Circuit Implementation of SVM Training

Sergio Decherchi, Paolo Gastaldo, Mauro Parodi, and Rodolfo Zunino

Abstract— **A central issue in computational intelligence is the training phase of a learning machine. In classification problems, in particular, Support Vector Machines are one of the most effective tools. In this work an analog low-complexity circuital implementation is proposed to address the learning stage of SVMs. The circuit is a co-content minimization network based on a suitable SVM formulation embedding bias removal. Moreover the circuit complexity (i.e. the density of the kernel matrix) is effectively controlled by resorting to a proper kernel function. Experimental evidence shows the effectiveness of the proposed approach.**

I. Introduction

Several approaches have been proposed to the effective support of the training process of Support Vector Machines [10]. When learning speed is of paramount importance, one might envision a hardware-based approach to the training strategy, and one should first choose between a digital and an analog approach to the circuit-support strategy.

The aim of this paper is to further characterize and empirically investigate the aspects and the potentialities of a general circuit model based on the co-content minimization [2]. The paper improves on an analog hardware solution to the learning stage of Support Vector Machines recently proposed [3]: in particular various datasets are considered in order to study the trade-off between circuital complexity and classification accuracy.

II. SVM

Binary-classification systems ascribe input 'test' patterns to either of two possible classes, and Support Vector Machines [10] have been proved widely as powerful tools to tackle such a class of problems. Given a set of n_p patterns $Z=\{ (\mathbf{x}_l, y_l); l=1,..,n_p; y_l \in \{-1,+1\}\}$, an SVM finds the optimal hyperplane $h(\mathbf{x}_i) = \mathbf{w}^T \mathbf{x}_i + b$ that separates the two classes; the corresponding decision function clearly is $f(\mathbf{x}_i) = sign(h(\mathbf{x}_i))$. Such a result is obtained by setting up a conventional quadratic problem, which is typically transformed into its dual version.

This work was partially supported by the University of Genoa under Grant "Accademic research projects Area 09, 2007"

The authors are with University of Genoa, Italy, Department of Biophysical and Electronic Engineering, Via Opera Pia 11/A. (e-mail {sergio.decherchi, paolo.gastaldo, mauro.parodi, rodolfo.zunino} @unige.it)

The dual problem is[1]:

$$\min_{\alpha} \Im(\boldsymbol{\alpha}) \text{ with } \Im(\boldsymbol{\alpha}) \stackrel{def}{=} \frac{1}{2}\boldsymbol{\alpha}^T Q \boldsymbol{\alpha} - \sum_{i=1}^{n_p} \alpha_i$$

$$\text{subject to: } \begin{cases} \sum_{i=1}^{n_p} \alpha_i y_i = 0 \\ 0 \le \alpha_i \le \Omega \qquad i = 1,...,n_p \end{cases} \tag{1}$$

where each element $q_{ij} = y_i y_j (x_i, x_j)$ of the matrix $\mathbf{Q}$ involves the inner product between a pair of input patterns.

The well-known kernel trick [10] allows SVMs to deal with arbitrary distributions that may not be linearly separable in the data space. Inner products are handled in a high-dimensional Hilbert space, yet disregarding the specific mapping of each single pattern. Thus the elements of the Hessian matrix $\mathbf{Q}$ can still be expressed as $q_{ij} = y_i y_j k_{ij}$, where $k_{ij} = K(x_i, x_j)$ and $K()$ expresses the dot product in a Hilbert space. When a solution vector $\boldsymbol{\alpha}$ is obtained then the decision function can be expressed as:

$$f(\mathbf{x}_j) = sign\left(\sum_{i=1}^{n_p} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}_j) + b \right) \tag{2}$$

The SVM training formulation (1) implies a constrained quadratic programming problem (CQP) on a convex cost function. The major practical advantage of this property is that polynomial-complexity Quadratic Programming (QP) algorithms [1] ensure convergence to the global minimum.

