to estimate these probability densities, given a set of data in each class. The computational complexity of these estimators is however much too large in most practical applications; we propose here a neural network aimed to estimate the probability density function underlying a set of data, in a sub-optimal way (while performances are quite similar to those in the optimal case), but with a strongly reduced complexity which makes the method useful in practical situations. The algorithm is based on a "competitive learning" vector quantization of the data, and on the choice of optimal widths for the kernels. We study the influence of this factor on the classification error rate, and provide examples of the use of the algorithm on real-world data.

1. Introduction

The Bayes rule, well known in statistics, provides a criterion to fix the boundaries between classes in a multi-dimensional classification problem, in order to minimize the number of misclassifications. The criterion however supposes the knowledge of the a priori probabilities of the classes, and also of their underlying probability density functions.

While an estimate of the a priori probability of each class can be computed through the ratio between the number of samples in this class available in the learning set by the total number of samples, estimating the probability density functions is a more difficult problem. The principle of Parzen windows [2] or kernel estimators is to sum normalized kernel functions centered on each point in the data set; it can be proven that the sum of all kernel functions converges asymptotically to the true probability density of data, given some (realizable) assumptions on the kernels and on the sum.

The convergence is however only asymptotic; in order to obtain a good estimate of the probability density functions, one has to use a very large number of samples in the data sets. Since the estimate requires the computation of one distance, one non-linear function (the kernel), and one addition per sample in the data set, this can very quickly lead to a computational load much too high for a real-world application.

The idea in using a neural network to reduce the complexity of the method is then to replace the original set of data by a reduced set with the same underlying probability function, through an adaptive vector quantization method. At the same time, the optimal parameters (width of the kernels) are estimated in order to obtain a good estimate of the probability density functions through the sum of kernels, even with this reduced set of data. The hypothesis that we use to compute the optimal values of the kernel widths is that the clusters, i.e. the influence region of each point in the reduced set, are sufficiently small so that the true probability density function may be roughly approximated by a constant over one cluster.

In this paper, we first provide a brief introduction to the Bayesian classification theory and its approximation by the use of kernel classifiers. We then present the method to replace the original set of data by a reduced set, including the computation of the optimal widths of the clusters and the underlying hypotheses. We also present results of the method on real-world data, and show how to slightly modify the parameters of the network in order to take into consideration slight deviations from our hypotheses. The simulations give a qual-

[*] Research fellow of the Belgian National Fund for Scientific Research (FNRS)

[†] Ph.D. researcher under an IRSIA (Institut pour l'Encouragement de la Recherche Scientifique dans l'Industrie et l'Agriculture) fellowship

IEEE ICNN'95 proceedings - International Conference on Neural Networks
Perth (Australia), 27 November - 1 December 1995, Causal Productions, ISBN 0-646-26352-8, pp. 990-994

itative view of how the hypotheses of our method and of [4] must be used in different situations.

2. Statistical classification: theory and practice

A classification problem consists in attributing a class label to an observed vector u of $\mathcal{R}^d$ among c known classes denoted ω_i, $1 \leq i \leq c$. In the Bayesian context, it is assumed that any vector u belonging to a given class ω_k is drawn from a single conditional density $p(u|\omega_k)$ and that the occurrence of any class ω_i has a constant probability denoted $P(\omega_i)$. With these assumptions, if all wrong decisions are given the same penalty, the Bayes classification decision will be to select the most probable class, i. e. the class for which the product $p(u|\omega_i)P(\omega_i)$ is maximum.

2.1. Bayes-like classification with kernel density estimate

According to the Bayes law, the knowledge of the conditional densities $p(u|\omega_i)$ and of the a priori probabilities $P(\omega_i)$ of each class is needed to take the decision which minimizes the probability of misclassification for an observed vector u. But these values are never known in real case problems: we only have at our disposal a finite set A_N of observations $x(n)$, $1 \leq n \leq N$ having known classes $\omega_{x(n)}$: $A_N = \cup\{A_{N_i}\}$ with $A_{N_i} = \{x(n),\, \omega_{x(n)} = \omega_i,\, 1 \leq n \leq N_i\}$ and $N = \sum_{i=1}^{c} N_i$. The a priori probabilities $P(\omega_i)$ may be simply estimated by the relative frequency of the class occurrences in the learning set $\hat{P}(\omega_i) = N_i/N$.

