Journal of Computing and Information Technology - CIT 14, 2006, 4, 283–289
doi:10.2498/cit.2006.04.03

Dendogram-based SVM for Multi-Class Classification

Khalid Benabdeslem[1] and Younès Bennani[2]

[1]University of Lyon 1 – PRISMa Lab, Villeurbanne Cedex, France
[2]University of Paris 13 – LIPN/CNRS Lab, Villetaneuse, France

This paper presents a new approach called dendogram-based support vector machines (DSVM), to treat multi-class problems. First, the method consists to build a taxonomy of classes in an ascendant manner done by ascendant hierarchical clustering method (AHC). Second, SVM is injected at each internal node of the taxonomy in order to separate the two subsets of the current node. Finally, for classifying a pattern query, we present it to the "root" SVM, and then, according to the output, the pattern is presented to one of the two SVMs of the subsets, and so on through the "leaf" nodes. Therefore, the classification procedure is done in a descendant way in the taxonomy from the root through the end level which represents the classes. The pattern is thus associated to one of the last SVMs associated class. AHC decomposition uses distance measures to investigate the class grouping in binary form at each level in the hierarchy. SVM method requires little tuning and yields both high accuracy levels and good generalization for binary classification. Therefore, DSVM method gives good results for multi-class problems by both, training an optimal number of SVMs and by rapidly classifying patterns in a descendant way by selecting an optimal set of SVMs which participate to the final decision. The proposed method is compared to other multi-class SVM methods over several complex problems.

Keywords: AHC, SVM, multi-class problems.

1. Introduction

Achieving both high accuracy and good generalization for complex problems of classification is a challenging problem, especially when data distribution is not linear and the number of classes is large. In this context, support vector machines have demonstrated superior performance [2]. However, SVM was originally designed for binary classification and its extension for multi-class classification is still an on-going research issue [3].

Actually, we distinguish two types of multi-class SVM approaches: one by directly considering all data in one optimization formulation [11] and the other by building and combining several binary classifiers [12,13]. In general, it is computationally more expensive to solve a multi-class problem than a binary problem with the same number of data. This work is devoted to the second approach, i.e. it solves a multi-class problem by decomposing it to several binary problems in a hierarchical way.

The popular methods which decompose multi-class problems into many binary class problems are "one-against-all" and "one-against-one" approaches [14].

The "one-against-all" approach is a simple and effective method for multi-class classification. Suppose there are K classes in the problem. We partition these K classes into two-class problems: one class contains patterns in one "true" class and the "others" class combines all other classes. A two-class classifier is trained for this two-class problem. We then partition the K classes into another original class, and the 'others' class contains the rest. Another two way classifier is trained. This procedure is repeated for each of the K classes, leading to K two-way trained classifiers.

In the recognition process, the system tests the new query pattern against each of the K two-way classifiers, to determine if it belongs to the given class or not. This leads to K scores from the K classifiers. Ideally, only one of the K classifiers will show a positive result and all other classifiers will show negative results, assigning the query pattern to a unique class. In practice, however, many patterns show positive on more than one class, leading to ambiguous classification results, the so-called 'False positive' problem. One of the main reasons for the false positive problem is that the decision boundary between one 'true' class and its complementary 'others' class cannot be drawn cleanly, due to the complexity of the 'others' class and close parameter proximity of some patterns.

In the "one-against-one" method, we train two-way classifiers between all possible pairs of classes; there are $K(K-1)/2$ of them. A new query pattern is then tested against these $K(K-1)/2$ classifiers and $K(K-1)/2$ scores (votes) are obtained. In a perfect case, the correct class will get the maximum possible votes, which is $(K-1)$ for all class-class pairs; and votes for other $(K-1)$ classes would be randomly distributed, leading to $[K(K-1)/2 - K-1]/(K-1) = (K-2)/2$ per class on average.

Subsequently, a K-class problem needs $K(K-1)/2$ binary SVMs with "one-against-one" approach and K SVMs for the "one-against-all" approach. Although the "one-against-one" approach demonstrates superior performance, it may require prohibitively expensive computing resources for many real world problems. The "one-against-all" approach shows somewhat less accuracy, but still demands heavy computing resources, especially for real time applications.

The new method proposed in this paper provides an alternative to the two presented methods. The proposed DSVM takes advantage of both the efficient computation of the ascendant hierarchical clustering of classes and the high classification accuracy of SVM for binary classification. Although DSVM needs (K-1) SVMs for K-class problem in the training phase, for the testing phase DSVM requires an optimal set of SVMs selected in a descendant way from the root of the taxonomy through the selected class among the "leaf" nodes.