When the number of free parameters in (1) becomes huge (e.g. $>10^4$ patterns), several decomposition algorithms have been proposed to cope with the optimization problem [1]. In such cases, it seems interesting to envision a hardware solution to the problem (1), especially when considering computational efficiency.

An approach to map SVM learning stage on an analog circuit was proposed in [4], and was characterized by a direct mapping of the SVM learning process into Chua's circuit [2]. The main drawback of that method was the circuit complexity in mapping the linear constraint in (1), which was replaced by two inequalities:

$$\sum_{i=1}^{np} y_i \alpha_i \ge 0 \text{ and } \sum_{i=1}^{np} y_i \alpha_i \le 0 \tag{3}$$

[1] the classical regularization parameter C will be denoted by Ω to avoid confusion with capacitances in the following of the paper.

978-1-4244-3553-1/09/$25.00 ©2009 IEEE

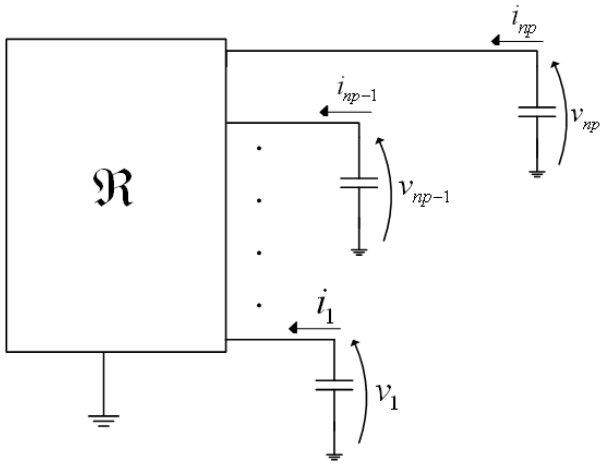

Fig. 1. Multi-terminal resistive network connected to capacitors

Such a formulation is formally correct but might bring about some issues in reaching a circuit stable state.

The present paper shows that by a reformulation of (1) co-content networks can apply effectively to the circuit-supported optimisation goal.

III. Co-Content Minimization Circuits

The basic principle underlying the co-content approach is that the minimum of a given functional can be found by means of a proper analog circuit. Let $\Re$ be a voltage-controlled, reciprocal, multiterminal resistor; for such component (either linear or nonlinear) one can formalize the so-called co-content potential function, $\Gamma(\mathbf{v})$, as

$$\Gamma(\mathbf{v}) = \int^{\mathbf{v}} \mathbf{i}(\xi) \cdot d\xi \tag{4}$$

where **v, i** denote the vectors of voltages and currents respectively, taken as shown in Fig.1. The current vector, **i**, can be written as the gradient of the co-content function:

$$\mathbf{i}(\mathbf{v}) = \nabla\Gamma(\mathbf{v}) \tag{5}$$

In the circuit obtained by connecting $\Re$ to a set of linear capacitors (Fig. 1), the voltages in **v** encode the state variables of the circuit. If one denotes as **C** the diagonal matrix containing the (possibly different) capacitance values, theory shows [11] that the time progression of $\Gamma(t)$ is ruled by the equation:

$$\frac{d\Gamma}{dt} = (\nabla\Gamma(\mathbf{v}))^t \frac{\partial \mathbf{v}}{\partial t} = -\left(\frac{\partial \mathbf{v}}{\partial t}\right)^t \mathbf{C} \frac{\partial \mathbf{v}}{\partial t} \le 0 \tag{6}$$

Expression (6) points out that, for any initial condition **v(0),** $\Gamma(t)$ points toward a minimum and that the corresponding stationary value of the current vector in the circuit is $\mathbf{i} \equiv \mathbf{0}$.

This property suggests that one could minimize an arbitrary functional, $\Im$, by mapping it on the co-content, $\Gamma(\mathbf{v})$, of a proper resistive multi-terminal component, connected to a set of linear capacitors as per Fig.1. The main structural idea is to bypass digital-based computations in the functional minimization, by directly using the intrinsic computational capabilities offered by the physical laws of circuits [2]. Since the solution is reached in real time as soon as the kernel matrix is computed, the time needed for SVM training reduces considerably respect software implementations.

