Neural Networks 49 (2014) 39–50

Contents lists available at ScienceDirect

Neural Networks

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

Biologically relevant neural network architectures for support vector machines

Magnus Jändel *

Swedish Defence Research Agency, SE 164 90 Stockholm, Sweden

ARTICLE INFO

Article history:
Received 8 December 2011
Received in revised form 5 June 2013
Accepted 18 September 2013

Keywords:
Support vector machine
Neural network
Competitive queuing memory
Perceptual learning

ABSTRACT

Neural network architectures that implement support vector machines (SVM) are investigated for the purpose of modeling perceptual one-shot learning in biological organisms. A family of SVM algorithms including variants of maximum margin, 1-norm, 2-norm and ν-SVM is considered. SVM training rules adapted for neural computation are derived. It is found that competitive queuing memory (CQM) is ideal for storing and retrieving support vectors. Several different CQM-based neural architectures are examined for each SVM algorithm. Although most of the sixty-four scanned architectures are unconvincing for biological modeling four feasible candidates are found. The seemingly complex learning rule of a full ν-SVM implementation finds a particularly simple and natural implementation in bisymmetric architectures. Since CQM-like neural structures are thought to encode skilled action sequences and bisymmetry is ubiquitous in motor systems it is speculated that trainable pattern recognition in low-level perception has evolved as an internalized motor programme.

© 2013 Elsevier Ltd. All rights reserved.

1. Introduction

Many living organisms learn new behaviors from one single exposure to significant sensor inputs. For example, snails learn food aversion from a single experience (Teyke, 1995). This form of learning is called one-shot learning or one-trial learning (Guthrie, 1935) and is common in nature but is hard to explain in neural network models since Hebbian learning requires many repetitions of appropriate stimuli before a new pattern is added to an existing repertoire of recognizable patterns. A mechanism where the brain remembers noteworthy sensory inputs and runs a background loop of simulated experiences would help to provide the repetitive training that is required for Hebbian learning. Such regurgitation could perhaps be performed in sleep when normal sensor inputs are disabled. A driving motivation for the present work is to find viable neural architectures for modeling biological one-shot learning according to this concept.

Support vector machines (SVMs) are pattern recognition algorithms with a firm foundation in optimization and generalization theory and a good track record of practical applications. SVMs are easy to use for non-experts and the performance of a standard SVM often rival that of expertly hand-crafted ANN. An interesting property of SVMs is that the learning state consists of a special set of training examples called support vectors. Significant new training examples may join the set of support vectors. Therefore, it appears that neural implementations of SVMs could be interesting as models of one-shot perceptual learning in nature.

The reasons for considering neural network implementations of SVMs are (1) finding efficient hardware implementations of SVMs, in particular as analogue circuits, and (2) using SVMs as models of biological neural functions. This paper follows the latter approach but we will here briefly review the literature of both branches. The SVM classification function is a weighted sum of kernel function values in which the input vector is one of the arguments to the kernel function. Neural networks can implement any continuous multivariate function with arbitrary accuracy (Cybenko, 1989) so it is not surprising that the SVM classification function readily is expressed as neural networks (Schölkopf & Smola, 2002). As pointed out by Yang, He, and Hu (2012), SVM training is a quadratic programming problem and recurrent neural networks are able to solve such problems. Hence it is feasible to implement both SVM classification and supervised iterative SVM training as artificial neural networks.

Anguita, Ridella, and Rovetta (1998) showed that SVMs can be realized as recurrent electronic circuits. Anguita and Boni (2003) reviewed VLSI implementations of SVMs. A two-layer artificial neural network that implements a 1-norm SVM was defined by Tan, Xia, and Wang (2000) and simplified with respect to the bias calculation by Anguita and Boni (2002). Xia and Wang (2004) demonstrated a one-layer recurrent ANN implementing a 1-norm SVM for which Perfetti and Ricci (2006) as well as Liu and Liu (2009) and Yang et al. (2012) proposed improvements intended to further optimize electronic circuit implementations. For comparison to the present work, it should be noted that these hardware-oriented implementations are in the context of supervised learning

* Tel.: +46 709277264; fax: +46 855503700.
E-mail addresses: magjan@foi.se, magnus@jaendel.se, magjan76@yahoo.se.

0893-6080/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.neunet.2013.09.006

in which an iterative update rule acts on a series of externally provided training examples and is hence not intended for modeling biological one-shot learning.

Support vector machines are used extensively for automating classification in computational biology (for a review see e.g. Noble (2004)). The literature on support vector machines as models of biological phenomena is, however, scant. Galán, Sachse, Galizia, and Herz (2003, 2004) analyzed the olfactory code of the honey bee and noted that the interaction between the antenna lobe and the mushroom body can be regarded as a biological realization of the classification function of a support vector machine. Odours trigger neural attractors in the antenna lobe and the pattern of activated attractors is classified by the mushroom body according to an SVM-like process. Viéville and Crahay (2004) introduced a biologically plausible SVM-like neural network classifier aiming at explaining the fast (100–150 ms) classification process in the visual cortex. The key idea is to classify based on distance to a set of known class prototypes. The prototypes are similar to support vectors and the Hebbian learning mechanism is guided by Vapnik learning theory (Vapnik, 1998). The present paper differs from Viéville and Crahay (2004) in that it explores neural network architectures for standard SVMs and that it focuses on explaining biological one-shot learning rather than the speed of visual perception.

