Neurocomputing 74 (2011) 2532–2540

Contents lists available at ScienceDirect

Neurocomputing

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

A parallel incremental extreme SVM classifier

Qing He [a,*], Changying Du [a,b], Qun Wang [a,b], Fuzhen Zhuang [a,b], Zhongzhi Shi [a]

[a] Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China
[b] Graduate School of the Chinese Academy of Sciences, Beijing, China

ARTICLE INFO

Available online 14 May 2011

Keywords:
Parallel extreme SVM (PESVM)
MapReduce
Incremental extreme SVM (IESVM)
Parallel incremental extreme SVM (PIESVM)

ABSTRACT

The classification algorithm extreme SVM (ESVM) proposed recently has been proved to provide very good generalization performance in relatively short time, however, it is inappropriate to deal with large-scale data set due to the highly intensive computation. Thus we propose to implement an efficient parallel ESVM (PESVM) based on the current and powerful parallel programming framework MapReduce. Furthermore, we investigate that for some new coming training data, it is brutal for ESVM to always retrain a new model on all training data (including old and new coming data). Along this line, we develop an incremental learning algorithm for ESVM (IESVM), which can meet the requirement of online learning to update the existing model. Following that we also provide the parallel version of IESVM (PIESVM), which can solve both the large-scale problem and the online problem at the same time. The experimental results show that the proposed parallel algorithms not only can tackle large-scale data set, but also scale well in terms of the evaluation metrics of speedup, sizeup and scaleup. It is also worth to mention that PESVM, IESVM and PIESVM are much more efficient than ESVM, while the same solutions as ESVM are exactly obtained.

1. Introduction

Unlike those conventional iterative implementations, Huang et al. [1,2] proposed a new learning algorithm called extreme learning machine (ELM) for single-hidden layer feedforward neural networks (SLFNs), which randomly chooses input weights and hidden biases and analytically determines the output weights of SLFNs. The learning process of ELM for an SLFN includes two steps. First, the input vectors are mapped into the hidden layer output vectors through the hidden layer of a SLFN, with its input weights and hidden biases randomly generated. Second, a minimum norm least squares solution of the output weights is obtained by computing the generalized inverse of the hidden layer output matrix [1,2]. Though it has been studied that ELM can provide good generalization performance at an extremely fast learning speed [1,2], ELM still tends to be over-fitting at the second step according to the empirical risk minimization (ERM) principle [33–35].

To overcome the over-fitting problem in ELM, according to Vapnik's structure risk minimization (SRM) principle [34,35], Liu et al. [3] formulated a new nonlinear support vector machine (SVM), called extreme support vector machine (ESVM).[1] Later, Frénay and Verleysen [4] also performed a similar work on standard SVM. In ESVM, similar as the first step of ELM's learning process, a nonlinear map function is explicitly constructed by a random SLFN's hidden layer. Liu et al. [3] made a significant contribution. They show that the ELM learning approach can be applied to SVMs directly by simply replacing SVM kernels with random ELM kernels and better generalization can be achieved [5,3]. The ESVM classifier can also be regarded as a special form of regularization networks [6], which classifies the data points similar as proximal SVM (PSVM) [7], multisurface PSVM [8], and least squares SVM (LSSVM) [9]. As has been shown in [3], ESVM can produce better generalization performance than ELM almost all of the time and can run much faster than other nonlinear SVM algorithms with comparable accuracy.

It is observed that ESVM must do the multiplications of several matrices to obtain the solution, and the computation complexity depends on the size of the training data set. This leads to the inefficiency of ESVM when dealing with large-scale data. Furthermore, ESVM in its current form can not be used in online learning. Actually, many real life machine learning problems can be more naturally viewed as online rather than batch learning problems, the data is often collected continuously in time.

With the fast development of cloud computing, many researchers have proposed some parallel learning algorithms [10–13], such as the parallel implementation of ELM [10]. And

* Corresponding author at: Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China. Tel.: +86 62600542.
E-mail addresses: heq@ics.ict.ac.cn (Q. He), 530751551@qq.com, ducy@ics.ict.ac.cn (C. Du).

[1] ESVM is essentially a linear PSVM [7] in the ELM feature space, which corresponds to a ELM with output weights trained by ridge regression (i.e. L2/Tikhonov-regularization on the output weights). So ESVM is not the same as standard SVM with ELM kernel [4].

0925-2312/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2010.11.036

lots of efforts have been devoted to the development of online/incremental learning[2] in recent years [14–23].

However, now more and more data processing problems are both large-scale and online, exploring effective and efficient algorithms that can solve both the large-scale problem and the online problem at the same time is of great significance. Although there have been many parallel and online learning algorithms, but most of them only focus on one problem, either large scale or online.

In this paper, we extend ESVM to an incremental learning algorithm IESVM, and develop the parallel implementations of ESVM and IESVM.

We note that Google has provided a PSVM: parallelizing support vector machine, but it does not support incremental learning [11]. And though several incremental SVM algorithms have been proposed [21–23], they are all serial algorithms and can not be parallelized easily, while our proposed IESVM can be parallel executed very easily.

We implement our PESVM based on MapReduce, which is a current and powerful parallel programming framework. The experiments show that PESVM scales well in terms of the evaluation metrics of speedup, sizeup and scaleup. By comparing ESVM with PESVM, IESVM and PIESVM, we observe that PESVM, IESVM and PIESVM can give exactly the same solutions as ESVM while saving much training time, which is shown in our experiments. The experiments also show that PIESVM has very fast incremental learning speed, which can be used to solve large-scale online learning problems efficiently.

The rest of the paper is organized as follows. We first give some preliminary knowledge in Section 2. Then Section 3 presents our parallel ESVM algorithm based on MapReduce. Section 4 gives our incremental ESVM classifier and its parallel implementation PIESVM. Then experimental results are shown in Section 5. Finally, we draw our conclusions in Section 6.