A consistent estimate of a multivariate probability density function can be obtained by a kernel density estimator [2, 3]. Using such estimator, the probability density in each class ω_i can be estimated by

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

where $\{x(n), 1 \leq n \leq N_i\}$ denote the available patterns in class ω_i and $K(\cdot)$ a radial kernel function depending only on the norm of its argument. Parameter $h(n)$ is called the *width factor* of the kernel. The estimator is said to be "variable" if $h(n)$ depends of $x(n)$ and "fixed" otherwise. Variable estimators always provide better estimates, but it is very difficult to locally compute the optimal value of $h(n)$.

Due to their nice analytical properties, radial Gaussian kernels in dimension d are often used:

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

So, a classifier based on kernel density estimation require an extremely light computational cost during the learning (a simple storage of the training patterns) and have very good Bayes-like classification performances. Unfortunately, for large training sets the required memory size and the computational cost of the classification become incompatible with hardware constraints and real time classification tasks. The purpose of the suboptimal Bayesian classifier presented here is to drastically reduce the number of kernels N_i in each class, in order to use (1) in realistic situations, avoiding to reduce the quality of the density estimation.

2.2. The suboptimal Bayesian classifier

The principle of the proposed method is to use a vector quantization technique to split into clusters the portion of the space where vectors can be found. The aim is thus 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 $M_i << N_i$, roughly keeping the same probability density of vectors for sets A_{N_i} and B_{M_i}.

For the estimation of probability densities in each class, we then use the reduced sets B_{M_i} to build variable kernels estimators of each class instead of the original sets A_{N_i}; this strongly decreases the number of operations involved in (1).

The vector quantization used is an iterative version of the "Generalized Lloyd Algorithm" [6], the neural "Competitive Learning" (CL); the iterative character of this rule is used to set the position of the centroids and to evaluate the inertia of each cluster in order to obtain an approximation of the optimal variable width factors associated to each cluster. The principle of this method is 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}. Inertia coefficients $i(m)$ associated to each cluster are initialized to zero. Then, each of the N_i patterns $x(n)$ is presented to the set B_{M_i}, and the centroid $c(a)$ closest from $x(n)$ is selected and moved in the direction of the presented pattern while its inertia coefficient is updated:

$$c(a) = c(a) + \alpha(x(n) - c(a)) \qquad (3)$$

$$i(a) = i(a) + \alpha(\|x(n) - c(a)\|^2 - i(a)) \qquad (4)$$

where a is the index of the closest centroid to a learning vector $x(n)$ and α is an adaptation factor

IEEE ICNN'95 proceedings - International Conference on Neural Networks
Perth (Australia), 27 November - 1 December 1995, Causal Productions, ISBN 0-646-26352-8, pp. 990-994

($0 \leq \alpha \leq 1$) 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} reflects the one of the training set A_{N_i}, and the inertia coefficients $i(m)$, $1 \leq m \leq M_i$, converge to the average inertia of points in the clusters associated to $c(m)$ ((4) being a kind of convex combination at each iteration between the previously estimated value of $i(a)$ and a new contribution $||x(n) - c(a)||^2$ due to the input vector $x(n)$):

$$i(m) \simeq \frac{1}{n(m)} \sum_{v \in C(m)} ||v - c(m)||^2 \qquad (5)$$

where the sum goes on every point v of the original training set belonging to $C(m)$, the cluster associated to the centroid $c(m)$ in the Voronoi tessellation obtained after the vector quantization, and $n(m)$ is the number of these points.