So the advantage of this approach mainly consists in the intrinsic parallel processing. This technique can be very useful in 'early learning' problems, requiring that a machine exhibits an extremely fast learning performance. Whenever the kernel matrix changes, modifications can be mapped via reprogramming of resistive network. The following Section describes the procedure to map Support Vector Machines to a proper co-content network.

IV. Hardware SVM Training

A. Circuit design strategy

The circuit complexity brought about by the linear constraint in (1) is handled by a reformulation of the SVM cost function. Theory shows [5] that, in the presence of a positive definite kernel function, the linear constraint in eq. (1) can be removed without affecting the generalization ability of the SVM. Indeed, forcing a null bias term *b* in the decision function (2) implies to take out the linear constraint in the dual problem formulation (1) [10]. This rewriting modifies the SVM functional as follows:

$$\min_{\alpha} \Im(\boldsymbol{\alpha}) \text{ with } \Im(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^T Q \boldsymbol{\alpha} - \sum_{i=1}^{n_p} \alpha_i \tag{7}$$
$$\text{subject to: } \quad 0 \le \alpha_i \le \Omega \qquad i = 1,...,n_p$$

Such a simplified formulation of the original problem holds for positive definite kernels. From a circuit perspective, it is interesting to anticipate that the "box constraints" on the parameters $\boldsymbol{\alpha}$ prevent the parameters themselves to reach unfeasible values; in other words, both constraints imply useful physical limitations on the voltage values of the associate vector coordinates **v**. The resulting design of a circuit mapping a functional, $\Im(\boldsymbol{\alpha})$, into its co-content potential, $\Gamma(\mathbf{v})$, follows two basic steps:

1. Setting up a correspondence between vectors $\boldsymbol{\alpha}$ and **v** (association step);
2. Setting up a topological structure for $\Re$ (circuit definition)

B. Association step

First, one sets a reference voltage, V_0, to map SVM parameters α_i into voltages v_i, hence the relation between SVM variables and voltages is defined as $\alpha_i = v_i / V_0$. Thus, the box constraint takes the form: $0 \le v_i \le \Omega V_0 \quad i = 1,...,n_p$, while the quadratic functional can be rewritten as

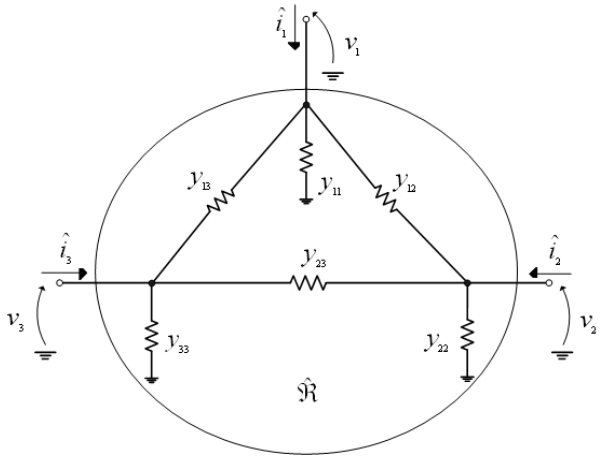

Fig. 2. Circuit topology: three terminal case

$$\frac{1}{2}\boldsymbol{\alpha}^t\mathbf{Q}\boldsymbol{\alpha}-\sum_{i=1}^{np}\alpha_i \;\rightarrow\; \frac{1}{2V_0^2}\mathbf{v}^t\mathbf{Q}\mathbf{v}-\frac{1}{V_0}\sum_{i=1}^{np}v_i \tag{8}$$

