Neurocomputing 86 (2012) 193–198

Contents lists available at SciVerse ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Letters

A compact neural network for training support vector machines ☆

Yun Yang [a], Qiaochu He [b], Xiaolin Hu [c,*]

[a] Department of Mathematical Science, Tsinghua University, Beijing 100084, China
[b] Department of Automotive Engineering, Tsinghua University, Beijing 100084, China
[c] State Key Laboratory of Intelligent Technology and Systems, Tsinghua National Laboratory for Information Science and Technology (TNList), Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China

ARTICLE INFO

Article history:
Received 30 April 2011
Received in revised form
22 November 2011
Accepted 22 January 2012
Communicated by H. Jiang
Available online 24 February 2012

Keywords:
Neural network
Support vector machine
Quadratic programming
Analog circuits

ABSTRACT

An analog neural network architecture for support vector machine (SVM) learning is presented in this letter, which is an improved version of a model proposed recently in the literature with additional parameters. Compared with other models, this model has several merits. First, it can solve SVMs (in the dual form) which may have multiple solutions. Second, the structure of the model enables a simple circuit implementation. Third, the model converges faster than its predecessor as indicated by empirical results.

1. Introduction

Engineering applications of artificial intelligence often require real-time solutions [1,2]. By employing artificial neural networks based on analog circuits implementation, the computing procedures are physically parallelled and distributed. After the pioneering work in this field by Hopfield and Tank [3,4], tremendous interests have been aroused for designing neural networks with analog circuits implementation in a variety of engineering applications (see [5–14] and references therein).

Support vector machines (SVMs) are widely used tools for classification [15] and regression [16]. It can be modeled as a quadratic programming (QP) problem, and therefore can be solved by some recurrent neural networks capable of solving this type of optimization problems, e.g., [17–20]. Specifically, [21] presents a one-layer recurrent neural network and [22] presents a simpler model in terms of circuits implementation. Both of the two networks converge to steady-states corresponding to the solutions of the SVMs under some conditions including:

- the Hessian of the objective function is positive definite; or
- the Hessian is positive semidefinite but the solution is unique.

In other words, the QP formulation has to be strictly convex, which may not be the case in many applications (see Remark 1). There exist some other networks which can potentially solve SVMs without strict convexity assumption (e.g., [18–20]), but none of them is as simple as the model in [22]. Then, is it possible to design a neural network that has this nice property but is very simple in structure?

In this letter, we present such a neural network for SVM learning. It is actually a model in [20] with additional parameters. Interestingly, this modification enables a very much simpler circuits implementation, which is comparable to the model in [22].

2. Architecture of the model

2.1. Preliminaries

Suppose that there are N training points for classification, where each input $\mathbf{z}_i \in \mathcal{R}^M$ is in one of the two classes $y_i = +1$ and $y_i = -1$, i.e., the training data is $(\mathbf{z}_i; y_i)$ for $i = 1, \ldots, N$. It is well-known that the support vector machine (SVM) training problem for classification can be formulated as a convex quadratic programming (QP) problem [15]

$$\begin{aligned} \min \quad & -\sum_{i=1}^{N} \alpha_i + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j q_{ij} \\ \text{s.t.} \quad & \sum_{i=1}^{N} \alpha_i y_i = 0, \quad 0 \le \alpha_i \le h, \ \forall i = 1, \ldots, N, \end{aligned} \tag{1}$$

☆The work was supported by the National Basic Research Program (973 Program) of China under Grant 2012CB316301, National Natural Science Foundation of China under Grants 60805023 and 61134012, and Basic Research Foundation of Tsinghua National Laboratory for Information Science and Technology (TNList).

* Corresponding author.
E-mail address: xlhu@tsinghua.edu.cn (X. Hu).

0925-2312/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2012.01.004

where $q_{ij} = y_i y_j K_{ij}(\mathbf{z}_i, \mathbf{z}_j)$ and $K_{ij}(\mathbf{z}_i, \mathbf{z}_j)$ is the so-called kernel function which can take various forms, e.g., $K_{ij}(\mathbf{z}_i, \mathbf{z}_j) = (\mathbf{z}_i^T \mathbf{z}_j + 1)^p$ with p an integer. The kernel function must satisfy Mercer's condition and the $N \times N$ dimensional matrix $\mathbf{Q} = \{q_{ij}\}$ must be positive semidefinite, which implies that the problem is convex. The parameter $h > 0$ is a user-defined constant to control the tradeoff between the maximization of the margin and the minimization of errors. The above problem is referred to as the dual formulation of the SVM.

Remark 1. In many applications, $\mathbf{Q}$ may not be positive definite. For instance, in linear SVM, $K_{ij} = \mathbf{z}_i^T \mathbf{z}_j$ and $\mathbf{Q}$ can be written as AA^T where $A = (y_1 \mathbf{z}_1, y_2 \mathbf{z}_2, \ldots, y_N \mathbf{z}_N)^T \in \mathcal{R}^{N \times M}$. When $N > M$, which is often the case in practice, $\text{rank}(\mathbf{Q}) < N$ and $\mathbf{Q}$ is positive semidefinite only. For another instance, it is easy to show that when there are repeated samples in the training set, $\mathbf{Q}$ is singular and thus positive semidefinite only.

2.2. Existing models

In this subsection, we briefly review a few state-of-the-art models for solving the SVM problem. A typical one-layer recurrent neural network is presented in [21] with the following dynamic equations:

$$\tau \dot{x}_i = -x_i + P\left(\sum_{j=1}^{N} (1 - q_{ij}) x_j - y_i \mu + 1\right), \quad \forall i = 1, \ldots, N,$$

$$\tau \dot{\mu} = \sum_{i=1}^{N} y_i x_i, \tag{2}$$

where $\tau > 0$ is a time constant, P is a projection operator (activation function) defined as follows

$$P(x) = \begin{cases} 0, & x \le 0, \\ x, & 0 < x < h, \\ h, & x \ge h, \end{cases}$$

and x_i and μ denote the states of the network, which are time-varying. In fact, x_i corresponds to the variable α_i in (1). Circuit implementation of this network follows the idea that each operator in this model can be implemented by a circuit module, e.g., an op-amp.

