MULTI-CLASS LEAST SQUARES CLASSIFICATION AT BINARY-CLASSIFICATION COMPLEXITY

Zineb Noumir, Paul Honeine

Institut Charles Delaunay (CNRS), LM2S
Université de technologie de Troyes
10010 Troyes, France

Cédric Richard

Laboratoire H. Fizeau (CNRS, OCA)
Université de Nice Sophia-Antipolis
06108 Nice, France

ABSTRACT

This paper deals with multi-class classification problems. Many methods extend binary classifiers to operate a multi-class task, with strategies such as the one-vs-one and the one-vs-all schemes. However, the computational cost of such techniques is highly dependent on the number of available classes. We present a method for multi-class classification, with a computational complexity essentially independent of the number of classes. To this end, we exploit recent developments in multifunctional optimization in machine learning. We show that in the proposed algorithm, labels only appear in terms of inner products, in the same way as input data emerge as inner products in kernel machines via the so-called the kernel trick. Experimental results on real data show that the proposed method reduces efficiently the computational time of the classification task without sacrificing its generalization ability.

1. INTRODUCTION

Many challenging classification tasks are multi-class problems, often shown to be more difficult than the binary classification. They imply classifying a data observation into one of a set of possible classes. Multi-class problems are encountered in most applications in signal and image processing, for instance for optical character recognition, speech application [1], face recognition [2], and classification of urban structures in an image [3].

Multi-class classification methods can be roughly divided into two categories. The first one consists of a *single machine* strategy, where a single optimization problem is solved in order to determine the multi-class classifier. These techniques require specific optimization algorithms, often with high computational cost [4, 5]. The second category attempts to take advantage of the performance of binary classifiers such as state-of-the-art least squares classifiers and support vector machines. To this end, the multi-class classification problem is decomposed into a number of binary classification subproblems. The goal is to solve these subproblems, then combine their results to determine the multi-class solution (see for instance [6]). Two strategies exist to extend binary classifiers to multi-class, the *one-versus-all* strategy where binary classifiers are constructed to separate one class from all the other classes, and the *one-versus-one* when separating one class from another (see [7] and references therein). For a problem of m classes, these strategies require solving and combining m binary classifiers for the former, and $m(m-1)/2$ for the latter. From the quantitative and comparative study given in [7] with least squares binary classifiers, the one-vs-all strategy gives at least similar, if not better, results than other methods, including the single machines. Still, this requires m binary classifiers, each constructed using the whole available training data.

In this paper, we show that one can solve a multi-class classification problem, with essentially the same computational complexity as a binary classifier. To this end, we take advantage of recent work in multifunctional optimization in machine learning [8]. We recast the multi-class problem as the estimation of a vector of outputs defining the class membership. By using a least squares formalism, the resulting optimization problem is roughly similar to the one given by a binary classifier. It turns out that its computational complexity is essentially independent of the number of classes, in the same sense as the computational cost in kernel-machines is independent of the dimensionality of the input data, thanks to the so-called kernel trick.

The rest of the paper is organized as follows. Section 2 outlines the least squares approach for a binary classification problem. We describe the proposed multi-class least squares algorithm in Section 3. Section 4 illustrates results obtained with our algorithm, with an image classification problem. Conclusions and further directions are given in Section 5.

2. LEAST SQUARES BINARY CLASSIFICATION

In supervised learning, one seeks a function that predicts well some output from a given input, based on a set of training input-output data, $(\boldsymbol{x}_k, y_k)$, for $k = 1, 2, \ldots, N$. To measure the excellence of the learned function $f(\cdot)$, a loss function is

This work was supported by ANR-08-SECU-013-02 VigiRes'Eau.

considered, such as the commonly used square error, namely, $\ell(f(\boldsymbol{x}_k), y_k) = |f(\boldsymbol{x}_k) - y_k|^2$. The minimization of this square error is considered for binary classification problems in [9, 10] (with connections to the *Fisher discriminant analysis*, however beyond the scope of this work). In this case, the output corresponds to the label, in practice $y_k = \pm 1$.

