TOWARDS MINIMIZING THE ENERGY OF SLACK VARIABLES FOR BINARY CLASSIFICATION

Margarita Kotti and Konstantinos I. Diamantaras

TEI of Thessaloniki, Department of Informatics, Sindos 57400, Greece
email: mkotti@it.teithe.gr,kdiamant@it.teithe.gr

ABSTRACT

This paper presents a binary classification algorithm that is based on the minimization of the energy of slack variables, called the Mean Squared Slack (MSS). A novel kernel extension is proposed which includes the withholding of just a subset of input patterns that are misclassified during training. The later leads to a time and memory efficient system that converges in a few iterations. Two datasets are exploited for performance evaluation, namely the adult and the vertebral column dataset. Experimental results demonstrate the effectiveness of the proposed algorithm with respect to computation time and scalability. Accuracy is also high. In specific, it equals 84.951% for the adult dataset and 91.935%, for the vertebral column dataset, outperforming state-of-the-art methods.

***Index Terms*—** Slack minimization, kernel methods, binary classification, support vector machines, iterative solving

1. INTRODUCTION

Machine learning is an ever evolving research area. Support vector machines (SVMs) are mathematically well-founded machine learning methods that are widely applied to many real-word domains. They are maximum margin classifiers that try to find the hyperplane which optimally separates the data into two categories. The term margin refers to the minimum distance from the separating hyperplane to the closest training feature vector. The key feature vectors are those at the margin and they are called support vectors. Support vectors rely only on a few data points to define the classifier's hyperplane. Accordingly, SVMs present the ability to generalize well even with a limited number of training data.

Linear SVM can be extended to nonlinear ones when the feature space is transformed into a higher feature space using a set of nonlinear basis functions. Hopefully, in the higher dimension feature space the input feature vectors may be separated linearly. An advantage of the SVM is that it is not necessary to explicitly implement this transformation and to determine the separating hyperplane. Instead a kernel representation can be used, where the solution is written as a weighted sum of the values of certain kernel function evaluated at the support vectors. In this paper we present an SVM alternative which instead of minimizing the norm of the weight vector and putting a penalty on the slack variables like SVMs, it minimizes the energy of the slack variables. For the kernel case, we propose a method that retains just a limited subset of training feature vectors for kernel computation. Accordingly, the proposed algorithm is efficient with respect to computational time and memory demands.

With respect to applications, SVMs are exploited at computer vision, bio-informatics and natural language processing. This may partially due to that both fields deal with high-dimensional problems, such as microarray processing tasks, and text categorization. Additionally, SVMs have been tested for speech and speaker recognition, emotion classification, e-learning, database marketing, intrusion detection, geo- and environmental sciences, finance time series forecasting, and high energy physics. The aforementioned list of applications is just indicative but not exhaustive.

Recent methods exploit the idea of constructing kernel algorithms, where the starting point is not a linear algorithm [1], but a linear criterion. The latter can be turned into a condition involving an efficient optimization over a large function class using kernels, thus yielding tests for independence of random variables, or tests for solving the two-sample problem. A linear criterion may be for example that two random variables have zero covariance, or that the means of two samples are identical. Other alternatives try to improve scalability, exploiting parallel SVM (PSVM) [2], which reduces memory use through performing a row-based, approximate matrix factorization, and which loads only essential data to each machine to perform parallel computation. Additional recent theory advancements include generalization bounds based on Rademacher complexity theory for model selection and error estimation. Moreover, a dimension-independent bound of the generalization error may be computed based on probably approximately correct (PAC) Bayesian theory.

The rest of the paper is organized as follows: In Section 2 the proposed method is analysed, in Section 3 experimental

This work has been supported by the "Thalis" Programme, project "Study and Forecasting of Economic Data using Machine Learning Methods", funded in part by the European Union and in part by the Greek Ministry of Education, Lifelong Learning and Religious Affairs.

results are presented on two datasets of radically different nature. Finally, conclusions are drawn is Section 4.

