HIGHER ORDER AUTOCORRELATIONS FOR PATTERN CLASSIFICATION

Vlad Popovici and Jean-Philippe Thiran

Swiss Federal Institute of Technology (EPFL)
Signal Processing Laboratory (LTS)
CH-1015 Lausanne, Switzerland
WWW home page: http://ltswww.epfl.ch
{vlad.popovici,jp.thiran}@epfl.ch

ABSTRACT

The use of higher-order local autocorrelations as features for pattern recognition has been acknowledged since many years, but their applicability was restricted to relatively low orders (2 or 3) and small local neighborhoods, due to combinatorial increase in computational costs.
In this paper a new method for using these features is presented, which allows the use of autocorrelations of any order and of larger neighborhoods. The method is closely related to the classifier used, a Support Vector Machine (SVM), and exploits the special form of the inner products of autocorrelations and the properties of some kernel functions used by SVMs. Using SVM, linear and non-linear classification functions can be learned, extending the previous works on higher-order autocorrelations which were based on linear classifiers.

1. INTRODUCTION

In most pattern recognition problems, each pattern can be described as a scalar function of time or spatial coordinates. In many cases, a translation or a scale change has no effect on class membership. Regarding each pattern as a point in a vector space, we wish to map all points corresponding to translated (or scaled) versions of one pattern in a single point. In addition, patterns which differ in other ways should map into distinct points, and in some sense, patterns which are similar should map into points that are close together.

The higher-order measures possess the uniqueness property for even orders [1] and they are shift-invariant. Higher-order autocorrelations have been previously used as features describing patterns [2], [3], [4], [5], but their applicability has been limited to second or third orders and a small local neighborhood, due to high computational costs. As noted in [5], for a second-order autocorrelation function and exploiting its symmetry, one gets 29 features for a 3×3 neighborhood and 205 features for a 5×5 neighborhood. Also, in all reported experiments, linear classifiers have been employed.

In this paper we propose a new method for combining higher-order autocorrelation functions and non-linear classifiers which is no longer limited to the second or third order and which can be applied on larger neighborhoods. This method relies on exploiting some properties of the inner products of autocorrelation functions which, in turn, allow us to avoid explicitly computing the autocorrelations. The paper is organized as follows: in section 2 we describe the autocorrelation functions and their inner product, section 3 describes briefly the classifier and finally, some experimental result and conclusions are presented in last sections.

2. FEATURE EXTRACTION BY LOCAL HIGH-ORDER STATISTICS

High-order statistics (HOS) measures are extensions of second-order measures (such as the autocorrelation function) to higher orders. Given a real-valued function $x(t)$, its n-th order autocorrelation function with displacements is defined by

$$r_x^{(n)}(\tau_1, \tau_2, \ldots, \tau_n) = \int x(t)\, x(t+\tau_1) \ldots x(t+\tau_n)\, dt \tag{1}$$

where $\tau_1, \ldots, \tau_n$ are displacements within a given neighborhood. This function, $r_x^{(n)}$, is shift-invariant, in the sense that $x(t)$ and $y(t) = x(t+\tau)$ have the same n-th order autocorrelation function. A detailed analysis of the properties of the autocorrelation function is presented in [1]. Here, we are interested in the inner product of two n-th order autocorrelations. Let $r_x^{(n)}$ and $r_y^{(n)}$ be the n-th order autocorrelations of patterns $x(t)$ and $y(t)$. Following [1], their inner product is given by

The work was carried out as part of the European Union IST Project 1999-11159, BANCA

$$
\begin{aligned}
&\langle r_x^{(n)},\, r_y^{(n)} \rangle = \\
&= \int \dots \int r_x^{(n)}(\tau_1, \dots, \tau_n) \cdot r_y^{(n)}(\tau_1, \dots, \tau_n)\, d\tau_1 \dots d\tau_n \\
&= \int \dots \int \left\{ \int x(t)\, x(t+\tau_1) \dots x(t+\tau_n)\, du \right\} \\
&\cdot \left\{ \int y(u)\, y(u+\tau_1) \dots y(u+\tau_n)\, du \right\} d\tau_1 \dots d\tau_n \\
&= \int \int x(t)\, y(u) \left\{ \int x(t+\tau)\, y(u+\tau) d\tau \right\}^n du\, dt \\
&= \int \int x(t)\, y(s+t) \left\{ \int x(v)\, y(v+s) dv \right\}^n ds\, dt \\
&= \int \left\{ \int x(v)\, y(v+s) dv \right\}^{n+1} ds
\end{aligned}
\tag{2}
$$

Combining autocorrelation of different orders in order to obtain a more descriptive feature vector can be done as follows. Let $I = \{i_1, \dots, i_m\}$ be a set of indices and let $R_x^{(I)}$ be the vector obtained by concatenating the autocorrelations $r_x^{(k)}$, where $k = i_1, \dots, i_m$, then it is obvious that

$$
\left\langle R_x^{(I)}, R_y^{(I)} \right\rangle = \sum_{k=i_1,\dots,i_m} \left\langle r_x^{(k)}, r_y^{(k)} \right\rangle \tag{3}
$$

meaning that computing the inner product of two compound feature vectors can be done by simply summing the inner products of the components.