At the end of the learning, and under the hypothesis of a sufficiently large number of centroids for a good coverage of the partition of the space where the classes are present, the clusters will be sufficiently small so that the true probability density inside each cluster can be approximated by a constant. We use this hypothesis to set the width factors of the Gaussian kernel function in order to keep the estimate (1) of the density as constant as possible over two consecutive clusters (clusters sharing the same border). Under this hypothesis, if we consider that the local arrangement of the centroids of consecutive clusters will be as the vertices of an hypercube, it may be shown [9] that the relation between $h(m)$, the optimal width factor of the Gaussian kernel function to set on $c(m)$ and the estimated inertia $i(m)$ is:

$$h(m)^2 = \frac{3}{2 \ln 2} \frac{i(m)}{d} \qquad (6)$$

where d is the data space dimension.

Finally, the estimation of probability density in each class will be calculated through a neural network implementing (1), applied on a set of centroids fixed by (3), the width of the kernels being fixed by (6). Bayesian classification is then realized by using the Bayes law where the probability densities are replaced by their estimates $\hat{P}(\omega_i)$ and $\hat{p}(M_i, u|\omega_i)$ (7).

3. Empirical results and discussion

3.1. Vector quantization effect on the codebook distribution

The first main hypothesis of the method we use to build the suboptimal Bayesian classifier is that the vector quantization process will lead to a distribution of centroids $c(m)$ in B_{M_i} similar to this of the training set A_{N_i} for each class. This hypothesis would be well verified if $n(m)$, the number of points belonging to $C(m)$ (the cluster associated to centroid $c(m)$ in the Voronoi tessellation obtained after the vector quantization) would approximately be constant for each cluster.

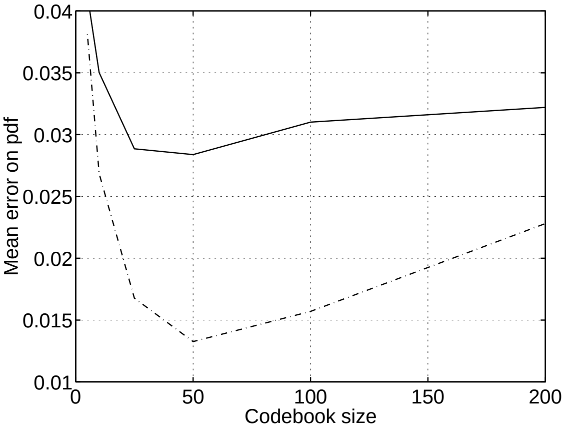

Fig. 1: Mean probability density estimation error of the suboptimal kernel estimator for the estimation of a Gaussian mixture with three modes using the number of points per cluster (dashed line) or not (solid line) .

Several experiments on artificial and real distributions showed us that this hypothesis is verified for large codebook sizes, but if we desire to drastically reduce the complexity of the estimator, the codebook size must be sufficiently small. In this case $n(m)$ can be locally approximated by a constant (over a few consecutive clusters), but will globally depend on the clusters position in the initial distribution. So, in order to keep the best approximation of the probability density function in each class, the estimator proposed in [4] will provide better results, and the equation of the kernel estimator based on the reduced design set B_{M_i}

$$\hat{p}(M_i, u|\omega_i) = \frac{1}{M_i} \sum_{m=1}^{M_i} K\left(\frac{u - c(m)}{h(m)}\right) \qquad (7)$$

has to be replaced by:

$$\hat{p}(M_i, u|\omega_i) = \frac{1}{N_i} \sum_{m=1}^{M_i} n(m)\, K\left(\frac{u - c(m)}{h(m)}\right) \qquad (8)$$

To illustrate this, we used the reduced estimator on a two-dimensional Gaussian mixture distribution with three modes containing 2500 training patterns $p(x) = p_1(x)/2 + p_2(x)/2 + p_3(x)$, where $p_1(x)$ and $p_2(x)$ are radial Gaussian functions of standard deviation $\sigma_x = \sigma_y = 0.2$ and of respective mean $(0,0)$ and $(0,2)$ while $p_3(x)$ is centered on (2,1) and has a diagonal covariance matrix with $\sigma_x = 0.2$ and $\sigma_y = 1$.