Previous work by the author of this paper shows that a particular SVM algorithm (zero-bias ν-SVM) can be expressed as a biologically plausible ANN that is capable of one-shot learning (Jändel, 2010a). The architecture and dynamics of this model have been compared to the olfactory system (Jändel, 2010a) and also to the burst dynamics of the thalamocortical system (Jändel, 2009). Jändel (2011) point to an evolutionary path along which a neural SVM could emerge from ubiquitous neural components.

Section 2 of the present paper defines an ensemble of SVM algorithms. Section 3 introduces the major neural building blocks, derives training rules and finally analyzes several different neural architectures for each SVM algorithm. Discussion and conclusions are found in Section 4.

2. Support vector machine algorithms

We consider SVMs for binary classification that execute in two different modes: classification and learning. In the classification mode, they receive a test vector $\mathbf{x}$ and output the predicted valence $y \in \{1, -1\}$ of the test vector (bold letters signify vectors). In the learning mode, they use a set of training examples for optimizing internal parameters called weights. Each training example $\{\mathbf{x}, y\}$ consists of a training vector and the associated known valence. All input vectors are considered to be real-valued vectors of equal length. The goal of training is to achieve optimal robust classification performance (see Cristianini and Shawe-Taylor (2000) for details about SVMs).

For future reference, we describe eight well known types of SVMs where each definition consists of the following four parts.

(I) The classification function,

$$f(\mathbf{x}) = \sum_{i=1}^{m} y_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) + b, \tag{1}$$

where K is the positive definite kernel function, α_i are weights and b is a real-valued bias factor. The input vector is classified to be of positive valence if $f(\mathbf{x}) \geq 0$ and to be of negative valence otherwise.

(II) The (dual) objective function $W(\boldsymbol{\alpha})$ where $\boldsymbol{\alpha}$ is the weight vector.

(III) A set of constraints. The positivity condition,

$$\forall i : \alpha_i \geq 0 \tag{2}$$

is satisfied for all SVMs. Biased ($b > 0$) SVMs always include the constraint,

$$\sum_{i=1}^{m} y_i \alpha_i = 0. \tag{3}$$

The constraints (2) and (3) are understood to hold even if they are not repeated in the following SVM definitions.

(VI) An algorithm for computing the value of the bias factor b.

The optimal set of weights is found by maximizing $W(\boldsymbol{\alpha})$ with respect to $\boldsymbol{\alpha}$ under the relevant set of constraints. After computing the bias factor the classification function is used for predicting the valence of test vectors.

Support vectors are training examples with weights $\alpha_i > 0$. The remaining *trivial examples* with $\alpha_i = 0$ do not contribute to the classification function. Support vectors are either *regular* support vectors or *outliers* where the former are support vectors that are correctly classified with sufficient margin. The precise definition of regular support vectors differs between the various SVM types.

For future use we define the classification margin of a training example $(\mathbf{x}_i, y_i)$,

$$M_i = y_i f(\mathbf{x}_i). \tag{4}$$

The margin is positive if the example is correctly classified and negative otherwise. Some of the SVM algorithms specify a target margin M_T for support vectors.

We further define the unbiased classification function,

$$h(\mathbf{x}) = f(\mathbf{x}) - b = \sum_{i=1}^{m} y_i \alpha_i K(\mathbf{x}, \mathbf{x}_i). \tag{5}$$

Introducing Kronecker delta functions,

$$\delta_+(y) = \begin{cases} 1 & \text{if } y = 1 \\ 0 & \text{if } y \neq 1 \end{cases}, \qquad \delta_-(y) = \begin{cases} 1 & \text{if } y = -1 \\ 0 & \text{if } y \neq -1, \end{cases} \tag{6}$$

and the set of regular support vectors SV_R, averages over positive and negative valence regular support vectors are defined according to,

$$\tilde{h}_* = \frac{1}{\tilde{m}_*} \sum_{i \in SV_R} \delta_*(y_i) h(\mathbf{x}_i), \tag{7}$$

where $\tilde{m}_* = \sum_{i \in SV_R} \delta_*(y_i)$ and the symbol $*$ refers to either $+$ or $-$.

The eighth types of SVMs to be defined come in four pairs where each pair includes a zero-bias ($b = 0$) and a biased SVM. The biased SVMs are first described.

2.1. Maximum-margin SVM

The maximum-margin SVM is a hard-margin SVM which means that all support vectors are regular with $M_T = 1$ and the remaining trivial examples have margins $M_T > 1$. Training will succeed only if these conditions are satisfied after optimization of the objective function,

$$W(\boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{m} y_i y_j \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j), \tag{8}$$

under the standard constraints (2) and (3). The bias factor can be computed according to,

$$b = -\frac{1}{2}(\tilde{h}_+ + \tilde{h}_-). \tag{9}$$

2.2. 1-norm SVM

The 1-norm SVM is a soft-margin SVM where support vectors may violate the target margin $M_T = 1$. A slack variable that measures how much the margin is surpassed is defined for each

Download English Version:

https://daneshyari.com/en/article/404094

Download Persian Version:

https://daneshyari.com/article/404094

Daneshyari.com