Neural Network Realization of Support Vector Methods for Pattern Classification

Ying Tan¹, Youshen Xia², Jun Wang²

¹ Department of Electronic Engineering and Information Science
University of Science and Technology of China, Hefei, China
² Department of Mechanical and Automation Engineering
The Chinese University of Hong Kong, Shatin, Hong Kong
Email: {ysxia, jwang}@mae.cuhk.edu.hk

1 Introduction

It is well known that the real-time processing ability of neural networks is one of their most important advantages. It allows neural networks to find numerous applications in many fields. However, the quadratic programming problem for training support vector machines (SVMs) is a computational burden [2,4] even though the training problem is formulated as a convex optimization problem. In particular, for the case where the matrix included in the quadratic program is semidefinite and training dataset is very large, the training of SVM is very time-consuming [5,6]. In this frequently happened case, more efficient algorithm has to be developed to speed up the training and evaluation of SVMs. In this respect, neural networks offer a very promising tool. If we combine the merits of the SVMs and neural networks, we can obtain a new neural network SVM with a better performance. Motivated by this idea, we apply a recurrent neural network to SVM training for pattern recognition. Specifically, a primal-dual neural network is exploited to solve the quadratic programming problem encountered in training SVMs. The properties of the network allow us to design SVMs without adjustable network parameters and give a better solution for ill-posed problems, e.g., semi-definite quadratic programming problem.

2 Support Vector Machines

Let \{ (xᵢ, yᵢ), i = 1, \cdots , N \} be a set of training examples, the i-th example \( xᵢ \in R^n \) (n is the dimension of the input space) belongs to a class labeled by \( yᵢ \in \{-1, 1\} \). The aim of the SVM approach is to define a hyperplane in a high dimensional feature space \( Z \), which divides the set of examples in the feature space such that all the points with the same label are on the same side of the hyperplane \[1,2\]. So, a SVM first maps \( x \) from the input space \( R^n \) to \( z = \Phi(x) \) in an \( m \)-dimensional feature space \( Z \). Then we find \( w \in R^m \) and \( b \) so as to construct a hyperplane \( w^T z + b \) in \( Z \). Finally, a decision function of the classifier for this "optimal" hyperplane can be given by

\[
f_{w,b} = sgn([w^T z] + b),
\]

for which the separation between the positive and negative examples is maximized. Without loss of generality, we consider the case when the training set is not separable in \( Z \). Since \( \Phi(x) \) is unknown,
some slack variable $\xi_i$ are first introduced [2] such that
\[
\begin{align*}
&y_i[w^T \Phi(x_i) + b] \geq 1 - \xi_i, \quad i = 1, \ldots, N \\
&\xi_i \geq 0, \quad i = 1, \ldots, N.
\end{align*}
\tag{2}
\]

The resultant problem becomes
\[
\min_{\mathbf{w}, b, \xi_i} R(\mathbf{w}, \xi_i) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + c \sum_{i=1}^{N} \xi_i
\tag{3}
\]
subject to Eq. (2), where $c > 0$ is a user-defined regularization parameter control the tradeoff between model complexity and training error, and $\xi_i$ measures the (absolute) difference between $\mathbf{w}^T \mathbf{z} + b$ and $y_i$. According to the Kuhn-Tucker theorem [11], the solution of this problem can be shown to have an expansion
\[
\mathbf{w} = \sum_{i=1}^{N} y_i \alpha_i \mathbf{z}_i
\tag{4}
\]
where the corresponding training examples $(\mathbf{z}_i, y_i)$ with nonzero coefficients $\alpha_i$ are called support vectors which meets the constraint Eq. (2). The coefficients $\alpha_i$ can be found by solving the following convex quadratic programming problem
\[
\max_{\alpha_i} -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j (\mathbf{z}_i^T \mathbf{z}_j) \alpha_i \alpha_j + \sum_{i=1}^{N} \alpha_i
\tag{5}
\]
subject to $\sum_{i=1}^{N} \alpha_i y_i = 0$ and $0 \leq \alpha_i \leq c, \quad i = 1, \ldots, N$.