While this loss function is minimized over all the training data, one also imposes regularity to the function, using for instance the Tikhonov regularization [11]. This gives the following least squares optimization problem:

$$\min_f \sum_{j=1}^N |f(\boldsymbol{x}_j) - y_j|^2 + \gamma \parallel f \parallel^2,$$

where γ is some positive tradeoff parameter. Moreover, in a kernel-based formalism [12], one seeks a function of the form

$$f(\cdot) = \sum_{j=1}^N \alpha_j \, y_j \, \kappa(\boldsymbol{x}_j, \cdot), \tag{1}$$

with $\kappa(\cdot,\cdot)$ being a positive semi-definite function, such as the linear kernel $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^T \boldsymbol{x}_j$ and the Gaussian kernel $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(\frac{1}{2\sigma^2}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2)$ with some bandwidth parameter σ. It is work noting that, since one often sets $y_j = \pm 1$ for a binary classification task, one may inject its sign into α_j with $\beta_j = y_j \alpha_j$, and take the equivalent expansion $f(\cdot) = \sum_{j=1}^N \beta_j \, \kappa(\boldsymbol{x}_j, \cdot)$ (see for instance [9]).

By substituting this expansion into the least squares optimization problem, we obtain a classical matrix formulation, which can be solved using off-the-shelf linear algebra techniques. The resulting coefficients are used in the decision for any $\boldsymbol{x}$, by comparing $f(\boldsymbol{x})$ to the threshold, often set to zero, according to the rule

$$f(\boldsymbol{x}) = \sum_{j=1}^N \alpha_j \, y_j \, \kappa(\boldsymbol{x}_j, \boldsymbol{x}) \underset{y=-1}{\overset{y=+1}{\gtrless}} 0.$$

Binary classifiers can be easily extended to operate for a multi-class classification task. Let m be the number of classes. In a one-vs-one strategy, one constructs $m(m-1)/2$ binary classifiers by taking only data from each pair of classes, while in a one-vs-all strategy, m binary classifiers are considered with all training data. Each binary classifier is defined by the decision function, of the form

$$f_k(\cdot) = \sum_{j=1}^N \alpha_{j,k} \, y_j \, \kappa(\boldsymbol{x}_j, \cdot). \tag{2}$$

Therefore, one needs to estimate $m \times N$ coefficients for the one-vs-all strategy. Next, we propose to estimate only N unknown variables by imposing some relation between these functions, $f_1(\cdot), f_2(\cdot), \ldots, f_m(\cdot)$.

3. MULTI-CLASS LEAST SQUARES CLASSIFICATION

In a multi-class classification problem, we consider a set of N training data, belonging to any of the m available classes. We propose to encode[1] the class membership using a m-column label vector $\boldsymbol{y}_k$, with its t-th entry given by

$$[\boldsymbol{y}_k]_t = \begin{cases} 1 & \text{if data } \boldsymbol{x}_k \text{ belongs to class } t; \\ 0 & \text{otherwise} \end{cases}$$

for $t = 1, 2, \ldots, m$. We propose to estimate all the functions in (2), using a single optimization problem, by requiring some relation between them. We regroup the m functions in the form $\boldsymbol{f}(\cdot) = [f_1(\cdot) \; f_2(\cdot) \; \cdots \; f_m(\cdot)]^T$. By analogy with (1) and following [13], we propose the following expansion for $\boldsymbol{f}(\cdot)$

$$\boldsymbol{f}(\cdot) = \sum_{j=1}^N \alpha_j \boldsymbol{y}_j \kappa(\boldsymbol{x}_j, \cdot).$$

In the proposed expression, all the functions in $\boldsymbol{f}(\cdot)$ share the same scalar value $\alpha_j \, \kappa(\boldsymbol{x}_j, \boldsymbol{x})$, for some j. This allows us to have only N unknowns, the α_j's, in the same way as a single binary classifier. Next, we show how to estimate these coefficients.