By choosing an arbitrary resistance value, R_0, and multiplying the dimensionless functional (8) by the power reference V_0^2/R_0, one defines the actual co-content potential function $\Gamma(\mathbf{v})$, as

$$\Gamma(\mathbf{v})=\frac{1}{2R_0}\mathbf{v}^t\mathbf{Q}\mathbf{v}-\frac{V_0}{R_0}\sum_{i=1}^{np}v_i \overset{def}{=} \Psi(\mathbf{v})-\Phi(\mathbf{v}) \tag{9}$$

The physical dimension of both Ψ and Φ is power, and is consistent with the co-content potential function terms. By taking the partial derivates of Ψ with respect to voltages v_i, one obtains the current terms:

$$\hat{i}_k=\frac{\partial\Psi}{\partial v_k}=\frac{1}{R_0}\sum_{i=1}^{np}q_{ki}v_i \tag{10}$$

Likewise, partial derivatives of Φ yield:

$$\tilde{i}_k=\frac{\partial\Phi}{\partial v_k}=-\frac{V_0}{R_0} \tag{11}$$

Finally by assembling $\nabla\Gamma(\mathbf{v})$ one obtains total currents; in particular, the current at the k-th terminal is:

$$i_k=\hat{i}_k+\tilde{i}_k. \tag{12}$$

C. Circuit definition

A constant current source can easily support the second term $\tilde{i}_k$ in the expression (12). The first term $\hat{i}_k$, because $\mathbf{Q}$ matrix is positive definite, can be implemented by a reciprocal circuit $\hat{\Re}$ containing two-terminal resistors only. The topological structure is not unique. A possible choice is sketched in Fig.2 for the very simple case $n_p=3$. Following [2], the general form of the $n_p\times n_p$ conductance matrix G is:

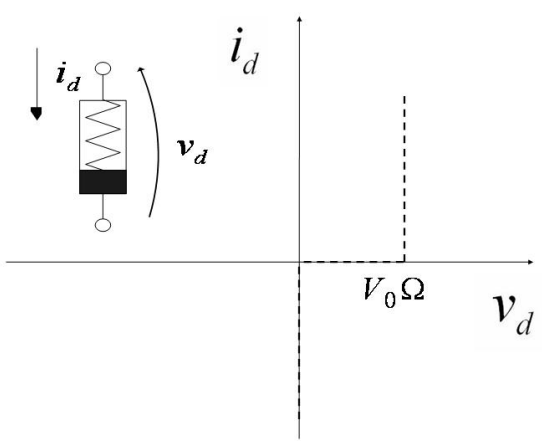

Fig. 3. Voltage limiting component and voltages-current curve

$$\mathbf{G}=\begin{pmatrix}\sum_k y_{1k} & -y_{12} & \cdots & \cdots & -y_{1(np-1)} & -y_{1np}\\ -y_{21} & \sum_k y_{2k} & \cdots & \cdots & -y_{2(np-1)} & -y_{2np}\\ -y_{31} & \cdots & \cdots & \cdots & \cdots & -y_{3np}\\ -y_{41} & -y_{42} & \cdots & \cdots & \cdots & -y_{4np}\\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\ -y_{np1} & -y_{np2} & \cdots & \cdots & \cdots & \sum_k y_{np,k}\end{pmatrix} \tag{13}$$

with $y_{ik}=y_{ki}$ according to the well-known reciprocity theorem. The correspondence between the y_{ij} elements and the elements q_{ij} of $\mathbf{Q}$ is

$$y_{ij}=\begin{cases}-\dfrac{q_{ij}}{R_0}; & i\neq j\\ \dfrac{\sum_k q_{ik}}{R_0}; & i=j\end{cases} \tag{14}$$

as one can find after some manipulations [2]. As a final step, the circuit implementation of the n_p box constraints is obtained through nonlinear resistors with the v_d-i_d characteristic in Fig. 3. As shown in [2], these resistors do not contribute to the co-content of $\Re$. The structure of $\Re$, resulting by connecting the multiterminal $\hat{\Re}$ defined by (14) to these nonlinear resistors and to the current generators (11) is shown in Fig. 4, where a set of identical linear capacitors completes the circuit.