2. MEAN SQUARED SLACK MINIMIZATION

2.1. Problem formulation

Let us consider the classification task for a set of training data

$$\mathcal{X} = \{(\mathbf{x}(i), t(i)) \mid \mathbf{x}(i) \in \mathbb{R}^n\}_{i=i}^{N} \quad (1)$$

where $\mathbf{x}(i)$ is the ith feature vector of size n, $t(i) \in \{-1, 1\}$ is the class label of $\mathbf{x}(i)$, and N is the size of $\mathcal{X}$. The classification task aims to find a proper weight vector $\mathbf{w} \in \mathbb{R}^n$ and bias b that solve the following set of inequalities:

$$t(i)(\mathbf{w}^T\mathbf{x}(i) + b) \geq 1, \quad i = 1, \cdots, N. \quad (2)$$

However, there may not exist any feasible solution for inequality (2). A treatment to overcome this obstacle has been proposed at [3]. It is useful to define a slack variable $\xi(i)$, associated with pattern i, so that it holds

$$t(i)[\mathbf{w}^T\mathbf{x}(i) + b] \geq 1 - \xi(i), \quad (3)$$

$$\xi(i) = \max\{1 - t(i)[\mathbf{w}^T\mathbf{x}(i) - b],\ 0\} \quad (4)$$

where $\xi(i) \geq 0$.

2.2. The linear case

Typically a maximum margin classifier, such as an SVM, seeks to minimize the norm of the weight vector $\mathbf{w}$ under the constraints described in (2) and putting a penalty on the slack variables. An alternative approach would be to minimize the **Mean Squared Slack** (**MSS**), as explored in our previous work [4]. Let us define

$$J_{MSS} = \frac{1}{2}\bar{E}\Big\{(1 - t[\mathbf{w}^T\mathbf{x} - b])^2 \ \Big|\ 1 > t[\mathbf{w}^T\mathbf{x} + b]\Big\} \quad (5)$$

where $\bar{E}\{X \mid Y\}$ is the empirical average of the sequence $X(i)$ under condition Y:

$$\bar{E}\{X \mid Y\} = \frac{1}{N_Y} \sum_{\substack{\text{all } i \text{ where} \\ Y \text{ is true}}} X(i) \quad (6)$$

and N_Y is the number of instances where Y is true. Note that we care only for those patterns which give $t(i)(\mathbf{w}^T\mathbf{x} + b) < 1$. This is reasonable, since, in the classification context, *only* the "bad" patterns that fail to satisfy inequality (2) should contribute to the cost, while all the others should not.

Motivated by our quest for a faster algorithm we explore here methods minimizing the MSS. The following statements are true [4]:

- if problem (2) is linearly separable, then the minimum $J_{MSS} = 0$ is attained by $[\mathbf{w}^T,\ b]$ iff $[\mathbf{w}^T,\ b]$ is a separating vector;
- if the problem is not linearly separable, then the cost function J_{MSS} attains its minimum for some $[\mathbf{w}^T,\ b]$ with $0 < \|[\mathbf{w}^T,\ b]\| < \infty$.

Algorithm 1 Slackmin algorithm (linear case)

Input: $MAX\_ITERATIONS$
{INITIALIZATION}
Initialize $\mathbf{w}, b$ to random values
{TRAINING}
for $MAX\_ITERATIONS$ iterations **do**
$\mathbf{y} = \mathbf{w}^T\mathbf{x} + b$
$S = \{i\ :\ 1 > t(i)[\mathbf{w}_k^T\mathbf{x}(i) + b]\}$;
{UPDATE RULES}
$\mathbf{x}_S = \{\mathbf{x}\ :\ S\ is\ true\}$
$\mathbf{t}_S = \{\mathbf{t}\ :\ S\ is\ true\}$
$l_S \rightarrow$ length of S
$\mathbf{R}_x = \frac{[\mathbf{x}_S\ \ \mathbf{1}][\mathbf{x}_S\ \ \mathbf{1}]^\mathbf{T}}{l_S}$
$\mathbf{m}_x = \frac{[\mathbf{x}_S\ \ \mathbf{1}]\mathbf{t}_S^\mathbf{T}}{l_S}$
$res = \mathbf{R}_x^+\mathbf{m}_x$
$\mathbf{w} = res(1:n)$
$b = res(n+1)$
{TERMINATING CONDITION}
if (misclassified=0 **or** l_S=0 **or** $norm(\mathbf{w}) \geq threshold$)) **then**
break
end if
end for
{TESTING}
$\mathbf{y} = \mathbf{w}^T\mathbf{x} + b$
return $\mathbf{y}$

