EXTRACTION OF ATTRIBUTES FOR VISUAL OBJECT RECOGNITION AND DNA MICROARRAY ANALYSIS

Sébastien Gadat SEBASTIEN.GADAT@CMLA.ENS-CACHAN.FR

ENS Cachan, CMLA
61, avenue du Président Wilson, 94235 Cachan Cedex, FRANCE

ABSTRACT

We introduce a new model formalizing selection of features from a large dictionary of variables that can be computed from a signal or an image. Features are extracted according to an efficiency criterion, on the basis of specified classification tasks. We estimate a probability distribution $\mathbb{P}$ on the complete dictionary, which distributes its mass over the more efficient or informative components. The method is then tested on several problems of signal processing like face detection, handwritten digit recognition and DNA microarray analysis.

1. INTRODUCTION

In many recent instances of signal classification problems (biology microarrays analysis, text interpretation, image classification), scientists have to deal with highly complex datasets involving a large number of variables. For many reasons, this abundance of variables harms recognition tasks: meaningless features act as artificial noise in data and limit the accuracy of classification algorithms. Moreover, many algorithmic or statistic reasons make essential to reduce dimension of the feature space (variance reduction to infer reliable conclusions (Bias-Variance dilema [1]), speed of classification algorithms like Support Vector Machines [2] or k-nearest neighbors). Also, there are other applications for which detecting the relevant explanatory variables is critical, and as important as correctly performing classification tasks. This is the case, for example, in biology, where describing the source of a pathological state is equally important to just detecting it [3].

Methods like singular value decomposition, or independant component analysis do not always yield relevant selection of variables, model selection methods based on penalization face a combinatorial challenge when the set of variables has no specific order and the search must be done over its subsets. Automatic feature space construction and variable selection from a large set have thus become an active research area using tree-structured classifiers [4, 5] or optimizing margin of SVM [6] for instance.

We formalize the feature selection problem as the design of a suitable probability measure on a set of admissible variables, and provide a learning algorithm for estimating this probability on a training set expecting that the most likely variables (according to this probability) will be the most useful. This estimated probability on the feature set will also be used to generate randomized classifiers The selection algorithm and the randomized classifier will be tested on a series of examples. The work is organized as follows. In section 2, we describe our feature extraction model and design a probability distribution over the feature space that permit to extract variables. Next, in sections 3, we define a stochastic adaptive algorithm to find an optimal weight distribution over features which solves the minimization problem infered from our model. We then provide several applications of our method, first with synthetic data, then on image classification and spam detection.

2. FEATURE EXTRACTION MODEL

We follow the general framework of supervised statistical pattern recognition: the input signal I belongs to $\mathcal{I}$ and is a realization of a random variable. Several computable quantities are accessible on I, forming a complete dictionary (set of features) denoted $\mathcal{F} = \{\delta_1, \dots \delta_f\}$. This set is assumed to be finite, although f can (and will) be a very large number, and our goal is to select the most useful variables. A classification task is then specified by a finite partition $\mathcal{C} = \{C_1, \dots C_N\}$ of $\mathcal{I}$; the goal is to predict the class $\mathcal{C}(I)$ from the observation $\delta(I)$.

We assume that a classification algorithm $\mathbb{A}$, is available, for both training and testing. We assume that $\mathbb{A}$ is adapted to a subset $\omega \subset \mathcal{F}$ of *active variables* taken in the dictionary $\mathcal{F}$. In training mode, $\mathbb{A}$ uses a database to build an optimal classifier $\mathbb{A}_\omega : \mathcal{I} \to \mathcal{C}$. The test mode simply consists in the application of $\mathbb{A}_\omega$ on a given signal. In the training phase, we first extract $\{\omega^{(1)}, \dots, \omega^{(n)}\}$, subsets of $\mathcal{F}$, and build the classifiers $\mathbb{A}_{\omega^{(1)}}, \dots, \mathbb{A}_{\omega^{(n)}}$. Then, perform classification in test phase using a majority rule for these n classifiers to obtain the final algorithm $\bar{\mathbb{A}}_{\mathcal{C}} = \bar{\mathbb{A}}(\omega^{(1)}, \dots, \omega^{(n)}, \mathcal{C})$. Note that we are not studying classifi-