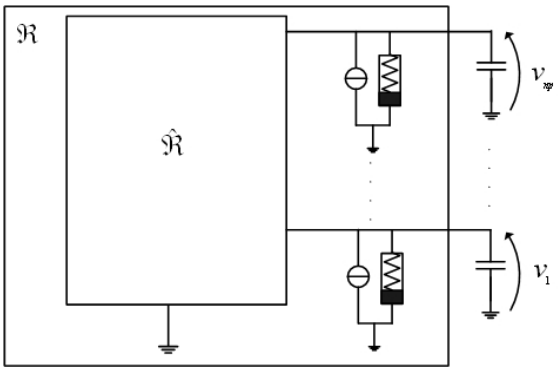

Fig. 4. Complete circuit

D. Complexity reduction

A crucial aspect of the overall approach concerns the structure of the resistive network, since the complexity of $\Re$ depends on the density of the kernel matrix. Using the popular linear or RBF kernel, for instance, tends to yield a dense Hessian matrix **Q**, and ultimately complicates the topology of the supporting network, $\Re$. Generally speaking, the number of two-terminal resistors implementing a full-density matrix Q is equal to $n_p(n_p+1)/2$. By contrast, a proper choice of the kernel function can lead to a sparse **Q** matrix, and reduce the complexity of $\Re$ accordingly.

In general, taking a positive-definite kernel matrix and annihilating its closest-to-zero elements yields a kernel matrix which is no longer positive definite. In other words, the sparse structure of a positive definite matrix must be guaranteed ab origine by a proper choice of the kernel.

A positive definite kernel generating a sparse Q-matrix has been defined in [6]. More specifically, thanks to this kernel the sparsity of **Q** can be tuned through a specific parameter (cut-off distance), according to the following definition.

Let $d(\mathbf{x}_l,\mathbf{x}_m) = \| \mathbf{x}_l - \mathbf{x}_m \|^2$ denote the distance between the pair of points $(\mathbf{x}_l,\mathbf{x}_m)$, and let r be a positive cut-off distance. The kernel-based inner product, $K(\mathbf{x}_l,\mathbf{x}_m)$, is set to 0 each time $d(\mathbf{x}_l,\mathbf{x}_m) \geq 2r$; otherwise, the quantity $K(\mathbf{x}_l,\mathbf{x}_m)$ is worked out by means of a recursive procedure:

$$
\begin{aligned}
k_{n,1}(\mathbf{x}_l,\mathbf{x}_m) &= 1 - \frac{\| \mathbf{x}_l - \mathbf{x}_m \|}{2r} \\
k_{n,2}(\mathbf{x}_l,\mathbf{x}_m) &= \arccos\left(\frac{\| \mathbf{x}_l - \mathbf{x}_m \|}{2r}\right) + \\
&\quad - \frac{\| \mathbf{x}_l - \mathbf{x}_m \|}{2r}\sqrt{1-\left(\frac{\| \mathbf{x}_l - \mathbf{x}_m \|}{2r}\right)^2} \\
k_{n,j}(\mathbf{x}_l,\mathbf{x}_m) &= \frac{j-1}{j} k_{n,j-2}(\mathbf{x}_l,\mathbf{x}_m) + \\
&\quad - \frac{1}{j}\frac{\| \mathbf{x}_l - \mathbf{x}_m \|}{2r}\left(1-\left(\frac{\| \mathbf{x}_l - \mathbf{x}_m \|}{2r}\right)^2\right)^{\frac{j-1}{2}}
\end{aligned}
\qquad (15)
$$

Recursions stop when the running index j reaches the dimension n of the original data space, so one has: $K(\mathbf{x}_l,\mathbf{x}_m) = k_{n,n}(\mathbf{x}_l,\mathbf{x}_m)$. As shown in [6], this kernel is positive definite and ensures good prediction performances while yielding sparse kernel matrices.