In Section 2, we present the basic concept of SVM for linear and non linear problems. In Section 3 we describe the concept of our DSVM method. Finally, Section 4 shows our results obtained by comparing the proposed method with other ones over several problems.

2. Support Vector Machines and Binary Classification

The support vector machine is originally a binary classification method developed by Vapnik and colleagues at Bell laboratories [5, 6], with algorithm improvements by others [7, 9]. SVM consists of projecting the input vectors into a high dimensional feature space, then searches for the linear decision boundary that maximizes the minimum distance between two class groups (Figure 1).

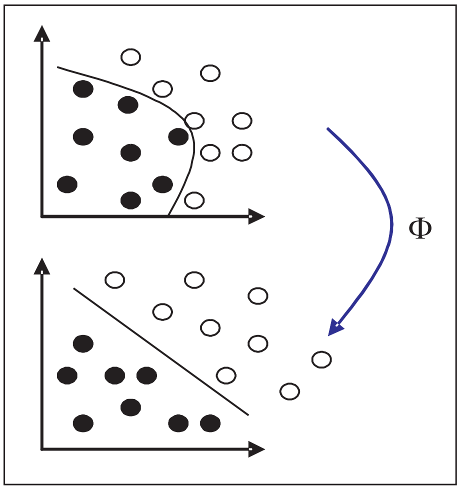

Fig. 1. General principle of SVM: projection of data in an optimal dimensional space.

In Figure 1 we can see that data are not linearly separable in the initial space a) and after projection (by function: Φ) they become separable in the high dimensional space b). SVM then consists of finding the optimal boundary for separating the positive class (dark circles) from the negative one (white circles).

SVM separate between these two classes via a hyperplane that is maximally distant from the positive samples and from the negative ones (Figure 1), then 'plot' the test data at the high dimensional space, distinguish whether it belongs to positive or negative according to the optimal hyperplane.

For a binary classification problem with input space X and binary class labels Y :

$$Y \in \{-1, 1\}.$$

Giving training samples (y_1, x_1) , ..., (y_l, x_l).

$$y_i \in \{-1,\ 1\} \tag{1}$$

the goal of SVM is to search for the optimal hyperplane

$$w \cdot x + b = 0 \tag{2}$$

with variables w and b that satisfy the following inequality

$$y_i(w \cdot x_i + b) \geq 1, \quad i = 1, \ldots, l, \tag{3}$$

defining the minimum distance between two class groups in the new projection.

$$d(w,b) = \min_{x/y=+1} \frac{x^T.w}{||w||} - \max_{x/y=-1} \frac{x^T.w}{||w||} \tag{4}$$

From e.q. (3), $\min\limits_{\{x:y=1\}} x.w{=}1$ and $\max\limits_{\{x:y=-1\}} x.w{=}-1$. Substituting back into e.q. (4), yields

$$d(w_0, b_0) = \frac{2}{||w_0||} = \frac{2}{\sqrt{w_0 w_0}}.$$

For a given training set w, b that maximizes $d(w_0, b_0)$ solves the following quadratic optimization problem:

$$\begin{aligned} &\min_w \frac{1}{2} w.w \\ &\text{s.t.}\ \ y_i(w.x_i + b) \geq 1 \quad i = 1, \ldots, l. \end{aligned} \tag{5}$$

If the given training sample set is linearly separable, the optimization problem (5) has feasible solutions. The optimal solution w, and b forms the best hyperplane that maximizes the margin between two different classes in the new projection. Because SVM search for the best separation hyperplane instead of the highest training sample accuracy, they never over-train on a sample data set. If the parameters are properly selected, SVM typically produce both excellent classification results and good generalization if parameters are properly selected. Not every problem is guaranteed to be linearly separable, so a soft margin hyperplane SVM was developed to separate the training set with a minimal number of errors [5].

A number of candidate kernel functions have been used in SVM, including polynomial

$$K(x,y) = (1 + x.y)^d\,,$$

exponential RBF

$$K(x,y) = exp\left(-\frac{||x-y||}{2\sigma^2}\right)$$

and Gaussian RBF

$$K(x,y) = exp\left(-\frac{||x-y||^2}{2\sigma^2}\right)\,.$$

3. Dendogram-based SVM

The DSVM method that we propose consists of two major steps: (1) computing a clustering of the known classes and (2) associating a SVM at each node of the taxonomy obtained by (1).