cation algorithms, $\mathbb{A}$, for which we use standard procedures; rather, we focus on the extraction mechanism creating the random subsets of $\mathcal{F}$. This randomization process depends on the way variables are sampled from $\mathcal{F}$, and we will see that the design of a suitable probability on $\mathcal{F}$ for this purpose can significantly improve the classification rates. This probability therefore comes as a new parameter and will be learned from the training set.

The algorithm $\mathbb{A}$ provides a different classifier $\mathbb{A}_\omega$ for each choice of a subset $\omega \subset \mathcal{F}$. We let q be the classification error: $q(\omega, \mathcal{C}) = \mathbf{P}(\mathbb{A}_\omega(I) \neq \mathcal{C}(I))$ which will be estimated by

$$\hat{q}(\omega, \mathcal{C}) = \hat{\mathbf{P}}(\mathbb{A}_\omega(I) \neq \mathcal{C}(I)) \tag{1}$$

where $\hat{\mathbf{P}}$ is the empirical probability on the training set. We shall consider two particular cases for a given $\mathbb{A}$:

- Multi-class algorithm: $\mathbb{A}$ is naturally adapted to multi-class problems and we then let $g(\omega) = \hat{q}(\omega, \mathcal{C})$.
- Two-class algorithms (like support vector machines) Denote by $\mathcal{C}_i$ the binary partition $\{C_i,\ \mathcal{I} \setminus C_i\}$. We then denote:

$$g(\omega) = \frac{1}{N} \sum_{i=1}^{N} \hat{q}(\omega, \mathcal{C}_i)$$

Since the evaluation of $\hat{q}$ is needed to compute g, we use the following procedure to infer an approximation of g:

- Sample a subset T_1 from the training set.
- Learn the classification algorithm on the basis of ω and T_1.
- Sample, with the same procedure a subset T_2 from the training set and compute the empirical error of the classifier on T_2.

Consider now a probability distribution $\mathbb{P}$ on $\mathcal{F}$. For an integer k, consider the distribution $\mathbb{P}_k = \mathbb{P}^{\otimes k}$ corresponding to k independent trials with distribution $\mathbb{P}$ and define the cost function $\mathcal{E}$ by

$$\mathcal{E}(\mathbb{P}) = \mathbb{E}_{\mathbb{P}_k} g(\omega) = \sum_{\omega \in \mathcal{F}^k} g(\omega) \mathbb{P}_k(\omega). \tag{2}$$

We wish to minimize this averaged error rate with respect to the selection parameter $\mathbb{P}$ and thus select the relevant features (those where $\mathbb{P}(\delta)$ is large). The global scheme that we propose for estimating $\mathbb{P}$ is summarized in Figure 1:

3. SEARCH ALGORITHMS

We minimize the energy $\mathcal{E}$ with respect to the probability $\mathbb{P}$. This requires computing the gradient of $\mathcal{E}$, and

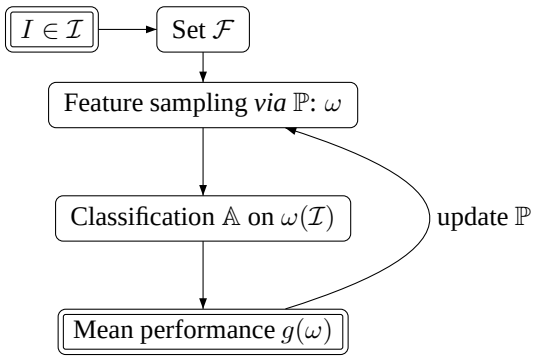

Fig. 1. Procedure for learning the probability $\mathbb{P}$

dealing with the constraints called $\mathcal{S}_\mathcal{F}$:

$$\sum_{\delta \in \mathcal{F}} \mathbb{P}(\delta) = 1 \tag{3}$$

$$\forall \delta \in \mathcal{F} \quad \mathbb{P}(\delta) \geq 0 \tag{4}$$

We will also denote $\mathcal{H}_\mathcal{F}$ the hyperplane in $\mathbb{R}^f$ which carries $\mathcal{S}_\mathcal{F}$, defined by (3) and the Euclidean projection $\pi_{\mathcal{H}_\mathcal{F}}$ on $\mathcal{H}_\mathcal{F}$ We use first a gradient descent to minimize $\mathcal{E}$, taking only constraint (3) into account. For the Euclidean metric:

$$\forall \delta \in \mathcal{F} \qquad \nabla_\mathbb{P} \mathcal{E}(\delta) = \sum_{\omega \in \mathcal{F}^k} \frac{C(\omega, \delta) \mathbb{P}_k(\omega)}{\mathbb{P}(\delta)} g(\omega)$$

where $C(\omega, \delta)$ is the number of occurences of δ in ω and the corresponding discrete scheme is

$$\mathbb{P}_{n+1} = \mathbb{P}_n - \epsilon_n \pi_{\mathcal{H}_\mathcal{F}} \left(\nabla \mathcal{E}(\mathbb{P}_n) \right) \quad \epsilon_n > 0 \tag{5}$$

We do not deal for the moment with constraints (4). It is all the same possible by switching to an exponential parametrization (6) or a constrained optimization algorithm (7).

For any $\delta \in \mathcal{F}$, define $y(\delta) = \log \mathbb{P}(\delta)$ and

$$\mathcal{Y} = \left\{ y = (y(\delta), \delta \in \mathcal{F}) \mid \sum_{\delta \in \mathcal{F}} e^{y(\delta)} = 1 \right\}$$

which is in one-to-one correspondence with $\mathcal{S}_\mathcal{F}$. If we call $\tilde{\mathcal{E}}(y) = \mathcal{E}(\mathbb{P})$, we can consider the gradient of $\tilde{\mathcal{E}}$ as a gradient on the variables $\mathbb{P}$ with the Kullback metric $\langle ; \rangle_\mathbb{P}$ on $\mathcal{S}_\mathcal{F}$, and denoting $\tilde{\nabla}$ the gradient with respect to $\langle ; \rangle_\mathbb{P}$, we have

$$\tilde{\nabla}_\mathbb{P} \mathcal{E}(\delta) = \nabla_y \tilde{\mathcal{E}}(\delta) = \sum_{\omega \in \mathcal{F}^k} \mathbb{P}_k(\omega) C(\omega, \delta) g(\omega)$$

The associated discrete evolution equation for $\mathbb{P}$ becomes

$$\mathbb{P}_{n+1}(\delta) = \frac{\mathbb{P}_n(\delta)}{K_n} e^{-\epsilon_n \left(\tilde{\nabla}_{\mathbb{P}_n} \mathcal{E}(\delta) - \kappa_n \mathbb{P}_n(\delta) \right)} \tag{6}$$

where K_n is second order and κ_n a projection term on $\mathcal{Y}$.

Learning $\mathbb{P}$ with (5) can also be adapted to naturally satisfy (4). We use a constrained diffusion to $\mathcal{S}_{\mathcal{F}}$

$$\mathrm{d}\mathbb{P}_t = -\nabla_{\mathbb{P}_t}\mathcal{E}(\mathbb{P}_t)\mathrm{d}t + \sqrt{\sigma}\mathrm{d}W_t + \mathrm{d}Z_t \qquad (7)$$

where σ is a non degenerate positive matrix on $\mathcal{H}_{\mathcal{F}}$, $\mathrm{d}W_t$ a standard gaussian increment and $\mathrm{d}Z_t$ a process which acounts for the jumps which appear when a reprojection is needed on $\mathcal{S}_{\mathcal{F}}$: $\mathrm{d}|Z_t|$ is positive if and only if $\mathbb{P}_t$ hits the boundary $\partial\mathcal{S}_{\mathcal{F}}$. Solutions to (7) (which is is well posed in our situation) depends on Skorokhod maps [7].