V. Experimental Results

The circuit capabilities have been tested through a set of experimental sessions on three known test sets from UCI repository [8]: Sonar, Ionosphere and Diabetes. The assessment of the generalization ability compared the method's performances with those attained by a software-based, high-precision SVM implementation, including a bias term. To measure the effect of using a sparse kernel matrix, different levels of the sparsity-controlling parameter were tested.

The software implementation adopted a conventional SMO-based algorithm, including Lin's first-order heuristic to select a working set [9]. In all experiments, the tolerance on the KKT conditions [9] was 10^{-3}. Circuit-based optimization runs were accomplished first by generating the components netlist in an automated fashion, then by applying Spice (Orcad 16.0) simulations.

The hardware simulations of the voltage-limiting section involved a classic diodes-based limiting circuitry. The model used was the default diode with a constant threshold of 0.594 V; this voltage value tuned the offset to get a precise box constraint when the parameter-representative voltage was 0 or the SVM bounding constant, Ω. Additional circuit-design parameters were set as follows: C = 1nF, V_0 = 1V, $R_0 = 1\,k\Omega$. For a given $\hat{\Re}$, the values of C and R_0 control the rate of convergence of the dynamic process in the circuit. All solution voltages values were measured upon completion of a transient interval of 80 μs. In all the cases, the circuit converged to the solution without instabilities, as expected.

The domain of kernel parameter *r* was set as [0.2, 4] and all attributes of data were normalized in [-1,+1]. Table I

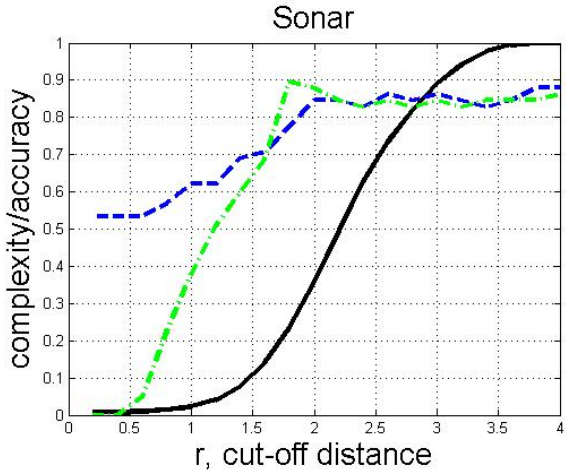

Fig.5 Sonar dataset: dashed line represents the software accuracy, continuous line is the circuital complexity and dashed-dotted line is svm hardware accuracy.

gives, for the various data sets, the splitting strategies of the data into training and test sets, and the associate values of the SVM regularization parameter Ω .

TABLE I
DATASETS SPLITTING AND C PARAMETERS

Dataset	#Training	#Test	Ω
Sonar	150	58	1
Ionosphere	251	100	8
Diabetes	230	538	1

Tables II, III, IV report on the test-set accuracy values and the complexity values for the hardware SVM together with the values obtained with SVM software with bias; fig.5,6,7 are the graphical counterparts of tables II, III, IV.
From an accuracy viewpoint, results suggest that the training circuitry reaches a satisfactory solution in all three cases.
However the most interesting aspect here is that a good level of accuracy (>75%) can be maintained also when the complexity of the resistor network $\hat{\Re}$ is severely limited through a proper value of the cut-off distance parameter *r*.

In particular:

- in the case of Sonar dataset with a sparsity level of 80% the accuracy is unchanged respect the full matrix training
- in Ionosphere an acceptable accuracy is obtained at a level of sparsity of 80%
- in Diabetes a level of sparsity of 70% can guarantee the maximum accuracy obtainable with full matrix

Note that the degrading results obtained when sparsification is near 100% are due to the absence of the bias term *b*. For *b=0* the decision function $sign\left(\mathbf{w}^T\mathbf{x}_i + b\right)$ can take uncorrect values when $\mathbf{w}^T\mathbf{x}_i$ is close to zero. In these cases, for a fair evaluation, the result has been considered as an error. Then the accuracy results reported here must be interpreted as the most conservative evaluations.

