Applied Mathematics and Computation 217 (2011) 5328–5337

Contents lists available at ScienceDirect

Applied Mathematics and Computation

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

Parallel algorithm for training multiclass proximal Support Vector Machines ☆

Lingfeng Niu

Research Center on Fictitious Economy and Data Science, Chinese Academy of Sciences, Beijing 100190, China

ARTICLE INFO

Keywords:
Parallel algorithm
Multiclass classification
Proximal Support Vector Machine
Post-processing
Low rank matrix approximation

ABSTRACT

In this paper we describe a proximal Support Vector Machine algorithm for multiclassification problem by one-vs-all scheme. The computational requirement for the new algorithm is almost the same as training one of its element binary proximal Support Vector Machines. Low rank approximation is taken to reduce computational costs when the kernel matrix is too large. An error bound estimation for the approximated solution is given, which is used as a stopping criteria for low rank approximation. A post-processing strategy is developed to overcome the difficulty arising from unbalanced data and to improve the classification accuracy. A parallel implementation of the algorithm using standard MPI communication routines is provided to handle large-scale problems and to accelerate the training process. Experiment results on several public datasets validate the effectiveness of our proposed algorithm.

1. Introduction

Support Vector Machine (SVM) [1–4] has become one of the most prominent machine learning techniques in recent years. The basic idea of SVM is to find a hyperplane in feature space, which can separate samples into two classes. Following this approach, several other classifiers have be invented [5–8]. One of them is proximal SVM, which was implemented by assigning points in each class to the closest of two parallel planes that are pushed apart as far as possible. This classifier has been independently proposed several times in history. The name proximal SVM is from Fung and Mangasarian [6,9]. Suykens and Vandewalle also derived it as a modification of the standard SVM, and named it as least square SVM [5,10]. This formulation also can be interpreted as an implementation of Tikhonov regularization with the classic square loss [11,12]. From this point of view, Rifkin [13] renamed it as regularized least square classification.

It has been shown that the same generalization bounds that apply to SVMs apply to proximal SVMs as well [13]. Extensive experimental results on both toy and real-world examples also demonstrate empirically that the performance of proximal SVMs is essentially equivalent to SVMs across a wide range of problems [5,6,10,13,9]. Proximal SVMs have become a highly variable alternative to SVMs. The choice between conventional SVM and proximal SVM is based on computational tractability considerations [13]. For the binary classification problems which essentially can be separated in the input space, linear proximal SVM can be trained very fast and has been considered as one of the most efficient classifiers [6].

The mathematical formulation of the proximal SVM model is quite simple. It is an equality constrained quadratic programming, whose solution can be obtained by solving a single system of linear equations. This is in direct contrast to the standard SVM which requires the solution of an inequality constrained optimization problem [13]. However, this does

☆ This work was initiated in 2007 when the author was a Ph.D. candidate at State Key Laboratory of Scientific and Engineering Computing, Institute of Computational Mathematics and Scientific/Engineering Computing, AMSS, CAS.

E-mail address: niulf@lsec.cc.ac.cn

0096-3003/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.amc.2010.11.056

not mean proximal SVMs are always easier to be solved than SVMs. The main difficulty lies in the dense kernel matrix. When massive datasets are used, the dense and large-scale kernel matrices can not fit the memory restrictions. Based on the sparsity structure of SVM solution, decomposition approaches [14–18] are derived to handle this memory problem. However, for proximal SVMs, in general all the component of solution is nonzero, which restricts the use of decomposition method. Fortunately kernel matrices have the property of low rank for lots of problems [19,20]. Therefore, we utilize the low rank approximation method in [21] to avoid explicit storage of the entire kernel matrix when the dimension of the kernel matrix is very large.