Since gradients are numerically untractables, we use a stochastic approximation based on the ODE method [8] to learn $\mathbb{P}$ with (5,6,7). The stochastic approach can be easily computed if we remark that the expression of $\nabla\mathcal{E}$ is an expectation

$$\nabla_{\mathbb{P}}\mathcal{E}(\delta) = \mathbb{E}\left[\frac{g(\omega)C(\omega,\delta)}{\mathbb{P}(\delta)}|\omega \sim \mathbb{P}_k\right]$$

It is now possible to explicit the learning algorithm which acts asymptotically as a similar way as (7) [9, 10]:

- Step 0: initialize $\mathbb{P}_0$ to the uniform law on $\mathcal{F}$.
- Step n: extract ω_n w.r.t. $\mathbb{P}_{n,k}$ and compute $g(\omega_n)$
- Update $\mathbb{P}_{n+1}$ using the formula:

$$\mathbb{P}_{n+1} = \mathbb{P}_n - \epsilon_n d_n + \sqrt{\epsilon_n}d\xi_n + dz_n$$

where

$$d_n(\delta) = \pi_{\mathcal{H}_{\mathcal{F}}}\left(\frac{g(\omega_n)C(\omega_n,\delta)}{\mathbb{P}_n(\delta)}\right)$$

and dz_n is the shortest vector to keep $\mathbb{P}_{n+1}$ into the simplex $\mathcal{S}_{\mathcal{F}}$.

4. EXPERIMENTS

Synthetic datas We consider 100 ternary variables and 3 classes and $\mathcal{I} = \{-1;0;1\}^f$ (features are coordinates of $I \in \mathcal{I}$). We define $\mu(\,;\mathcal{G})$ a probability for which all δ in $\mathcal{F}$ are independent, $\delta(I)$ follows a uniform distribution on $\{-1;0;1\}$ if $\delta \notin \mathcal{G}$ and $\delta(I) = 1$ if $\delta \in \mathcal{G}$. We take $\mathcal{F}_1 = \{\delta_1;\delta_3;\delta_5;\delta_7\}, \mathcal{F}_2 = \{\delta_2;\delta_4;\delta_6;\delta_8\}$ and $\mathcal{F}_3 = \{\delta_1;\delta_4;\delta_8;\delta_9\}$ and noise uniformly the distributions obtained. $\mathbb{A}$ is a M-nearest neighbour, with distance given by

$$d(I_1,I_2) = \sum_{\delta\in\omega}\chi_{\delta_1(I)\neq\delta_2(I)}$$

We compare the previous algorithms (figure 2). Exact algorithm is faster but is quiclky captured in a local minimum although stochastic algorithms avoid more traps. The stochastic Euclidean method and reflected diffusion achieve better results faster. Moreover, we observe that the features which are preferably selected are those which lie in several subspaces $\mathcal{F}_i$ and are *reusable features*, the knowledge of which being very precious information for the understanding of pattern recognition problems.

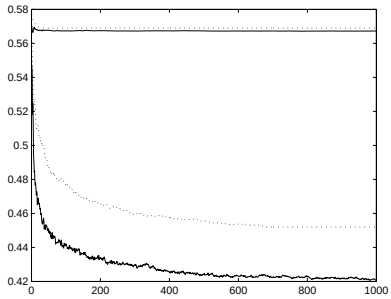

Fig. 2. Exact gradient (top full line) vs. exponential gradient (top dashed line) vs. Euclidean stochastic gradient (bottom dashed line) vs. Reflected diffusion (bottom full line)

Face detection We use in this section the face database from MIT, which contains 19×19 gray level images. $\mathcal{F}$ is a set of binary edge detectors, as developed in [4] and the classification algorithm $\mathbb{A}$ which is used here is a Support Vector Machine. We summarize results in figure 3.