One way to achieve our goal the optimization of J_{MSS} through the Karush-Kuhn-Tucker (KKT) conditions. The full theoretical analysis can be found in [4]. In short, we compute the gradient of J_{MSS} w.r.t. $\mathbf{w}$, and b

$$\mathbf{g}_w = \mathbf{R}_x\mathbf{w} + b\mathbf{m}_x - \mathbf{m}_{tx} \quad (7)$$

$$g_b = \mathbf{m}_x^T\mathbf{w} + b - \mathbf{m}_t \quad (8)$$

where $\mathbf{R}_x = \bar{E}\{\mathbf{x}\mathbf{x}^T \mid 1 > t[\mathbf{w}^T\mathbf{x} + b]\}$, $\mathbf{m}_{tx} = \bar{E}\{t\mathbf{x} \mid 1 > t[\mathbf{w}^T\mathbf{x} + b]\}$, $\mathbf{m}_x = \bar{E}\{\mathbf{x} \mid 1 > t[\mathbf{w}^T\mathbf{x} + b]\}$, $\mathbf{m}_t = \bar{E}\{t \mid 1 > t[\mathbf{w}^T\mathbf{x} + b]\}$. Setting the gradient equal to zero we obtain

$$\begin{bmatrix} \mathbf{w}^* \\ b^* \end{bmatrix} = \begin{bmatrix} \mathbf{R}_x & \mathbf{m}_x \\ \mathbf{m}_x^T & 1 \end{bmatrix}^+ \begin{bmatrix} \mathbf{m}_{tx} \\ \mathbf{m}_t, \end{bmatrix} \quad (9)$$

where $+$ stands for pseudo-inverse matrix.

The linear slackmin algorithm is Algorithm 1. It is stated that n is the number of features.

2.3. The kernel trick

In order to facilitate the computation of nonlinear separating surfaces we can use a nonlinear mapping $\mathbf{\Phi} : \mathbb{R}^n \rightarrow \mathbb{R}^m$

of the input vectors $\mathbf{x}$ into an m-dimensional space, where $m > n$ (sometimes $m = \infty$). Let us define

$$S = \{i \ : \ t(i)y(i) < 1\}, \tag{10}$$

The novelty of the kernel extension is that we may retain a subset G of S, i.e. we may select just $|G|$ training feature vectors among those that fail to satisfy inequality (2). In other words, our aim is to fix an upper limit to the cardinality $|G|$ and form $G \subset S$ by picking at most $|G|$ elements out of S. This is a well justified selection since the classifier utilizes only the support vectors, that is a limited number of the initial N training vectors. The latter is in accordance with the SVM theory. Actually $|G|$ may be much smaller than $|S|$. The advantages of the aforementioned choice are two-fold

- Less *computational time* is needed since the size of the linear system to be solved in each iteration is limited.
- Less *memory* is needed which is especially important for real datasets which tend to be large-scale ones.

There is an amplitude of ways that G may be formed. For this paper, we choose to propose a "first come-first kept" method, i.e. we retain the *first* $Z_{\mathbf{a}}$ training feature vectors that belong to S. In other words, the cardinality of G is $Z_{\mathbf{a}}$. Here, $Z_{\mathbf{a}}$ initially equals n and it may reach a maximum value $Z_{\mathbf{a}}\_max$ through a scale up factor $Z_{\mathbf{a}}\_scale$.

In this case, it is true that the separating vector $\mathbf{w}$ in $\mathbb{R}^m$ is a linear combination of the mapped inputs

$$\mathbf{w} = \sum_{j \in G} a(j)\mathbf{\Phi}(\mathbf{x}(j)). \tag{11}$$

Notice that the summation is done over G instead of S. We shall avoid the explicit computation of $\mathbf{\Phi}()$ using the scalar kernel function $K(\mathbf{x}, \mathbf{y}) = \mathbf{\Phi}(\mathbf{x})^T\mathbf{\Phi}(\mathbf{y})$. Therefore, the output is defined as

$$y(i) = \sum_{j \in G} K(\mathbf{x}(i), \mathbf{x}(j))a(j) + b, \tag{12}$$

Then based on [4], by setting $b = 0$, Eq. (9) in conjunction with Eq. (11) becomes

$$\mathbf{a} = \mathbf{K}^+\mathbf{t}_G, \tag{13}$$

where

$$\mathbf{a} = [a(i)]_{i \in G} \in \mathbb{R}^{|G| \times 1}, \tag{14}$$

$$\mathbf{K} = [K_{ij}]_{i \in G, j \in G} = [\mathbf{k}_i]^T_{j \in G} \in \mathbb{R}^{|G| \times |G|}, \tag{15}$$

$$\mathbf{t}_G = [t(i)]_{i \in G} \in \mathbb{R}^{|G| \times 1}. \tag{16}$$

The proposed slackmin algorithm for the kernel case is summarized in Algorithm 2.