Let's take a set of samples $x_1, x_2, \ldots, x_n$ labeled each one by $y_i \in \{c_1, c_2, \ldots, c_k\}$, k is the number of classes ($k \leq n$).

The first step of DSVM method consists of calculating k gravity centers for the k known classes. Then AHC clustering is applied over these k centers (Figure 2).

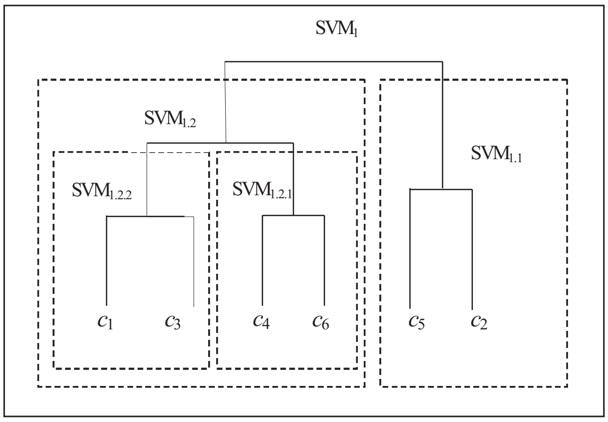

Fig. 2. Grouping classes in hierarchical way by AHC method.

Figure 2 shows an example of a taxonomy done by AHC [16] algorithm over the k classes.

In the second step, each SVM is associated to a node and trained with the elements of the two subsets of this node. For example, in Figure 2 which illustrates clustering of 6 classes SVM_1 is trained by considering elements of $\{c_5, c_2\}$ as positives and elements of $\{c_1, c_3, c_4, c_6\}$ as negatives; $SVM_{1.2}$ is trained by considering elements of $\{c_4, c_6\}$ as positives and elements of $\{c_1, c_3\}$ as negatives. The concept is repeated for each SVM associated to a node in the taxonomy. In final, we will train $(k-1)$ SVM for k-class problem.

The advantage of training in DSVM is to a priori separate the classes in a hierarchical way. That facilitates the class separation for the SVM. In fact, SVM_1 found easily boundary separation between $\{c_5, c_2\}$ and $\{c_1, c_3, c_4, c_6\}$. The level of difficulty for boundary separation increases from the root through the leaves. The idea of DSVM is that it is preferable to solve many small problems in a hierarchical way than to solve a complex great problem. For classifying a pattern query, DSVM presents it to the "root" SVM which provides an output for right or left on the taxonomy. The procedure is repeated for each selected node in the "way" of the classification (Figure 3) until arriving to a leaf which finally represents the associated class for our pattern query.

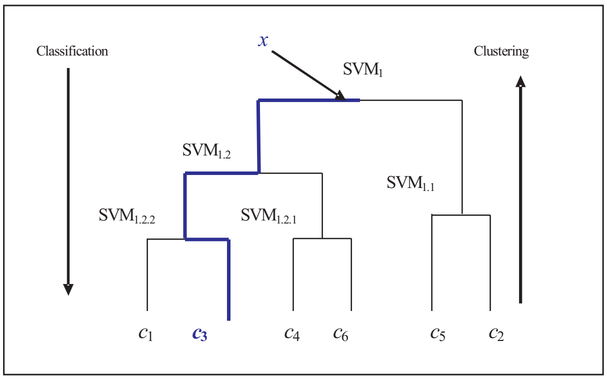

Fig. 3. Classification way by DSVM.

In Figure 3 we show the classification way of a given pattern: x via DSVM. x is presented to SVM_1 (root), then output of SVM_1 is : $x \in \{c_1, c_3, c_4, c_6\}$. Next, x is presented to $SVM_{1.2}$, the output decision is $x \in \{c_1, c_3\}$. Finally, x is presented to $SVM_{1.2.2}$ when the output is: $x \in \{c_3\}$. DSVM provides a trace of classification of x which is $SVM_1 \rightarrow SVM_{1.2} \rightarrow SVM_{1.2.2}$. We can see that classification procedure is more optimal with DSVM than with the other multi class SVM and it provides a unique solution for the pattern x. In fact, in our example, the system requires 3 among the 5 trained SVMs for making the decision.