The estimator was built with a codebook size varying from 5 to 200; the CL learning consisted of 10 presentations of the 2500 training patterns with a α adaptation factor linearly decreasing from 0.3 to 0.001. Figure 1 shows the evolution of the mean error on the probability density function (pdf) estimation (the square root of the mean square error computed over a 50x50 grid covering more than 99.9% of the distribution) for estimators built with (7) and (8) using width factors provided by (6).

The vector quantization process leading to values of $n(m)$ which are "locally constant", the hypothesis used to obtain the "optimal" value of the $h(m)$ width factor (equation 6) is still verified, even if the values of $n(m)$ are not "globally constant". But, as we will see in the following, the actual optimal value of $h(m)$ will also depend on the data space dimension and on the codebook size.

3.2. The optimal width factor

As said in section 2.2 the hypothesis leading to the "optimal" value of the $h(m)$ width factor (6) is that the number of centroids is sufficiently large so that the CL learning leads to clusters small enough in order to allow to approximate the true probability density inside each cluster by a constant.

On the other hand, as the codebook size decreases, the vector quantization will lead to larger clusters which do no more have the above mentioned property of being "small"; we can thus guess that (6) will be no more valid and that the optimal width factor $h(m)$ will decrease. In fact, if the codebook size exactly corresponds to the number of modes in the learning distribution the optimal value of $h(m)$ will corresponds to the maximum likelihood estimate of the standard deviation of an isotropic Gaussian centered on centroid $c(m)$ and modeling the mode of the distribution centered on $c(m)$ [4, 3]. This minimum value of the optimal $h(m)$ is linked to the averaged inertia coefficient $i(m)$ by:

$$h(m)_{min}{}^2 = \hat{\sigma}^2 = \frac{i(m)}{d} \tag{9}$$

So, depending on the codebook size, the width factor providing the best approximation will be

$$h(m)_{opt} = \gamma \sqrt{\frac{3}{2\ln 2}\,\frac{i(m)}{d}} \tag{10}$$

where γ is a multiplying factor depending on the codebook size, on the number of modes in the initial distribution and on the data space dimension (γ being egal to 0.6798 when $h(m)_{opt}$ egal $h(m)_{min}$ and to 1.0 when the codebook size become sufficient). Extended experiments [8] proved this: the optimal width factor found in simulations varies about from $\gamma = 0.7$ for a small number of clusters to $\gamma = 1$ for a large number of clusters. We can also mention that the optimal width factor is always obtained with γ closer from 0.7 than from 1 in large dimensions; this is due to the "empty space phonomenon" described in [5], by which the number of samples in large dimensions can always be considered as small.

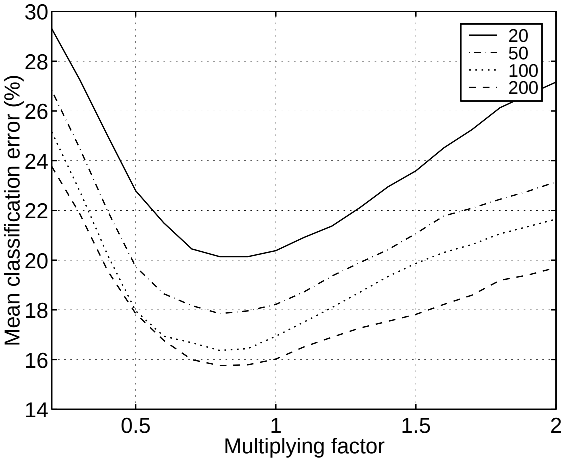

Fig. 2: Mean classification error on the "phoneme" database with 20, 50, 100 and 200 centroids.

3.3. A real-world problem

Tests have been carried out on a real-world classification database used in the European ROARS ESPRIT project [1]: "phoneme". Its aim is to distinguish between the classes of nasal and oral vowels. The database contains 5404 vowels coming from isolated syllables (for example: *pa, ta, pan,...*). Five different attributes characterize each vowel: the amplitudes of the five first harmonics, normalised by the total energy (integrated on all frequencies).