Algorithm 2 Slackmin algorithm (kernel case)

Input: $MAX\_ITERATIONS$, $Z_{\mathbf{a}}\_max$, $Z_{\mathbf{a}}\_scale$
{INITIALIZATION}
Initialize $\mathbf{a} = \mathbf{0}$, b=0, and $Z_{\mathbf{a}}$=n
{TRAINING}
for $MAX\_ITERATIONS$ iterations **do**
  $G = [\,]$
  for p=1 TO N **do**
    $\mathbf{k}_T = [K_{pi}], i = 1, \cdots, N$
    $\mathbf{y}(p) = \mathbf{k}_T\mathbf{a} + b$
    $S = \{p \ : \ 1 > \mathbf{t}(p)\mathbf{y}(p)\}$;
    if S is true **then**
      $l_S \rightarrow$ length of S
      if $l_S \leq Z_{\mathbf{a}}$ **then**
        $G = [G \ p]$
      else
        $increaseZ_{\mathbf{a}}flag$ = true
      end if
    end if
  end for
  Compute $\mathbf{K} = [K_{ij}]_{i \in G, j \in G}$
  if $increaseNZ_{\mathbf{a}}flag$ = true **then**
    $Z_{\mathbf{a}} = min((Z_{\mathbf{a}}\_scale \times Z_{\mathbf{a}}), Z_{\mathbf{a}}\_max)$
  end if
  {UPDATE RULES}
  $\mathbf{a} = \mathbf{K}^+\mathbf{t}_G$
  {TERMINATING CONDITION}
  if (misclassified=0 **or** l_S=0 **or** $norm(\mathbf{a}) \geq threshold$) **then**
    break
  end if
end for
{TESTING}
$\mathbf{K}' = [K_{ij}]_{i \in N, j \in N}$
$\mathbf{y} = \mathbf{K}'\mathbf{a} + b$
return $\mathbf{y}$

3. EXPERIMENTAL EVALUATION

3.1. Databases

In our experiments we used the adult data set [5] available at the UCI machine learning repository. It is among the larger datasets, including 48842 samples (unknown values are not removed). The task is to predict if the income of a person is greater than 50K based on 14 census parameters, such as age, education, marital status, sex, occupation, work class, and so forth.

An additional dataset, related to classification of pathologies of the vertebral column is considered [5]. The dataset contains values for six biomedical features derived from the shape and orientation of the pelvis and lumbar spine. The aforementioned biomedical features are utilized to classify orthopaedic patients into 2 classes normal (100 patients) or abnormal (210 patients). The dataset is one of the most recent

Table 1. Experimental results for slackmin algorithm (linear kernel)

Dataset	Iterations	Train Set						Test set					
		time elapsed (s)	confusion matrix	$accuracy$	PRC	RCL	F_1	time elapsed (s)	confusion matrix	$accuracy$	PRC	RCL	F_1
vertebral column	10	0.136	62 21 17 148	84.677%	74.699%	78.481%	76.543%	0.001	20 5 1 36	90.323%	80.000%	95.238%	86.957%
adult	10	0.391	4875 1701 4514 27984	84.094%	74.133%	51.922%	61.071%	0.006	1189 430 1109 7040	84.244%	73.440%	51.741%	60.710%

additions to UCI machine learning repository.

3.2. Experimental Results

During performance evaluation 80% of the samples of each dataset are retained for training and the remaining 20% for testing. Two scenarios are exploited to assess the efficiency of the proposed approach: the linear case and the polynomial kernel. For the second case, power and offset are user-defined parameters.