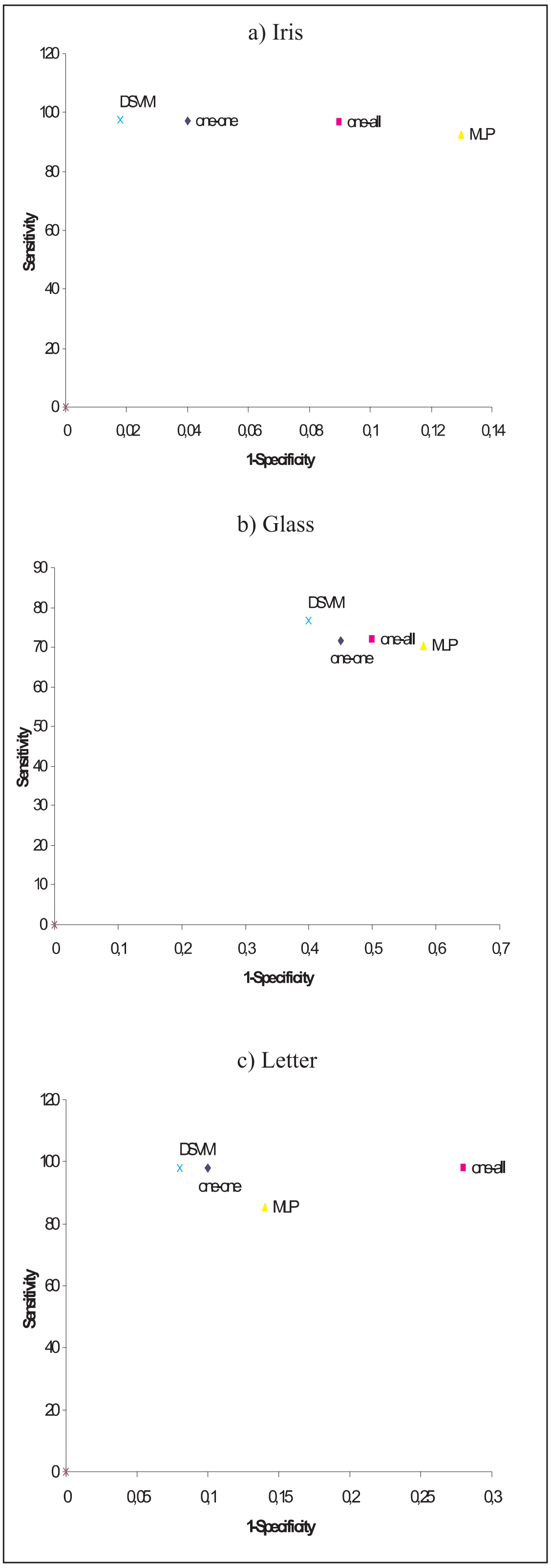

Fig. 4. ROC curves for the three problems.

4. Experiment Results

4.1. Data for Validation

We have performed several experiments on three known problems from the UCI Repository of machine learning databases [15]. The chosen databases are: "Iris", "Glass" and "Letter". We give the problem statistics in Table 1.

problem	#training data	#testing data	#class	#attributes
Iris	100	50	3	4
Glass	142	72	6	9
Letter	10000	5000	26	16

Table 1. Problem statistics

In Table 1 we can see that for each problem we have used $2/3$ of data for training and $1/3$ for testing, and that the 3 problems differ in dimensional input space, in size of database and in the number of classes.

4.2. Accuracy Measures

The goal of these experiments is to evaluate our method vs. "one-against-one", "one-against-the other" and MLP (Multi layer perceptron) methods. The most important criterion in evaluating the performance of these methods is their accuracy rate. In addition, we will present the time of training of each method and the number of trained SVMs for multi class SVM methods. Accuracy of the results obtained with discriminative methods is commonly measured by the quantity of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN). In addition to these quantities, standard sensitivity and specificity measures defined by:

Sensitivity=TP/(TP+FN),

Specificity=TN/(TN + FP),

are also useful in assigning the classification accuracy. All these quantities are used in the evaluation of classification methods in this work.

4.3. Results

Experiments were performed using "one-against-one", "one-against-all", Multi Layer perceptron (MLP) and the proposed DSVM methods. Average classification accuracies for test data of each problem conducted with each classifier are listed in Table 2.

problem	One-against -one	One-against -all	MLP	DSVM
Iris	97.333 [83.23,100]	96.667 [81.9, 100]	92.48 [74.42, 100]	97.619 [83.83, 100]
Glass	71.495 [50.38,90.43]	71.963 [50.92, 90.78]	70.340 [49.06, 89.56]	76.76 [56.58, 94.74]
Letter	97.98 [97.18,98.71]	97.88 [97.06, 98.63]	85.236 [83.28, 87.14]	98.012 [97.21, 98.74]

Table 2. Accuracy comparison between methods.