In a least squares sense, we consider the following optimization problem

$$\min_{\boldsymbol{f}} \sum_{j=1}^N \|\boldsymbol{f}(\boldsymbol{x}_j) - \boldsymbol{y}_j\|^2 + \gamma \, R(\boldsymbol{f}),$$

where the regularization term is given by

$$R(\boldsymbol{f}) = \sum_{i,j=1}^N \alpha_i \alpha_j \boldsymbol{y}_i^T \boldsymbol{y}_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

By substituting the expression of $\boldsymbol{f}(\cdot)$ in this optimization problem, we get

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \sum_{i=1}^N \boldsymbol{y}_i^T \boldsymbol{y}_i - 2 \sum_{i=1}^N \boldsymbol{y}_i^T \sum_{j=1}^N \alpha_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) \, \boldsymbol{y}_j \\ + \sum_{i=1}^N \sum_{j,k=1}^N \alpha_j \alpha_k \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) \kappa(\boldsymbol{x}_i, \boldsymbol{x}_k) \, \boldsymbol{y}_j^T \boldsymbol{y}_k \\ + \gamma \sum_{i,j=1}^N \alpha_i \alpha_j \, \boldsymbol{y}_i^T \boldsymbol{y}_j \, \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j). \end{aligned}$$

We drop the first term, since it is independent of $\boldsymbol{\alpha}$. In matrix form, this optimization problem can be written as

$$\min_{\boldsymbol{\alpha}} -2 \, \boldsymbol{d} \boldsymbol{\alpha} + \boldsymbol{\alpha}^T \boldsymbol{G} \boldsymbol{\alpha} + \gamma \, \boldsymbol{\alpha}^T \boldsymbol{\Omega} \boldsymbol{\alpha}, \tag{3}$$

[1] In this paper, we do not discuss the issue of optimal coding.

where $\boldsymbol{G}$ is a matrix whose (j,k)-th entry is

$$[\boldsymbol{G}]_{j,k} = \boldsymbol{y}_j^T \boldsymbol{y}_k \boldsymbol{K}^2(j,k)$$

$\boldsymbol{d}$ is a vector whose j-th entry is

$$[\boldsymbol{d}]_j = \sum_{i=1}^{N} \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)\, \boldsymbol{y}_i^T \boldsymbol{y}_j,$$

and $\boldsymbol{\Omega}$ is a matrix whose (i,j)-th entry is

$$[\boldsymbol{\Omega}]_{i,j} = \boldsymbol{y}_i^T \boldsymbol{y}_j\, \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

By taking the gradient of the objective function in (3) with respect to $\boldsymbol{\alpha}$, namely

$$-\boldsymbol{d} + \boldsymbol{G}\boldsymbol{\alpha} + \gamma\, \boldsymbol{\Omega}\, \boldsymbol{\alpha},$$

and setting it to zero, we obtain the final solution

$$(\boldsymbol{G} + \gamma\, \boldsymbol{\Omega})\, \boldsymbol{\alpha} = \boldsymbol{d}. \tag{4}$$

We conclude that a multi-class regularized least squares classifier can be obtained by solving a single system of N linear equations with N unknowns. This requires the inversion of a N-by-N matrix, $\boldsymbol{G} + \gamma\, \boldsymbol{\Omega}$. The computational complexity of this algorithm is cubic in the number of training data, N, but independent of the number of classes m. This is made possible here thanks to the fact that the label vectors only appear in terms of inner products, with $\boldsymbol{y}_i^T \boldsymbol{y}_j$ in $\boldsymbol{\Omega}$, $\boldsymbol{G}$ and $\boldsymbol{d}$. This reflects an analogy with the kernel trick in kernel machines, where data are involved only in terms of inner products, namely $\boldsymbol{x}_i^T \boldsymbol{x}_j$.

To predict the class membership of any $\boldsymbol{x}$, the decision rule compares the output $\boldsymbol{f}(\boldsymbol{x})$ with the set of label vectors $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots \boldsymbol{y}_m$, with

$$d(\boldsymbol{x}) = \arg\max_t\, \boldsymbol{y}_t^T \boldsymbol{f}(\boldsymbol{x}).$$

This decision rule can also be written using inner products between label vectors, namely

$$d(\boldsymbol{x}) = \arg\max_t \sum_{i=1}^{N} \alpha_i\, \boldsymbol{y}_t^T \boldsymbol{y}_i\, \kappa(\boldsymbol{x}_i, \boldsymbol{x}). \tag{5}$$