The Mercer’s Theorem [1] tells us that there exists a continuous kernel $K(\cdot, \cdot)$ such that $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j)$. By substituting Eq. (4) into Eq. (1), the decision function of the classifier can be expressed as
\[
f(\mathbf{x}) = sgn[\sum_{i=1}^{N} \alpha_i y_i \cdot K(\mathbf{x}_i, \mathbf{x}) + b].
\tag{6}
\]

3 Neural Realization of SVMs

For convenient computation here, let $a_i = \alpha_i y_i$. Then Eqs. (5) is equivalent to
\[
\min_{\mathbf{a}} \frac{1}{2} \mathbf{a}^T \mathbf{Q} \mathbf{a} - \mathbf{y}^T \mathbf{a}
\tag{7}
\]
subject to $\mathbf{e}^T \mathbf{a} = 0$ and $1 \leq \mathbf{a} \leq \mathbf{h}$, where $\mathbf{a} = (a_1, a_2, \ldots, a_l)^T$, $\mathbf{y} = (y_1, y_2, \ldots, y_l)^T$, the $i$-th entry of matrix $Q_{ij} = K(x_i, x_j)$, $\mathbf{e} = (1, \ldots, 1)^T \in \mathbb{R}^l$, $\mathbf{l} = (l_1, l_2, \ldots, l_l)^T$, $\mathbf{h} = (h_1, h_2, \ldots, h_l)^T$, for $i = 1, \ldots, l$,
\[
l_i = \begin{cases} -c & y_i = -1 \\ 0 & y_i = 1 \end{cases}, \quad h_i = \begin{cases} c & y_i = 1 \\ 0 & y_i = -1 \end{cases}.
\]
According to dual theory [11], the dual problem of (5) is

\[
\begin{align*}
\text{Maximize} & \quad -\frac{1}{2}a^T Qa + v^T v - h^T u \\
\text{subject to} & \quad Qa - es - y + v - u = 0, \\
& \quad v \geq 0, u \geq 0
\end{align*}
\]

(8)

where \(v, u \in \mathbb{R}^l\) and \(s \in \mathbb{R}\) are the vectors of dual decision variables. By the complementary slackness theorem [11], \(a^*\) and \((s^*, u^*, v^*)\) are optimal solutions respectively to primal problem (5) and the dual problem (9) if and only if \(a^*\) and \(s^*\) satisfy \(e^T a^* = 0, 1 \leq a \leq h\), and the following condition

\[
a^* = P_\Omega (a^* - Qa^* - es^* + y)
\]

where \(\Omega = \{a \in \mathbb{R}^l| \, 1 \leq a \leq h\}\), \(P_\Omega : \mathbb{R}^l \rightarrow \Omega \subset \mathbb{R}^l\) is a projection operator which is defined by

\[
P_\Omega (a_i) = \begin{cases} 
 l_i & a_i < l_i \\
 a_i & l_i \leq a_i \leq h_i \\
 h_i & a_i > h_i.
\end{cases}
\]

By the projection Theorem [11] we see that the problem (5) and its dual (8) are equivalent to the following linear variational inequality problem: find \(z^* \in X\) such that

\[
(z - z^*)^T F(z^*) \geq 0, \quad \forall z \in X
\]

(9)

where \(z = (a, s)^T, X = \{z \in \mathbb{R}^{l+1}| \, a \in \Omega\}, F(z) = Mz + q\), and

\[
M = \begin{bmatrix} Q & e \\ -e^T & 0 \end{bmatrix}, \quad q = \begin{bmatrix} -y \\ 0 \end{bmatrix}.
\]