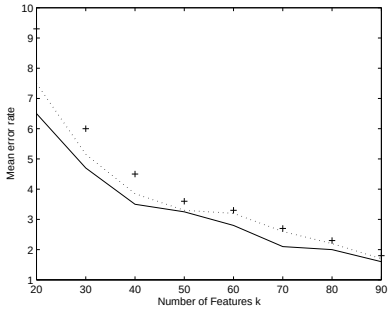

Fig. 3. Average classification error of faces recognition on the test set using a uniform law (crossed line) and $\mathbb{P}_\infty$, learned with a Riemannian stochastic gradient method (dashed line) or reflected diffusion (full line).

The results are quite good since we achieve a performance of 1.6% error rate after learning with a reflected diffusion or 1.7% with a stochastic exponential gradient (1.95%, before learning). Our features extraction method based on learning the distribution $\mathbb{P}$ thus improve significantly the performances of classification, particularly for weak classifiers ($k = 20$ or 30 for example) as shown in figure 3. We note again that reflected diffusion performs better than the exponential gradient. The analysis of the most likely features is also interesting and occurs in meaningful positions

since favored edge detectors are located near the nose, eyes, and mouth of faces.

DNA Microarray Analysis We benchmark our selection of features on the standard Leukemia cancer dataset downloaded from http://www.iitk.ac.in/kangal/bioinformatics. Datas are preprocessed and transformed to a collection of 3859 genes of 72 leukemia samples. Datas are divided in 47 samples if Acute Lymphoblastic Leukemia (ALL) and 25 samples of Acute Myeloblastic Leukemia (AML). We divide samples in a training set (60% of datas) and a test set (40%) which is independent from the training set. We use for $\mathbb{A}$ a SVM classifier and $k = 100$ genes are extracted at each step. We show in figure 4 the evolution of the mean error rate $\mathcal{E}$ on our training set.

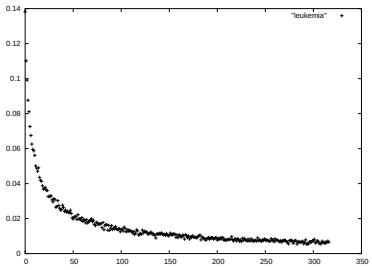

Fig. 4. Evolution of $\mathcal{E}$ for the cancer classification problem (AMM vs. AML) on the training set using a reflected diffusion (full line).

We achieve a very low rate of error on our training set (0.5% mean misclassification rate with $\mathbb{P}_\infty$) and the probability map permit to exhibit important genes for the classification problem. We then rank genes by decreasing probability obtained after our learning procedure and estimate global misclassification rate on our test set using the top ranked genes without any votind procedure (the performances reported are individual performances of $\mathbb{A}$ run on the space determined by the most selected genes). Figure 5 shows the error obtained with the number of genes selected. We ob-

Number of genes	$2 \mapsto 19$	20
Number of misclassified samples	1	0

Fig. 5. Number of misclassified samples of the test set using several numbers of selected genes

tain similar results as [3] since we only need 20 genes to perfectly classify datas. Our method of selection of genes is higly effective since the mean misclassification rate using 20 genes randomly selected among the global set of variables is greater than 30% and outperform results provided by [11] (15% error rate with 50 genes). Our results are nearly similar to those reported in [12]: by using a 10-fold cross validation method, we obtain a very low rate of error with 4 genes (5% error) and a perfect classification with 20 genes. Moreover, the genes selected by our algorithm are consistent with some genes selected in other works [13, 11].

Handwritten number recognition We have tested our algorithm on the US Postal database: the images are 16×16 grayscaled maps, with intensity between 0 and 255. We use the same feature set, $\mathcal{F}$, as in the faces example. The improvement of the detection rate is also similar to the previous example, as shown in figure 6.