Classifier performance is evaluated through several sets of figures of merit to facilitate future comparisons. Let us define as tp as true positive, fn as false negative, fp as false positive, and tn as true negative. Then, is is true that $accuracy = 100 \times (tp + tn)/(tp + tn + fp + fn)$, $PRC = 100 \times tp/(tp + fp)$, $RCL = 100 \times tp/(tp + fn)$, and $F_1 = (2 \times PRC \times RCL)/(PRC + RCL)$. Detailed results are available in Table 1 and Table 2 for the linear case and the polynomial kernel, respectively. It is stated that the columns of the confusion matrix correspond to the actual label and the rows to the predicted one. All experiments were performed on a 2.67 MHz processor with 4GB of RAM, with a Windows-7 32 bit operating system. The software platform exploited is MATLAB®R2010a.

3.3. Discussion

As it is obvious from Table 1 and Table 2, the algorithm exhibits advantages with respect to computation time, scalability, and efficiency. With respect to computation time, experiments were performed on a 2.67 MHz processor with 4GB of RAM, with a Windows-7 32 bit operating system. Focusing on the linear case, the time for training and testing the classifier demonstrates a magnitude of fractions of seconds. To provide some insight to related literature, according to [6] the execution time for a linear SVM equals 5 s for the adult dataset. It is also true that the proposed algorithm converges quickly, as it is depicted in Figure 1 and in Figure 2 for the linear case of the vertebral column and the adult dataset respectively. Referring to scalability, the adult dataset is considered to be a large-scale dataset, containing a total of 48842 samples, whereas the vertebral column dataset consists of 310 samples, proving the proposed algorithm's ability to handle effectively a limited number of training data.

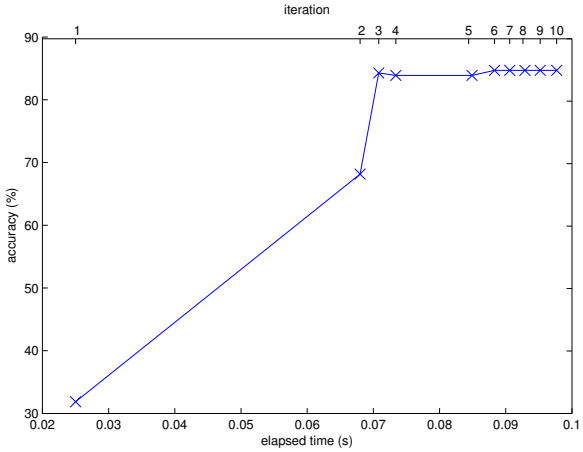

Fig. 1. Elapsed time (s) per iteration with respect to $accuracy(\%)$, for the training phase of slackmin linear algorithm (vertebral column dataset).

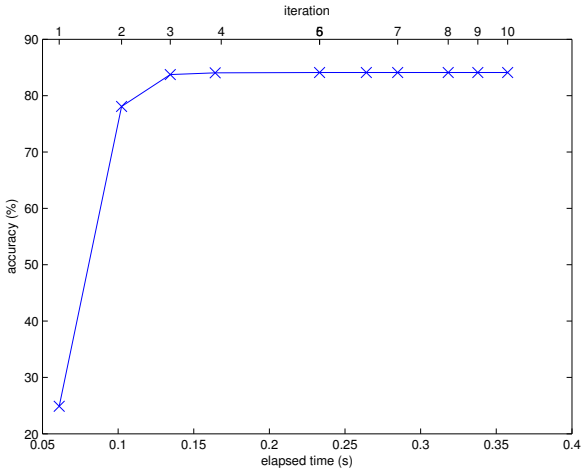

Fig. 2. Elapsed time (s) per iteration with respect to $accuracy(\%)$, for the training phase of slackmin linear algorithm (adult dataset).

Table 2. Experimental results for slackmin algorithm (polynomial kernel)

Dataset	Offset	Power	Iterations	$Z_{\mathbf{a}}\_scale$	$Z_{\mathbf{a}}\_max$	Train Set						Test set					
						time elapsed (s)	confusion matrix	$accuracy$	PRC	RCL	F_1	time elapsed (s)	confusion matrix	$accuracy$	PRC	RCL	F_1
vertebral column	2	2	20	1.5	2000	1.296	62 17 17 152	86.290%	78.481%	78.481%	78.481%	0.002	19 3 2 38	91.935%	86.364%	90.476%	88.372%
adult	2	2	15	2.6	2500	1568.249	5374 2020 4011 27669	84.565%	72.681%	57.262%	64.057%	6.973	1312 480 990 6986	84.951%	73.214%	56.994%	64.094%