4. EXPERIMENTATIONS

Two experiments were conducted to illustrate the pertinence of the proposed approach. First, an illustrative application is considered, with the IRIS data from the UCI Machine Learning Repository, often used as a benchmark for classification algorithms. It consists of samples of flowers representing $m = 3$ iris species. Each species consists of 50 observations, four features were measured from each sample, they are the length and the width of sepale and petale. For illustration, only sepale length and petale width were used. A set of $N = 120$ training data was used for learning the multi-class classifier. Some preliminary experiments were conducted to tune the parameters, leading to $\gamma = 100$ and $\sigma = 0.60$ for the Gaussian kernel. Figure 1 illustrates the resulting separating boundaries, and shows the decision rule given in (5) in the third dimension. For a comparative study, we considered the least squares one-vs-all strategy, as given in Section 2. Using the remaining set of 30 data, both multiclass classification methods gave essentially similar classification error, equal to 3%. In an attempt to provide a measure of computational requirements[2], the one-vs-all classifier was trained with a (average) total CPU time of up to 0.011 seconds, while the proposed algorithm required only 0.003 seconds.

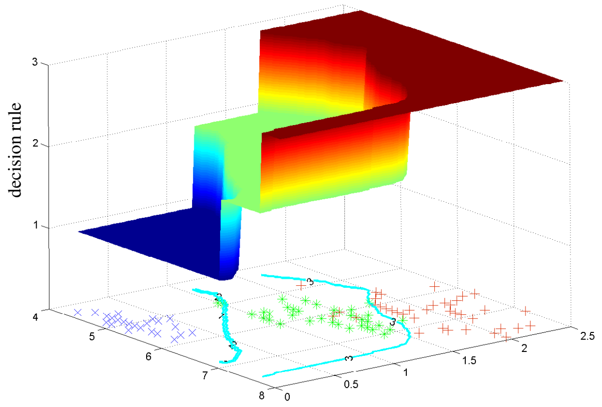

Fig. 1. Illustration of the IRIS classification task, with the separating boundaries (—) and the decision rule taking values 1, 2 and 3.

For the second application, we considered a multi-class classification task, where the number of available data is not

Table 1. The 7 classes with the ratio of train/test samples in the hyperspectral image.

Class-name	#train	#test
⬜ Asphalt	210	3 088
🟩 Meadows	236	5 216
🟩 Trees	196	1 279
🟪 Metal sheets (painted)	256	1 345
🟫 Bare soil	332	1 264
🟥 Self-blocking bricks	225	1 693
🟨 Shadow	28	215
	1 492	**14 100**

[2] To offer a comparative study, both algorithms were implemented on a MATLAB running on a Windows.

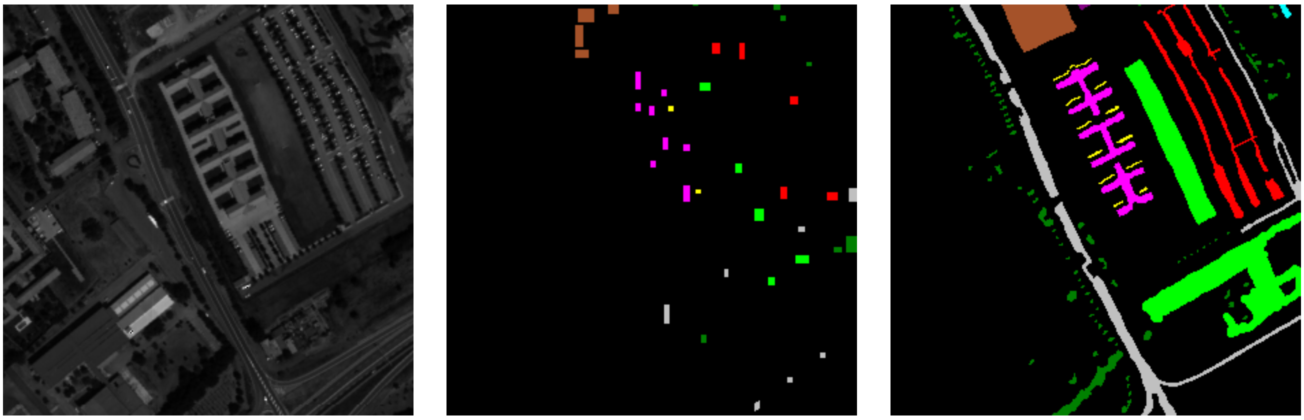

Fig. 2. The hyperspectral image (slice at mid spectral-band) (left), with the spatial distribution of the training (middle) and test (right) data. The legend of the 7 classes is given in Table 1.

the same for all classes. Recently adopted in remote sensing, hyperspectral images are cubes of data, measuring spectral composition within a spatial view. As opposed to the conventional 3-color system, the spectral information over a hundred of bands provides greater analysis of the composition of objects in an image scene. In monitoring urban structures with airborne or satellite images, one is often confronted with a large number of classes.