3. SUPPORT VECTOR MACHINES FOR PATTERN CLASSIFICATION

Below we will briefly present the Support Vector Machines (SVMs) and we will point out their advatages in the case of high-order autocorrelations. For a more comprehensive description, the reader is referred to [6], [7]. The task of learning from examples, for a two-class pattern recognition problem, can be formulated as follows: given a set of functions

$$
\{f_\alpha\}_{\alpha \in \Lambda} \ , \ f_\alpha : \mathbb{R}^n \to \{-1, +1\}
$$

and a set of examples

$$
\{(\mathbf{x}_i, y_i)\}_{i=1}^l \ \subset \mathbb{R}^n \times \{-1, +1\},
$$

each one generated according to an unknown probability distribution function $P(\mathbf{x}, y)$, we want to find a function f_{α^*} which minimizes the risk of misclassification of the new patterns randomly drawn from P, given by the risk functional:

$$
R(\alpha) = \frac{1}{2} \int |f_\alpha(\mathbf{x}) - y|\, dP(\mathbf{x}, y)
$$

For any $\eta \in [0, 1]$, the following inequality holds with probability $1 - \eta$:

$$
R(\alpha) \le R_{emp}(\alpha) + \sqrt{\frac{h\left(\log \frac{2l}{h} + 1\right) - \log \frac{\eta}{4}}{l}} \tag{4}
$$

providing an upper bound on the risk functional, where $R_{emp}(\alpha)$ is the empirical risk and h is the Vapnik-Chervonenkis (VC) dimension. The second term on the right hand side of (4) is called the VC confidence. The goal is to minimize the risk functional by minimizing its upper bound. There are two strategies: keep the VC confidence fixed and minimize the empirical risk, or fix the empirical risk to a small value and minimize the VC confidence. The SVMs are taking the second approach and we will briefly describe the method below.

In the linear separable case we are looking for an *optimal hyperplane* $\langle \mathbf{w}, \mathbf{x} \rangle + b = 0$ which minimizes the VC confidence while providing the best generalization capabilities. The decision function can be written as

$$
f(\mathbf{x}) = \operatorname{sgn}\left(\langle \mathbf{w}, \mathbf{x} \rangle + b\right) \tag{5}
$$

where by $\langle \cdot, \cdot \rangle$ we denote the inner product of two vectors. Geometrically, the problem to be solved resides in finding the hyperplane that maximizes the sum of distances to the closest positive and negative training examples. This distance is called *margin* and the optimal hyperplane is obtained by minimizing $\|\mathbf{w}\|^2$ subject to a set of constraints. By relaxing the constraints, the non-separable case can also be handled and the optimization problem can be stated as minimize

$$
\Phi(\mathbf{w}, \xi) = \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{l} \xi_i \tag{6}
$$

subject to the following constraints

$$
\begin{aligned}
\xi_i &\ge 0, \quad i = 1, \dots, l \\
y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) &\ge 1 - \xi_i, \quad i = 1, \dots, l
\end{aligned}
\tag{7}
$$

where C is a user-defined constant and $\sum_{i=1}^l \xi_i$ is the upper bound on the number of misclassifications on the training set. Introducing the Lagrange multipliers $\alpha_i \ge 0$ and using Kuhn-Tucker theorem, the solution can be expressed as

$$\mathbf{w} = \sum_{i=1}^{l} y_i \alpha_i \mathbf{x}_i, \tag{8}$$

with $\alpha_i > 0$ corresponding to those examples $(\mathbf{x}_i, y_i)$ that strictly obey (7). These $\mathbf{x}_i$ are called *support vectors*. The coefficients α_i are the solutions of the following quadratic programming problem: maximize

$$W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \tag{9}$$

subject to

$$0 \leq \alpha_i \leq C, \quad i = 1, \ldots, l, \text{and} \sum_{i=1}^{l} \alpha_i y_i = 0 \tag{10}$$

Then, the decision function becomes

$$f(\mathbf{x}) = \text{sgn}\left(\sum y_i \alpha_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b \right) \tag{11}$$

where the sum is taken over all support vectors. One thing must be noted here: the only way data appears in above equations ((9)-(11)) is in the form of inner products.