TABLE II
COMPARISON FOR SONAR DATASET

r	Complexity	SW SVM	HW SVM
0.2	0.007	0.534	0
0.4	0.007	0.534	0
0.6	0.0098	0.534	0.05
0.8	0.014	0.569	0.224
1	0.022	0.621	0.379
1.2	0.04	0.621	0.51
1.4	0.077	0.69	0.6
1.6	0.141	0.707	0.689
1.8	0.236	0.776	0.896
2	0.358	0.845	0.879
2.2	0.496	0.845	0.845
2.4	0.628	0.828	0.828
2.6	0.735	0.862	0.845
2.8	0.819	0.845	0.828
3	0.891	0.862	0.845
3.2	0.942	0.845	0.828
3.4	0.977	0.828	0.845
3.6	0.9944	0.845	0.845
3.8	0.9988	0.879	0.845
4	1	0.879	0.862

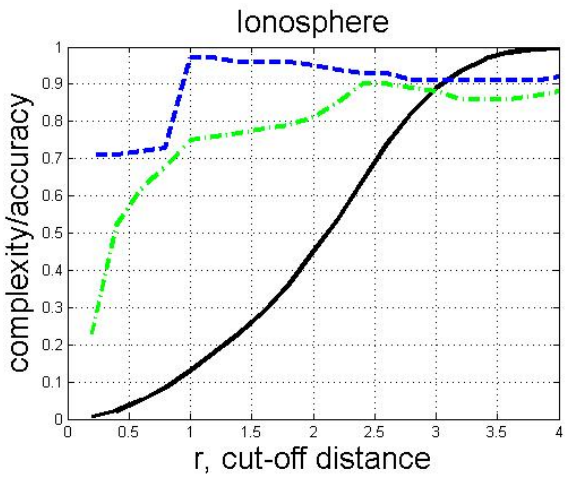

Fig.6 Ionosphere dataset: dashed line represents the software accuracy, continuous line is circuital complexity and dashed-dotted line is svm hardware accuracy.

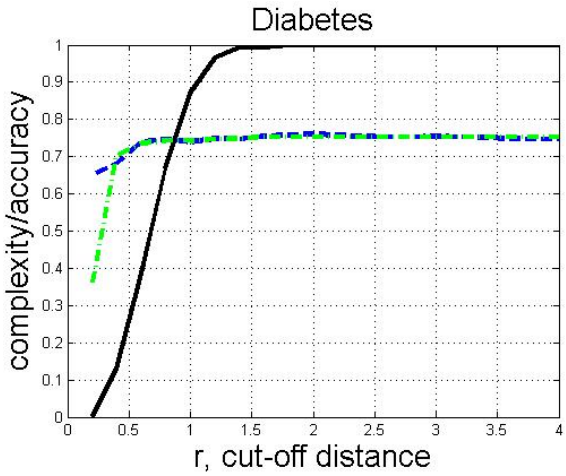

Fig.7 Diabetes dataset: dashed line represents the software accuracy, continuous line is circuital complexity and dashed-dotted line is svm hardware accuracy.

TABLE III
COMPARISON FOR IONOSPHERE DATASET

r	Complexity	SW SVM	HW SVM
0.2	0.007	0.71	0.23
0.4	0.022	0.71	0.52
0.6	0.05	0.72	0.62
0.8	0.0851	0.73	0.68
1	0.13	0.97	0.75
1.2	0.18	0.97	0.76
1.4	0.23	0.96	0.77
1.6	0.29	0.96	0.78
1.8	0.36	0.96	0.79
2	0.45	0.95	0.81
2.2	0.5351	0.94	0.85
2.4	0.638	0.93	0.9
2.6	0.74	0.93	0.9
2.8	0.821	0.91	0.89
3	0.89	0.91	0.88
3.2	0.936	0.91	0.86
3.4	0.967	0.91	0.86
3.6	0.986	0.91	0.86
3.8	0.9945	0.91	0.87
4	0.998	0.92	0.88