Table 2 presents the result of comparing the four methods. For SVM-based methods, Gaussian RBF kernel is used in the training phase. MLP is trained with different number of hidden neurons for each problem, and optimal numbers given best MLP's accuracy are selected.

The values between brackets represent the confidence intervals for 95% confidence level, computed as described in [1]. In Figure 4, the ROC (Receiver operating characteristics) space shows that DSVM approach gives better total result in both sensitivity and specificity than the other methods. In Table 3 we also report the training time and the number of trained SVMs.

problem	One-against -one Time/ #svm	One-against -all Time/ #svm	MLP Time	DSVM Time/ #svm
Iris	0.04/3	0.10/3	0.047	0.03/2
Glass	2.42/15	10/6	140	0.09/5
Letter	298.08/325	1831/26	4500	140/25

Table 3. Training time (in sec) and #SVMs.

We can see that both training time and number of trained SVMs are considerably decreased in

DSVM learning method. In addition, the classification time is also low because of the optimal set of SVMs selected in a descendant way in the taxonomy.

problem	Classes	Sensitivity (%)	Specificity (%)
Iris	Class1	100	100
	Class2	92.87	100
	Class3	100	94
Glass	Class1	84	68
	Class2	86	84
	Class3	0	0
	Class4	93.56	100
	Class5	100	100
	Class6	97	82
Letter	Class1	98	100
	Class2	100	92
	Class3	96	93
	Class4	98	97
	Class5	100	93
	Class6	96	92
	Class7	97	99
	Class8	100	88
	Class9	97	87
	Class10	99	91
	Class11	100	92
	Class12	98	94
	Class13	99	91
	Class14	97	100
	Class15	97	93
	Class16	98	97
	Class17	94	98
	Class18	100	94
	Class19	98	88
	Class20	97	92
	Class21	100	78
	Class22	99	96
	Class23	98	100
	Class24	97	96
	Class25	97	75
	Class26	98	88

Table 4. Detailed test results for different classes of the three problems with DSVM.

The results of the classification of DSVM for each class are given in detail for each problem in Table 4.

The sensitivity and specificity are the range of 92-100 % and 94-100%, respectively, for "Iris" problem, 94-100% and 75-100% for "Letter" problem. For "Glass" problem, globally the accuracy is good for all classes except class 3, because the samples of this class are "very" non-linearly distributed in the input space and they are generally recognized in class 1. That results in confusion between the two classes.

5. Conclusion and Future Works

A new hierarchical support vector machines (DSVM) approach has been developed. This method utilizes a taxonomy of classes and decomposes a multi-class problem to a descendant set of binary-class problems. AHC method is used to group all classes in an ascendant hierarchy. This clustering allows us to separate the classes and to build different subsets from database for different sub-problems. Then SVM classifier is applied at each internal node to construct the best discriminant function of a binary-class problem.

In this paper, DSVM was evaluated using a series of experiments. Compared with the two famous multi-class SVM methods and a MLP-based neural networks DSVM consistently achieves both high classification accuracy and good generalization. DSVM takes its advantage from two good methods: (1) AHC clustering, which uses distance measures to investigate the natural class grouping in hierarchical way and (2) original binary SVM classifiers to separate the different classes because of their solid mathematical foundations. Combining these two methods, DSVM extends binary-SVM to a fast multi-class classifier.

Future work reports to develop a dynamic Kernel in the taxonomy for treating the different binary-classification. This dynamic kernel will take into account the difficulty of data separation of positive and negative samples from the root through the leaf nodes in the hierarchy.

References

[1] Y. BENNANI, Multi-expert and hybrid connectionist approach for pattern recognition : speaker identification task . *International Journal of Neural Systems*, Vol. 5, No. 3, (1994) pp. 207–216.

[2] B. E. BOSER, I. GUYON, AND V. VAPNIK, A Training Algorithm for Optimal Margin Classifiers, in *Computational Learning Theory*, 1992, pp. 144–152.

[3] C.-W. HSU AND C.-J. LIN, A comparison of methods for multi-class support vector machines, *IEEE Transactions on Neural Networks*, 2002., vol. 13, pp. 415–425.