With respect to efficiency, the proposed algorithm outperforms state-of-art methods. Here, we focus on two most recent papers utilizing the same datasets. In specific, in [7] a novel SVM formulation, namely ν-SP-SVM, is proposed. This method includes constrains that drop the weights associated to irrelevant features. In specific, a fraction ν of the features is retained. The paper presents SVM with 1-norm and 2-norm regularization parameter. The adult dataset has been used for experimentation. During training, 10% of the training dataset is randomly selected. SVMs have been trained 100 times, with different randomly selected training data and their averaged results are reported. In specific, the method accomplishes an accuracy less than 84% both for the 1-norm ν-SP-SVM and the 2-norm ν-SP-SVM. For the described experimental protocol, the standard 2-norm SVM presents an averaged accuracy of 83.67($\pm$0.3)% and the standard 1-norm SVM achieves an averaged accuracy of 84.03($\pm$0.2)%.

Ensemble classifiers' performance is studied in [8]. Fuzzy Adaptive Resonance Theory (ART) and Self Organizing Map (SOM) networks are applied as base classifiers to produce ARTIE (ART networks in Ensembles) and MUSCLE (Multiple SOM Classifiers in Ensembles) models, respectively. A technique based on particle swarm optimization and simulated annealing is proposed for tuning the base classifier's parameters. Both ensemble classifiers are comprised of 10 base classifiers, whereas decisions are made using the majority voting rule. Additionally, a standard ensemble of SVM classifiers is tested, comprising 10 base classifiers that utilize the SMO algorithm along with an RBF kernel. Experimentation is carried out on vertebral column dataset. For ARTIE model the best accuracy is 83.87($\pm$5.89)%, for MUSCLE model 85.81($\pm$9.40)%, and for the standard SVM ensemble 86.55% (no standard deviation is provided).

4. CONCLUSIONS

This paper presents the slackmin algorithm. For the kernel case, time and memory efficiency is optimized by selecting a limited subset of those feature vectors that are misclassified during training. Here, we applied a "first come-first kept"s strategy, i.e. the subset consists of those feature vectors that are first detected to be misclassified.

The algorithm is computationally efficient and fast, since it manages to converge in a few iterations. With respect to efficiency, slackmin algorithm demonstrates improved figures of merit when compared to recent state-of-the-art approaches. The best accuracy for the adult dataset equals 84.951% and for the vertebral column dataset accuracy is 91.935%.

In the future, the proposed system can be exploited as a base classifier of an ensemble system. Additionally, more datasets may be tested, so as to study the performance of the proposed algorithm for diverse classification problems.

5. REFERENCES

[1] T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel methods in machine learning," *Annals of Statistics*, vol. 36, pp. 1171-1220, June 2008.

[2] E. Y. Chang, K. Zhu, H. Wang, H. Bai, J. Li, Z. Qiu, and H. Cui, "PSVM: Parallelizing support vector on distributed computers," in *Proc. 21st Annual Conf. Neural Information Processing Systems*. 2007, vol. 20, pp. 144–152.

[3] C. Cortes and V. Vapnik, "Support-Vector Networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.

[4] K. I. Diamantaras and M. Kotti, "Binary classification by minimizing the mean squared slack," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, 2012.

[5] A. Frank and A. Asuncion, "UCI machine learning repository," 2010.

[6] S. Sivakumari, R. Praveena Priyadarsini, and P. Amudha, "Performace evaluation of SVM kernels using hybrid PSO-SVM," *Int. Journal Artificial Intelligence and Machine Learning*, vol. 9, pp. 19–25, February 2009.

[7] V. Gómez-Verdejo, M. Martínez-Ramón, J. Arenas-García, M. Lázaro-Gredilla, and H. Molina-Bulla, "Support vector machines with constraints for sparsity in the primal parameters," *IEEE Trans. Neural Networks*, vol. 22, no. 8, pp. 1269 –1283, August 2011.

[8] C. L. C. Mattos and G. A. Barreto, "ARTIE and MUSCLE models: Building ensemble classifiers from fuzzy ART and SOM networks," *Neural Computing and Applications*, pp. 1–13, 2011, Online first.