In [10], we proposed a recurrent neural network for solving (9):

\[
\frac{dz}{dt} = (I + M^T)(P_\Omega [z - F(z)] - z)
\]

which generalizes the existing recurrent neural network [9] for solving the standard linear and quadratic programming problem. Note that

\[
P_X[z - F(z)] - z = \begin{bmatrix} P_X[a - (Qa + es - y)] - a \\ -e^T a \end{bmatrix},
\]

then the above neural network can be written as

\[
\frac{d}{dt} \begin{bmatrix} a \\ s \end{bmatrix} = \begin{bmatrix} (I + Q)(P_\Omega [a - (Qa + es - y)] - a) - ee^T a \\ -e^T P_\Omega [a - (Qa + es - y)] \end{bmatrix}
\]

(10)

This network consists of a two-layer structure and has \(l + 1\) neurons. The projection operator \(P_\Omega(\cdot)\) may be implemented by using a piecewise activation function. The circuit realizing the recurrent neural network can be easily implemented by using simple hardware without analog variable multipliers or penalty parameter. More important, the present size of the neural network is almost same
as that of (5) due to only a equality constraints. For the stability of the neural network we have the following result.

**Theorem 1.** Assume that the matrix $Q$ is positive semidefinite. Then the neural network of (10) is stable in the sense of Lyapunov and globally converges to a point $(\mathbf{a^*}, s^*)$ where $\mathbf{a^*}$ is an optimal solution to (5). Specially, the neural network of (10) is globally asymptotically stable when (5) and its dual has only one optimal solution.

In order to compute the threshold $b$ in decision function, we use the formulation below

$$b = \frac{1}{l_{SV}} \left( \sum_{j=1}^{l_{SV}} y_j - \sum_{j=1}^{l_{SV}} \sum_{i=1}^{l_{SV}} a_i K(\mathbf{x}_j, \mathbf{x}_j) \right) / l_{SV},$$

(11)

where $l_{SV}$ denotes the number of support vectors.

Now, we combine the kernel function computation and the neural network for finding support vector as well as the decision function computation, and the neural realization of SVMs for classification can be easily realized by a block diagram shown in Fig 1. The following convergence result is a direct result of Theorem 1.

**Corollary 1.** Assume that the Mercer’s condition [1] is satisfied. Then a decision function is obtained by the output of the neural network shown in Fig. 1.

Figure 1: A block diagram of the neural realization of SVMs for classification.

### 4 Experimental Results

**A. The Iris classification problem**

Let’s use a neural network to train a SVM with a Gaussian RBF kernel with the width of 0.1 for the iris benchmark problem. The data of the iris problem are characterized with four attributes (i.e., the petal length and width, setal length and width). The goal is to classify the class of iris based on these based on these four attributes. The dataset consist of 150 samples belonging to three classes (i.e., virginica, versicolor, setosa), each class has 50 samples. We choose 120 samples for the training set and the remaining is used as testing set. For illustration of our proposed neural network, we train the SVM to separate the class I from class II and separate the class II from class III with the penalty factor $c = 1$. We choose a Gaussian RBF function as the kernel of the SVM. Fig. 3 shows the optimal decision boundary of the SVM to separate the first class from second class. Fig. 4 gives the decision boundary of the SVM to separate the second class from the third class.
B. The Adult Benchmark

This experiment is taken on the set of training data of the UCI “adult” benchmark. The benchmark task is to predict whether or not a household has an income greater than 50,000 dollars based on 14 other fields in a census form. Eight of those fields are categorical, while six are continuous. The six fields are quantized into quintiles, which yields a total of 123 sparse binary features. There are 9 training files in the original data of the adult benchmark. For illustrate our network method, we adopted the first part of the training set to train a SVM. There are 800 samples in the training set and 2000 samples in the testing set in our experiment. We trained the SVM of Gaussian RBF kernel with width of 10 and $c = 1$, and we trained also a SVM of cubic polynomial kernel with degree= 3, whose results are listed in Table 1. Denote the energy error of the proposed neural network by $r(z) = z - P_{\Omega}(z - F(z))$, Fig. 4 shows the transient behavior of the energy error of the proposed neural network in the RBF kernel case.

References


Figure 4: The transient behavior of the energy error of the proposed neural network with the initial point $z_0 = 0 \in \mathbb{R}^{801}$.