The hyperspectral image, provided by the HySenS project, is of the University of Pavia, Italy, with a 300-by-300 pixels and 103 frequency bands, illustrated in Figure 2 (left). Ground truth information about 7 classes were included to train and test the classifiers, as given in Table 1 illustrated in Figure 2 (middle and right) (see [3] for more details). The Gaussian kernel was used, with its bandwidth value set to the maximum value in the training data. Both the proposed method and the one-vs-all method give almost similar results, with an overall error of 8% for both. It is worth noting that our algorithm estimates $N = 1\,492$ coefficients by the inversion of a $1\,492$-by-$1\,492$ matrix, independent of the number of classes, while the one-vs-all strategy estimates $7 \times 1\,492$ coefficients for the $m = 7$ classes, thus operates the inversion of a $10\,444$-by-$10\,444$ matrix.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we studied the multi-class least squares classification, and showed that one can do multi-class classification at binary-classification complexity. While the idea is simple, its relevance was illustrated on real data, showing that our approach reduces efficiently the computational complexity without sacrificing its accuracy. In future work, we exploit further the proposed approach, by incorporating a nonlinear measure of similarity between label vectors, i.e., the use of a kernel on labels as opposed to the (linear) one in this paper.

6. REFERENCES

[1] A. Klautau, N. Jevtic, and A. Orlitsky, "Combined binary classifiers with applications to speech recognition," vol. 4. International Conference on Spoken Language Processing, 2002.

[2] Z. Lihong, S. Ying, Z. Yushi, Z. Cheng, and Z. Yi, "Face recognition based on multi-class SVM," in *Proc. 21st annual international conference on Chinese control and decision conference*. IEEE Press, 2009.

[3] P. Honeine and C. Richard, "The angular kernel in machine learning for hyperspectral data classification," in *Proc. IEEE Workshop on Hyperspectral Image and Signal Processing : Evolution in Remote Sensing (WHISPERS)*, Reykjavík, Iceland, 2010.

[4] J. Weston and C. Watkins, *Support Vector Machines for Multi-Class Pattern Recognition*. Proc. Seventh European Symposium On Artificial Neural Networks, 1999.

[5] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," *Journal of Machine Learning Research*, vol. 1, pp. 143–160, 2002.

[6] J. Suykens and J. Vandewalle, "Multiclass least squares support vector machines," in *Proc. International Joint Conference on Neural Networks*. World Scientific, 1999.

[7] R. Rifkin and A. Klautau, "In defense of one-vs-all classification," *Journal of Machine Learning Research*, vol. 5, pp. 101–141, 2004.

[8] T. Evgeniou, C. A. Micchelli, and M. Pontil, "Learning multiple tasks with kernel methods," *Journal of Machine Learning Research*, vol. 6, pp. 615–637, 2005.

[9] R. Rifkin, "Everything old is new again: A fresh look at historical approaches in machines learning," in *PhD thesis, MIT*, 2002.

[10] P. Zhang and J. Peng, "SVM vs regularized least squares classification," in *Proc. 17th International Conference on Pattern Recognition*. Washington, DC, USA: IEEE Computer Society, 2004.

[11] A. N. Tikhonov and V. Y. Arsenin, *Solutions of Ill-Posed Problems*. V. H. Winston & Sons, Washington, D.C.: John Wiley & Sons, New York, 1977.

[12] B. Schölkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," in *Proc. 14th Annual Conference on Computational Learning Theory and 5th European Conference on Computational Learning Theory*. Springer-Verlag, 2001.

[13] S. Szedmak and J. Shawe-Taylor, "Multiclass learning at one-class complexity," 2005, technical Report, ISIS Group, Electronics and Computer Science. (Unpublished).