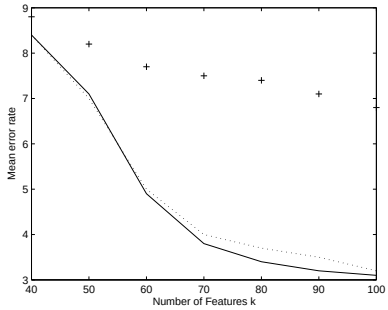

Fig. 6. Mean error rate for uniform law (crossed line) vs. exponential descent (dashed line) vs. reflected diffusion (full line) on the test set.

Results noticed in the figure 6 are obtained using a majority vote with 10 binary SVM-classifiers on each binary classification problem C_i vs. $I \setminus C_i$ and learning a probability distribution to extract features improves significantly error rates. The final average error rate on the US Postal database is about 3.2% for 10 elementary classifiers per class $\mathcal{C}_i$, with 100 binary features per elementary classifier. The performance is not as good as the one obtained by the tangent distance method (2.7% error rate of classification), but we here use very simple (edge) features, and is better than linear or polynomial Support Vector Machines runned on whole pixels (8.9% and 4% error rate) and than sigmoid kernels (4.1% [14]) with a reduced complexity (measured, for example by the needed amount of memory).

5. CONCLUSION, FUTURE WORK

We introduce a precise mathematical model to formalize the search for optimal features. We learn a probability distribution on the original dictionary, based on a gradient descent of an energy $\mathcal{E}$ within the simplex $\mathcal{S}_\mathcal{F}$. The experiments show that the performance is significantly improved over an initial rule in which features are simply uniformly distributed. In a future work, $\mathcal{F}$ can be modified during the algorithm, by allowing for combination rules, involving a hybrid evolution in the set of probability measures. This will be implemented as a generalization of our constrained diffusion algorithm to include jumps in the underlying feature space.

6. REFERENCES

[1] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma," *Neural Computation*, vol. 4, pp. 1–58, 1992.

[2] V. Vapnik, *Statistical learning theory*, John Wiley & Sons Inc., 1998.

[3] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46, pp. 389–422, 2002.

[4] Y. Amit and D. Geman, "A computational model for visual selection," *Neural computation*, vol. 11, pp. 1691 – 1715, 1999.

[5] L Breiman, "Arcing classifiers," *Ann. Statist.*, vol. 26, no. 3, pp. 801–849, 1998.

[6] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Machine Learning*, vol. 46, no. 1–3, pp. 131–159, 2002.

[7] P. Dupuis and H. Ishii, "On Lipschitz continuity of the solution mapping to the Skorokhod problem, with applications," *Stochastics Stochastics Rep.*, vol. 35, no. 1, pp. 31–62, 1991.

[8] H. Kushner and G. Yin, *Stochastic approximation and recursive algorithms and applications*, Springer-Verlag, 2003.

[9] S. Gadat, "Apprentissage d'un vocabulaire symbolique pour la détéction d'objets dans une image," *Thèse de l'École Normale Supérieure de Cachan*, 2004.

[10] Sebastien Gadat and Laurent Younes, "A stochastic algorithm of features extraction for pattern recognition," *Preprint CMLA*, p. 22, 2004.

[11] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Collier, M.L. Loh, J.R. Downing, M. A. Caligiuri, and C. D. Bloomfieldand E. S. Lander, "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring.," *Science*, vol. 286, pp. 531–537, 1999.

[12] D. Geman, C. d' Avignon, D. Naiman, and R. Winslow, "Classifying gene expression profiles from pairwise mrna comparisons," *Statistical Applications in Genetics and Molecular Biology*, , no. 3, 2004.

[13] Kalyanmoy Deb and Raji Reddy, "Classification of two-class cancer data reliably using evolutionary algorithms," *Technical Report*, 2003.

[14] B. Schölkopf and A.J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press, Cambridge, MA, 2002.