[4] S. KUMAR, J. GHOSH, AND M. CRAWFORD, Hierarchical Fusion of Multiple Classifiers for Hyperspectral Data Analysis, *International J. Pattern Analysis and Applications*, 2002., vol. 5, no. 2, pp. 210–220.

[5] V. N. VAPNIK, *The Nature of Statistical Learning Theory*, second ed., Springer, New York, 1999.

[6] C. J. C. BURGES, *A tutorial on support vector machine for pattern recognition*, Data Min. Knowl. Disc. 2 (1998) 121.

[7] E. OSUNA, R. FREUND AND F. GIROSI, An improved training algorithm for support vector machines. *Neural networks for Signal Processing VII – proceeding of 1997 IEEE Workshop*, pp. 276–285.

[8] Z. Y. LI, S. W. TANG, S. C. YAN, *Multi-class SVM classifier based on pairwise coupling*, Lect. Notes Comput. Sci. 2388 (2002) 321.

[9] T. JOACHIMS, Making large scale SVM learning practical. In scholkopf, B Bruges C and Smola A (eds), *Advances in kernel methods-support vector learning*. MIT Press, Cambridge, MA. 1998.

[10] N. CRISTIANINI, J. SHAWE-TAYLOR, An Introduction to Support Vector Machines, Cambridge University, Cambridge, 2000.

[11] Y. GUERMEUR, A. ELISSEEFF AND H. PAUGAM-MOISY. A new multi-class SVM based on a uniform convergence result. *IJCNN2000*, Come, Vol. IV, pp. 183–188.

[12] E. J. BREDENSTEINER AND K. P. BENNETT, Multicategory classification by support vector machines. Computational optimizations and applications, 1999., pp. 53–79.

[13] Y. CHEN, M. M. CRAWFORD AND J. GHOSH, Integrating support vector machines in a hierarchical outputs space decomposition framework. `http://studentweb.engr.utexas.edu/Chen2/yangchi04hierarchical.pdf`

[14] M. P. S. BROWN, W. N. GRUNDY, D. LIN, N. CRISTIANINI, C. SUGNET, M. J. R. ARES AND D. HAUSSLER, Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proc Natl Acad Sci*, USA 97, pp. 262–267.

[15] C. L. BLAKE AND C. J. MERZ, UCI repository of machine learning databases. *Technical report*. University of California, Department of information and Computer science, Irvine, CA, 1998. available at: `ftp://ftp.ics.uci.edu/pub/machine-learning-databases/`

[16] J-M. BOUROCHE, G. SAPORTA, *L'analyse des données*. Presse universitaire de France 1994.

Received: June, 2006
Accepted: September, 2006

Contact addresses:
Khalid Benabdeslem
University of Lyon1 – PRISMa Lab
8 Boulevard Niels Bohr
69622 Villeurbanne Cedex
France
`kbenabde@bat710.univ-lyon1.fr`

Younès Bennani
University of Paris13 – LIPN/CNRS Lab
99 Avenue J.Baptiste Clément
93430 Villetaneuse
France
`younes.bennani@lipn.univ-paris13.fr`

DR. KHALID BENABDESLEM graduated from USTO University where he received his engineer diploma in 1998. Thereafter, he gained an MSC (DEA) in Artificial Intelligence from the Paris 13 University in 2000. In 2003, he received his PhD degree in Computer Science from the University of Paris 13. After two Postdoctoral Fellowships (CNRS and INRIA) he is currently associate Professor at the University of Lyon 1 and a member of Learning and Knowledge Discovery (LKD) team, PRISMa Lab. His main researches are centred on machine learning (Connectionist, Probabilistic and Statistic)

PROF. YOUNÈS BENNANI received his B.S. in Mathematics and Computer Science from Rouen University, his M.S. and Ph.D. degrees in Computer Science from the University of Paris 11 (ORSAY) in 1992, and the HDR (Habilitation à Diriger des Recherches) degrees in Computer Science from the University of Paris 13 in 1998. He joined the Computer Science Laboratory of Paris-Nord (LIPN-CNRS) in 1993. He is currently Full Professor of Computer Science with research interest in theory of Neural Networks, Statistical Pattern Recognition and Datamining. He is also interested in the application of these models to speech/speaker/languages recognition, diagnosis of complex systems, abnormal situation detection, Webmining and call mining. He has published 2 books and approximately 100 papers in refereed conference proceedings or journals or as contributions in books. Prof. Younès Bennani is the head of the Machine Learning research team at the LIPN-CNRS. He gives MS lectures on neural networks and statistical pattern recognition at the Paris 13 University.