Simulations consisted in measuring the performances of the suboptimal Bayesian classifier (8) built with a total number of 20, 50, 100 or 200 clusters (for all classes together). The reported error percentages were obtained by a Averaged Holdout test method over five different partitions in a learnset and a testset of equivalent size (2702 patterns) and the Competitive Learning consisted of 10 presentations of the 2702 training patterns with the α learning factor linearly decreasing from 0.3 to 0.001. The errors were computed for a γ multiplying factor varying from 0.2 to 2; value 0.67 corresponds to the maximum likelihood estimate (9) and 1.0 to (6).

Figure 2 clearly shows a minimum in the value of the error for a multiplying factor $\gamma \simeq 0.8$. It is important to mention that a large number of simulations carried out on other databases showed similar qualitative results.

4. Conclusion

The use of kernel estimators with reduced design sets provided by vector quantization techniques enables to approach the Bayesian classification solution with a minimum amount of computations.

While the vector quantization process is deemed to have converged to centroids having the same distribution as the initial points, experiments showed that this process leads to clusters including different number of points. The solution we use to increase the quality of approximation is to take into account the number of points associated to each cluster.

Another problem is the evaluation of the appropriate optimal widths factors for the kernels used in neural networks which estimate the probability density functions; in this paper, we proposed the use of a γ multiplying factor which could take into account the effects of the data space dimension, of the codebook size and of the particularities of the distributions to approximate. With the hypothesis of small clusters (verified with large codebooks in small dimensions), γ is close from 1, which can be seen as an experimental proof of the hypothesis used in [7, 9]. When the codebook size decreases, γ decreases too, what confirms the results of [4] when the number of classes decreases to reach the number of modes of the distribution. The experiments presented in this paper may thus be seen as an unified way to present the optimal width kernel factors of radial Gaussian kernel estimators, depending on the hypotheses on the size of the clusters and the dimension of the space.

5. Acknowledgments

Part of this work has been funded by the ESPRIT-BRA project 6891, ELENA-Nerves II, supported by the Commission of the European Communities (DG XIII). All simulations where run on the Packlib simulator developed at EPFL (Lausanne, Switzerland) in the framework of the ELENA project.

References

[1] P. Alinat, "Periodic Progress Report 4," Tech. Rep., ROARS Project ESPRIT II- Number 5516, February 1993. Thomson report TS. ASM 93/S/EGS/NC/079.

[2] T. Cacoullos, "Estimation of a multivariate density," *Annals of Inst. Stat. Math.*, vol. 18, pp. 178–189, 1966.

[3] P. Comon, "Supervised classification: a probabilistic approach," in *ESANN95-European Symposium on Artificial Neural Networks*, (M. Verleysen, ed.), (Brussels, Belgium), D facto publications, April 1995.

[4] P. Comon, G. Bienvenu, and T. Lefebvre, "Supervised design of optimal receivers," in *NATO Advanced Study Institute on Acoustic Signal Processing and Ocean Exploration*, (Madeira, Portugal), July 26-Aug. 7 1992.

[5] P. Comon, J. Voz, and M. Verleysen, "Estimation of performance bounds in supervised classification," in *ESANN94-European Symposium on Artificial Neural Networks*, (M. Verleysen, ed.), (Brussels, Belgium), pp. 37–42, D facto publications, April 1994.

[6] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. 28, pp. 84–95, January 1980.

[7] J. Voz, M. Verleysen, P. Thissen, and J. Legat, "Handwritten digit recognition by suboptimal bayesian classifier," in *Neural Networks and their applications 94*, (Marseille), p. , IUSPIM, December,15-16 1994.

[8] J. Voz, M. Verleysen, P. Thissen, and J. Legat, "A practical view of suboptimal bayesian classification," in *IWANN95-Proceedings of the International Workshop on Artificial Neural Networks*, (P. Mira, Cabestany, ed.), (Malaga, Spain), Springer-Verlag Lecture Notes in Computer Sciences, June 1995. Submitted.

[9] J. Voz, M. Verleysen, P. Thissen, and J. Legat, "Suboptimal bayesian classification by vector quantization with small clusters," in *ESANN95-European Symposium on Artificial Neural Networks*, (M. Verleysen, ed.), (Brussels, Belgium), D facto publications, April 1995. To be published.