TABLE IV
COMPARISON FOR DIABETES DATASET

r	Complexity	SW SVM	HW SVM
0.2	0.002	0.651	0.363
0.4	0.131	0.682	0.703
0.6	0.385	0.74	0.736
0.8	0.67	0.747	0.742
1	0.87	0.74	0.742
1.2	0.966	0.749	0.747
1.4	0.9931	0.747	0.749
1.6	0.995	0.755	0.753
1.8	1	0.757	0.753
2	1	0.76	0.753
2.2	1	0.757	0.753
2.4	1	0.755	0.753
2.6	1	0.753	0.753
2.8	1	0.753	0.753
3	1	0.755	0.753
3.2	1	0.753	0.753
3.4	1	0.751	0.753
3.6	1	0.749	0.753
3.8	1	0.747	0.753
4	1	0.747	0.753

In general the results discussed here appear to be quite encouraging. However it seems to remain a certain dependence of the effectiveness of the sparsification from the chosen dataset. It should be also underlined that the proposed approach can use other sparse kernel representations and that, in every case, the network is able in giving a coherent solution in both the case of sparse or not sparse network.

VI. CONCLUSIONS

In this work a modification of SVM formulation to make it compatible with a co-content minimization network implementation is proposed; in particular bias removal allowed eliminating the linear constraint term.

Further the circuit complexity was strongly reduced and controlled by using an effective sparse kernel function. Experimental evidence confirmed that the circuit approach attained accuracy classification levels comparable with those yielded by high precision software implementation.

The learning skill of the network depends on the value of the kernel parameter, hence a future line of investigation concerns the definition of an operative criterion of selection of *r*. Another possible extension regards the application of other possible sparse kernels presented in literature and the comparison of the obtained results with those obtained in the present case. As a final step a concrete prototypal VLSI implementation of the network will be developed.

REFERENCES

[1] T. Joachims, "Making Large-Scale SVM Learning Practical". In: Advances in Kernel Methods Support Vector Learning, B. Schölkopf, C. Burges, and A. Smola (ed.), MIT Press, 1999.

[2] L.O. Chua, ,Lin, G.N "Non linear programming without computation", IEEE Trans. on Circuits and Systems, 1984, 31, pp. 182-188.

[3] S. Decherchi, P. Gastaldo, M. Parodi, R. Zunino, "Low-complexity linear circuit implementation of Support Vector Machines training", Electronic Letters, 44(25), Dec, 2008, pp. 1478 - 1479

[4] D. Anguita, S. Ridella, S. Rovetta "Circuital implementation of Support Vector Machines", Electronics Letters, 1998, 34, pp. 1596-1597.

[5] T. Poggio, S. Mukherjee, R. Rifkin, A. Rakhlin,, A. Verri: "B", 2001, AI Memo 2001-011 CBCL Memo 198, Massachusetts Institute of Technology.

[6] S. Boughorbel, J. Tarel, , F. Fleuret, N. Boujemaa "GCS Kernel For SVM-Based Image Recognition". Proc. Int. Conf. on Artificial Neural Networks, ICANN 2005, Sept. 2005, Warsaw, vol. II, pp. 595-600.

[7] T. Evgeniou, M. Pontil, T. Poggio "Regularization Networks and Support Vector Machines" Advances in Computational Mathematics, Springer Netherlands 13(1) April, 2000.

[8] Hettich, S. and Bay, S. D. (1999). The UCI KDD Archive [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Department of Information and Computer Science

[9] C.C.Chang, C.J. Lin . "LibSVM: a library for Support Vector Machines" [http://www.csie.ntu.edu.tw/~cjlin/papers/libsvm.pdf]

[10] Cortes C, Vapnik VN, "Support vector networks", 1995, Machine Learning, 20:273—297.

[11] L. O. Chua, C. A. Desoer, and E. S. Kuh, "Linear and Nonlinear Circuits". New York: McGraw-Hill, 1987.