Proximal SVM is inherently a binary classifier: it classifies a sample as being positive or negative. In contrast, many problems we are interested in are multiclass classification. Namely, there are more than two classes, and our job is to pick a single class to which a data point belongs. Considerable efforts have been devoted to develop efficient training algorithms for multiclassification problems [4,22–26]. Usually the computational requirement for training multiclassifiers is much higher than the same scale binary classification problems. Nowadays, a popular approach for multiclassification is decomposing a multiclass problem into multiple independent binary classification tasks, and applying some scheme such as one-vs-all, one-vs-one, directed acyclic graph or the error-correcting codes to build a multiclassifier on results of these binary predictors. Over all those approaches, one-vs-all is a competitive candidate in practice [26]. In this paper, based on the special structure of proximal SVMs, we propose an algorithm to reduce the computational requirement for training a multiclass proximal SVM by one-vs-all scheme to almost the same as a single underlying binary problem.

The rest of this paper is organized as follows. In the next section we present the algorithm details for solving linear kernel proximal SVMs. A theorem estimating the error bound of approximated solutions and the post-processing technique to improve the performance are also given. Section 3 proposes a new proximal SVM model for the nonlinear kernel and extends the new algorithm to that case. Parallel implementation and experiments, which demonstrate the utility of the suggested method, are presented in Section 4. We wrap things in Section 5 with some concluding remarks and leading directions for further study.

2. The new training algorithm for linear multiclass proximal SVMs

2.1. Preliminaries

Suppose we have m samples $\{\mathbf{x}_i,\ i=1,\dots,m\}\subseteq\mathbb{R}^n$. For each data point $\mathbf{x}_i$, label $y_i\in\{-1,1\}$ indicates which class it belongs to. Proximal SVMs for binary classification with linear kernel [6] can be formulated as the following quadratic programming (QP):

$$\min_{w,\gamma,\xi}\ \frac{v}{2}\|\xi\|^2+\frac{1}{2}(w^Tw+\gamma^2), \tag{1a}$$

$$\text{s.t. } D(Aw-e\gamma)+\xi=e, \tag{1b}$$

where $\|\cdot\|$ is 2-norm, $A=(\mathbf{x}_1,\dots,\mathbf{x}_m)^T\in R^{m\times n}$, $D=diag(y_1,\dots,y_m)$ and e denotes the vector with all the entries 1. For the equality constraints in (1), we denote λ_i as the dual variable of constraint for sample $\mathbf{x}_i$. Then the KKT conditions for QP (1) are

$$w-A^TD\lambda=0, \tag{2a}$$

$$\gamma+e^TD\lambda=0, \tag{2b}$$

$$v\xi-\lambda=0, \tag{2c}$$

$$D(Aw-e\gamma)+\xi-e=0. \tag{2d}$$

Eliminating the primal variable (w,γ,ξ) by the Eqs. (2a)–(2c), we get the equations for λ from (2d) as follows:

$$\left(\frac{I}{v}+D(AA^T+ee^T)D\right)\lambda=e. \tag{3}$$

Once λ is computed, w and γ can be obtained by (2a) and (2b), respectively.

Now we consider multiclassification problems. There are still m data points $\{\mathbf{x}_i,\ i=1,\dots,m\}\subseteq\mathbb{R}^n$, but this time label y_i has k different choices $\{1,2,\dots,k\}$. Using one-vs-all scheme, we need k different binary classifiers, each one is trained to distinguish the samples in one class from the samples in the other classes. When it is desired to classify a new sample, the k classifiers are run, and the classifier which outputs the largest (most positive) value is chosen. For each $s\in\{1,2,\dots,k\}$, to separate class s from the rest, we define

$$D_{i,i}=\begin{cases}1, & \text{if } y_i=s,\\ -1, & \text{if } y_i\neq s.\end{cases}$$

Once the k minimization problems of th e form (1) are solved (with different D defined as above), k unique separating planes are generated:

$$\mathbf{x}^Tw_i-\gamma_i=0,\quad i=1,\dots,k. \tag{4}$$