The problem of non-linear decision boundaries is solved by mapping the training set into a high-dimensional (possible infinite-dimensional) feature space where the training will be carried out as in the linear case. This mapping is done by means of *kernel functions* - functions that define an inner product in the feature space. Let $K(\mathbf{x}_i, \mathbf{x}_j)$ be such a kernel function. All the above considerations can be extended to the case of kernel functions and in the equations (9)-(11) the inner products of the form $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ will be replaced by $K(\mathbf{x}_i, \mathbf{x}_j)$.

This means that if there would exist a kernel function that could be expressed as a function of inner products of data, we could use (2) and (3) for computing the kernel values in the case of autocorrelation features, avoiding the extremely expansive task of explicitly computing the autocorrelations.

This kind of kernel exists and two common examples of non-linear ones are the polynomial and the sigmoidal kernels which could be written as:

$$K_P(\mathbf{x}_i, \mathbf{x}_j) = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + 1)^p \tag{12}$$

and

$$K_S(\mathbf{x}_i, \mathbf{x}_j) = \tanh\left(\kappa \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \delta\right) \tag{13}$$

Considering $\mathbf{x}_i$ as compound autocorrelation-feature vectors, we obtain the corresponding kernels which allow non-linear classifiers to be trained.

4. EXPERIMENTS

In this section some experiments and preliminary results are presented. The goal of the experiments was to prove the possibility of using higher order autocorrelation functions for pattern classification.

The patterns used for training/testing the classifiers were human faces and the kernel used was a second degree polynomial kernel (12). The training data set was the XM2VTSDB database ([8]) from which 206 images were used for training and 30 for testing. The negative examples (263 for training and 40 for testing) were randomly generated from images containing no human faces. As we were interested in studying the influence of using the higher orders of autocorrelations, we did not tune the training parameters (polynomial degree, the constant C) and the training has been done in a single step.

Using these severe constraints, the system was trained to classify human faces. The following table summarizes the preliminary results:

Autocorrelation orders	Neighborhood size			
	3×3	5×5	7×7	9×9
1	30	32.86	31.43	38.57
1,2	27.15	32.86	37.14	31.43
1,2,3	15.71	28.57	41.43	32.86
1,2,3,4	11.43	25.71	27.14	30
1,2,3,4,5	10	20	22.86	25.43
1,2,3,4,5,6	8.57	14.29	18.57	15.71

Table 1. Misclassification rates (%)

It must be noted that the images have not been preprocessed - an increase in detection rates is expected in the case of applying some specific preprocessing (histogram equalization, removing the background by applying a mask). Also, the training has to be extended by using bootstrapping techniques for generating more significant negative examples.

In the final paper, more detailed examples will be presented.

5. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a new method for using higher order autocorrelations for pattern classification which is able to deal with any autocorrelation order and with significantly larger neighborhoods. Also, both linear and non-linear classifiers can be trained by means of different ker-

nels, as long as the kernel function can be expressed in terms of inner products of the input data.

Due to the fact that autocorrelations are shift-invariant, a preprocessing step consisting in a log-polar transform ([3],[4]) would make the classifier to be rotation-invariant. Moreover, by combining this preprocessing with a multi-scale approach, we expect to obtain a robust pattern classifier with a high degree of rotation/scaling invariance.

Acknoledgments

The first author thanks to Genevieve Dardier for valuable discussions and helpful comments.

6. REFERENCES

[1] J. A. McLaughlin and J. Raviv, "Nth-order autocorrelations in pattern recognition," *Information and Control*, vol. 12, pp. 121 – 142, 1968.

[2] T.M. Breuel, "Higher-order statistics in visual object recognition," Memo 93-02, IDIAP, June 1993.

[3] K. Hotta, T. Kurita, and T. Mishima, "Scale invariant face detection method using higher-order local autocorrelation features extracted from log-polar image," in *Automatic Face and Gesture Recognition. Proceedings. Third IEEE International Conference on*. 1998, pp. 70–75, IEEE.

[4] T. Kurita, K. Hotta, and T. Mishima, "Scale and rotation invariant recognition method using higher-order local autocorrelation features of log-polar image," in *Third Asian Conference on Computer Vision*, 1998, pp. 89–96.

[5] M. Kreutz, B. Volpel, and H. Janssen, "Scale-invariant image recognition based on higher order autocorrelation features," *Pattern Recognition*, vol. 29, no. 1, 1996.

[6] Vladimir N. Vapnik, *The Nature of Statistical Learning Theory*, Springer Verlag, 1995.

[7] Nello Cristianini and John Shawe-Taylor, *Support Vector Machines*, Cambridge University Press, 2000.

[8] K. Messer, J. Matas, J. Kittler, J. Luettin, and G. Maitre, "Xm2vtsdb: The extended m2vts database," in *AVBPA'99*, S. Akunuri and C. Kullman, Eds., 1999, pp. 72–77.

[9] I. Sekita, T. Kurita, and N. Otsu, "Complex autoregressive model and its properties," Tech. Rep., Electrotechnical Laboratory, 1991.