Another SVM network is presented in [22]. The dynamic equations are as follows:

$$\tau \dot{x}_i = 1 - \sum_{j=1}^{N} q_{ij} \alpha_j - y_i \mu + d_i(\alpha_i - x_i), \quad \forall i = 1, \ldots, N,$$

$$\tau \dot{\mu} = \sum_{i=1}^{N} y_i \alpha_i, \tag{3}$$

where $d_i > 0$ for all $i = 1, \ldots, N$, $\alpha_i = P(x_i)$ and other notations are the same as in (2). Note that, here it is α_i instead of x_i that corresponds to the variables of (1). Similar to the network (2), this network can be implemented by circuit modules, too. Its major advantage over the network (2) is that if $d_i = 2 + \sum_{j=1}^{N} |q_{ij}|$, it can be implemented by a very simple circuit, which takes advantage of the nonlinear properties of op-amps.

In [20], a neural network was proposed for solving variational inequalities and convex optimization problems. When tailored for solving (1), it is governed by the following dynamic equations:

$$\tau \dot{x}_i = 1 - \sum_{j=1}^{N} (q_{ij} + y_i y_j) \alpha_j - y_i \mu + \alpha_i - x_i, \quad \forall i = 1, \ldots, N,$$

$$\tau \dot{\mu} = \sum_{i=1}^{N} y_i \alpha_i, \tag{4}$$

where $\alpha_i = P(x_i)$.

In terms of performance, the network (4) is superior to (2) and (3) because when the Hessian matrix $\mathbf{Q}$ is positive semidefinite, only (4) can guarantee the global convergence to solutions of the SVM (1). The other two require positive definiteness of $\mathbf{Q}$, or positive semidefiniteness of $\mathbf{Q}$ plus uniqueness of the solution. Therefore, the neural network (4) has broader applications than the other two. However, its structure is not as simple as (3). Note that there exist other neural networks capable of solving positive semidefinite SVMs (e.g., [18]), but their structures are not as simple as (3) either.

2.3. A revised model

Let us revise the model (4) by adding a constant $d_i > 0$ for $i = 1, \ldots, N$, then the dynamic equations for the revised model are

$$\tau \dot{x}_i = 1 - \sum_{j=1}^{N} (q_{ij} + y_i y_j) \alpha_j - y_i \mu + d_i(\alpha_i - x_i), \quad \forall i = 1, \ldots, N,$$

$$\tau \dot{\mu} = \sum_{i=1}^{N} y_i \alpha_i, \tag{5}$$

where $\alpha_i = P(x_i)$.

If we set $d_i = 2 + \sum_{j=1}^{N} |q_{ij} + y_i y_j|$, there exists a very simple circuit for realizing this model as shown in Fig. 1, where (a) shows the block diagram and (b) shows the circuit realization scheme. Note that only the $\dot{x}_i$ equation is shown in Fig. 1(b). Here we set V_s slightly lower than V_{CC}, then the equation governing this circuit is

$$R_0 C_0 \dot{u}_i = \frac{V_S}{h} - \sum_{j=1}^{N} (q_{ij} + y_i y_j) v_j - y_i v_\mu + \left(2 + \sum_{j=1}^{N} |q_{ij} + y_i y_j|\right)(v_i - u_i),$$

where

$$v_i = \begin{cases} 0, & u_i \le 0, \\ u_i, & 0 < u_i < V_s, \\ V_s, & u_i \ge V_s, \end{cases}$$

which is determined by the saturation property of the op-amp. Then v_i, u_i, v_μ and $R_0 C_0$ correspond to α_i, x_i, μ and τ in (5), respectively. For ensuring that the resistance $R_0/(q_{ij} + y_i y_j)$ shown in Fig. 1(b) is always positive, the absolute value of $q_{ij} + y_i y_j$ is used. So, if for some j, $q_{ij} + y_i y_j$ is positive (negative), then the corresponding input voltage to the op-amp should be v_j $(-v_j)$.

In what follows, we will show that the additional constants d_i does not affect the stability property of the network. The analysis is followed by revising the proof for the network (4) presented in [20].

3. Stability analysis

Let $\mathcal{S}^* = \{\boldsymbol{\alpha}^* \in \mathcal{R}^N | \boldsymbol{\alpha}^* \text{ solves } (1)\}$. First, we rewrite (5) in the vector form

$$\begin{cases} \tau \dot{\mathbf{x}} = \mathbf{D}(\boldsymbol{\alpha} - \mathbf{x}) - \mathbf{Q}\boldsymbol{\alpha} + \mathbf{e} - \mathbf{y}(\mu + \mathbf{y}^T \boldsymbol{\alpha}), \\ \tau \dot{\mu} = \mathbf{y}^T \boldsymbol{\alpha}, \end{cases} \tag{6}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_N)^T$, $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_N)^T$, $\mathbf{e} = (1, \ldots, 1)^T$, $\mathbf{y} = (y_1, \ldots, y_N)^T$, $\mathbf{Q} = [q_{ij}]_{N \times N}$, $\mathbf{D} = \text{diag}(d_1, \ldots, d_N)$, $P(\mathbf{x}) = (P(x_1), \ldots, P(x_N))^T$ and the other notations are the same as before.

Let $((\mathbf{x}^*)^T, \mu^*)^T$ denote an equilibrium point of (6). We have the following results.