A word about our notations. All vectors will be column vectors unless transposed by a superscript$'$. The scalar product of two vectors x and y in n-dimensional space R^n will be denoted by $x'y$, and the 2-norm of a vector x is denoted by $\|x\|$. For a matrix $A \in R^{m\times n}$, A_i is the ith row of A which is a row vector R^n, while $A_{.j}$ is the jth column of A. A column vector of ones of arbitrary dimension will be denoted by e. The identity matrix of arbitrary dimension will be denoted by I.

2. Preliminary knowledge

2.1. Review of extreme SVM

To derive our parallel incremental ESVM classifier, we first give a brief description of the ESVM formulation [3].

Consider the 2-class classification problem of classifying m points in n-dimensional real space R^n, represented by the $m\times n$ matrix A. A $m\times m$ diagonal matrix D with $+1$ or -1 along its diagonal specifies the membership of class $A+$ or class $A-$ of each point A_i. For this problem, the extreme support vector machine with a linear kernel, which has the same form as the linear PSVM [7], is given by the following quadratic program with parameter $\nu>0$ and linear equality constraint (y is the slack variable):

$$\min_{(w,r,y)\in R^{n+1+m}} \quad \frac{\nu}{2}\|y\|^2+\frac{1}{2}\left\|\begin{bmatrix} w\\ r\end{bmatrix}\right\|^2 \\ \text{s.t.} \quad D(Aw-er)+y=e \tag{1}$$

which tries to find the proximal planes: $x'w-r=\pm 1$, where w,r are the orientation and the relative location to the origin respectively. These two planes are proximal to the points in class $A+$ and class $A-$ respectively, and the plane $x'w-r=0$, which is midway between the above two proximal planes, is chosen as the separating plane which acts as below:

$$x'w-r\begin{cases} >0, & \text{then} \quad x\in A+\\ <0, & \text{then} \quad x\in A-\\ =0, & \text{then} \quad x\in A+ \text{ or } x\in A-\end{cases} \tag{2}$$

The nonlinear ESVM classifier [3] is obtained by applying the above linear formulation in a feature space, which is introduced by a specially devised nonlinear mapping function. The mapping function $\Phi(\cdot): R^n\to R^{\tilde{n}}$, which maps the input vectors into the vectors in a feature space, can be constructed as follows:

$$\Phi(x)=G(Wx^1) \\ =\left(g\left(\sum_{j=1}^{n}W_{1j}x_j+W_{1(n+1)}\right),\ldots,g\left(\sum_{j=1}^{n}W_{\tilde{n}j}x_j+W_{\tilde{n}(n+1)}\right)\right)' \tag{3}$$

where $x\in R^n$ is the input vector, $x^1=[x'\ 1]'$, $W\in R^{\tilde{n}\times(n+1)}$ is a matrix whose elements are randomly generated, $\Phi(x)$ is the vector corresponding to x in the feature space, and the notation $G(\cdot)$ represents a map which takes a matrix Z with elements z_{ij} and returns another matrix of the same size with elements $g(z_{ij})$, where g is an activation function (typically the sigmoidal function). Note that x, W can be interpreted as the input vector and the input weights and hidden biases of an SLFN respectively, and $\Phi(x)$ is the hidden layer's output vector of x in ELM algorithm.

For the $m\times n$ matrix A, $\Phi(A)$ is defined as $\Phi(A)=[\Phi(A_1'),\ldots,\Phi(A_m')]'$.

Then, the nonlinear ESVM can be formulated as the following quadratic program problem with a parameter $\nu>0$:

$$\min_{(w,r,y)\in R^{\tilde{n}+1+m}} \quad \frac{\nu}{2}\|y\|^2+\frac{1}{2}\left\|\begin{bmatrix} w\\ r\end{bmatrix}\right\|^2 \\ \text{s.t.} \quad D(\Phi(A)w-er)+y=e \tag{4}$$

From the linear constraint of (4) we can get an explicit expression of y. Substituting y by its explicit expression in the objective function of (4), we can get the following unconstrained minimization problem:

$$\min_{(w,r)\in R^{\tilde{n}+1}} \frac{\nu}{2}\|D(\Phi(A)w-er)-e\|^2+\frac{1}{2}\left\|\begin{bmatrix} w\\ r\end{bmatrix}\right\|^2 \tag{5}$$

Setting the gradient with respect to w and r to zero and noting that $D^2=I$ gives the following necessary and sufficient optimality conditions for (5):

$$\nu\Phi(A)'(\Phi(A)w-er-De)+w=0 \\ \nu e'(-\Phi(A)w+er+De)+r=0 \tag{6}$$

By solving the linear system of Eqs. (6) we can obtain the following simple expression for w and r in terms of problem data:

$$\begin{bmatrix} w\\ r\end{bmatrix}=\left(\frac{I}{\nu}+E_\Phi'E_\Phi\right)^{-1}E_\Phi'De \tag{7}$$

where $E_\Phi=[\Phi(A)\ \ -e]\in R^{m\times(\tilde{n}+1)}$.

Now for an unseen point x, the nonlinear classifier works as follows:

$$\Phi(x)'w-r\begin{cases} >0, & \text{then} \quad x\in A+\\ <0, & \text{then} \quad x\in A-\\ =0, & \text{then} \quad x\in A+ \text{ or } x\in A-\end{cases} \tag{8}$$

Since linear ESVM has a very similar form as nonlinear ESVM, the only difference between them is that nonlinear ESVM is modeled in a feature space introduced by an explicit mapping

[2] In this paper, we use the terms "incremental learning" and "online learning" interchangeably.