Journal of Systems Engineering and Electronics, Vol. 17, No. 4, 2006, pp. 910~915

Solving large-scale multiclass learning problems via an efficient support vector classifier

Zheng Shuibo[1], *Tang Houjun*[1], *Han Zhengzhi*[1] *& Zhang Haoran*[2]

1. School of Electrical and Information Engineering, Shanghai Jiaotong Univ., Shanghai 200030, P. R. China;
2. Dept. of Electronic Engineering, Zhejiang Normal Univ., Jinhua 321004, P. R. China

(Received May 23, 2005)

Abstract: Support vector machines (SVMs) are initially designed for binary classification. How to effectively extend them for multiclass classification is still an ongoing research topic. A multiclass classifier is constructed by combining SVM^{light} algorithm with directed acyclic graph SVM (DAGSVM) method, named DAGSVM^{light}. A new method is proposed to select the working set which is identical to the working set selected by SVM^{light} approach. Experimental results indicate DAGSVM^{light} is competitive with DAGSMO. It is more suitable for practice use. It may be an especially useful tool for large-scale multiclass classification problems and lead to more widespread use of SVMs in the engineering community due to its good performance.

Keywords: support vector machines (SVMs), multiclass classification, decomposition method, SVM^{light}, sequential minimal optimization (SMO).

1. INTRODUCTION

In recent years, a new type of classifier, support vector machines[1~2], is receiving adoption increasingly as a state-of-the-art tool to solve knowledge discovery problems. SVMs are based on the statistical learning theory of Vapnik[1] and quadratic programming optimization.

Support vector machines (SVMs) are initially designed for binary classification problem. How to effectively extend them for multiclass classification is still an ongoing research topic. Currently there are two types of approaches for multiclass SVMs classification. One type is by directly considering all data in one optimization formulation while another is by constructing and combining several binary classifiers. There are three methods applied for SVMs multiclass classification problem: one-against-all, one-againstone-one and DAGSVM. DAG method is more suitable for practical use than the other methods[3~5].

Joachims' algorithm SVM^{light} (version 5.0)[6~7] is an efficient and effective decomposition method for large-scale SVMs learning tasks. But SVM^{light} is specially designed for binary classification. This work effectively extended SVM^{light} to multiclass classification with DAGSVM[4] method. The multiclass SVM^{light} package is named DAGSVM^{light}. Selection of the working set is the most crucial issue in decomposition algorithms. This paper proposes a new method to select the working set which is identical to the working set selected by SVM^{light} approach. In machine learning community, the problem of handwritten digit recognition is a good example to test the multiclass classification performance. Experimental results on a typical large-scale multiclass benchmark set of handwritten digit and Protein data set are discussed to evaluate the DAGSVM^{light}.

2. FORMULATION OF DAGSVM^{light} ALGORITHM

A multiclass classifier, named DAGSVM^{light}, is constructed by combining SVM^{light} algorithm with DAGSVM method.

2.1 SVM^{light} Approach

Decomposition method is the first practical method for solving large-scale SVMs training problem. It was initially proposed in Ref.[11]. The key idea of decomposition is to fix all but a small number of free optimization variables, and to solve a sequence of constant size problem. The set of variables optimized at a current iteration is denoted as the working set. Because the working set is re-optimized, the value of the objective function is improved at each iteration, provided that the working set is not optimal before re-optimization. Iteration is stopped when termination criteria, derived from Karush-Kuhn-Tucker (KKT) conditions, are satisfied to a required precision.

The QP problem in order to train SVMs is shown below

$$\begin{aligned} &\min W(\alpha)=\frac{1}{2}\alpha^{\mathrm{T}}Q\alpha-\alpha^{\mathrm{T}}1 \\ &\text{s. t. } \alpha^{\mathrm{T}}y=0 \\ &0\le\alpha\le C1 \end{aligned} \tag{1}$$

where $(1)_i=1$, $(Q)_{ij}=y_iy_jK(x_i, x_j)$.

At each step index set is partitioned in two sets B of free variables and N of fixed variables. The set B is referred to as the working set. Suppose that α, y, and Q are decomposed with respect to B and N

$$\alpha=\begin{bmatrix}\alpha_B\\ \alpha_N\end{bmatrix}, y=\begin{bmatrix}y_B\\ y_N\end{bmatrix}, Q=\begin{bmatrix}Q_{BB} & Q_{BN}\\ Q_{NB} & Q_{NN}\end{bmatrix}$$

Optimization of the working set is also a quadratic program. Rearrange the terms of the objective function, the equality constraint and the box constraint in E_q. (1), and omit the constant terms independent of αB from the objective. The resulting quadratic program becomes the following QP subproblem

$$\begin{aligned}\min\ & W(\alpha_B)=\frac{1}{2}\alpha_B^{\mathrm{T}}Q_{BB}\alpha_B+\alpha_B^{\mathrm{T}}(Q_{BN}\alpha_N-1)\\ \text{s. t. }& \alpha_B^{\mathrm{T}}y_B=-\alpha_N^{\mathrm{T}}y_N\\ & \alpha_B-Cl\le 0\\ & \alpha_B\ge 0\end{aligned}\tag{2}$$

Subproblem (2) is a positive semidefinite quadratic program problem which is small enough to be solved efficiently by off-the-shelf QP packages, such as MINOS and LOQO.

In order to achieve a rapid decrease in the objective function, it is required a good working set such that the current iteration will make much progress towards the target function. The strategy based on Zoutendijk's method is to find a steepest feasible direction d of decent which has only q non-zero elements. The following optimization problem

$$\begin{aligned}&\min\ g(\alpha)^{\mathrm{T}}d\\ &\text{s. t. } y^{\mathrm{T}}d=0\\ &d_i\ge 0\quad \text{for } i \text{ such that } \alpha_i=0\\ &d_i\le 0\quad \text{for } i \text{ such that } \alpha_i=C\\ &-1\le d_i\le 1\\ &\#\{d_i\mid d_i\ne 0\}=q\end{aligned}\tag{3}$$

where $g(\alpha)$ is the vector of partial derivatives at α.

For QP (1):

$$g(\alpha)=Q\alpha-1\tag{4}$$

The working set composition can be stated as follows: Sort data elements by g_iy_i in decreasing order and pick the $q/2$ elements from the top of the list such that $0<\alpha_i<C$, or $d_i=-y_i$ satisfies (3). Select $q/2$ elements from the bottom of the list such that $0<\alpha_i<C$, or $d_i=y_i$ satisfies (3).

Many efficient tricks and strategies are utilized in the algorithm of Joachims' SVM^{light} to speed up training and make the computational cost inexpensive. Specially, a shrinking strategy to reduce the size of the problem and a caching strategy to allow a trade-off between memory consumption and training time are implemented. If q is less than the size of working size, this prevents zig-zagging.

2.2 Multiclass Classification Methods

The earliest implementation for SVMs multiclass classification is the one-against-all method[12]. This method constructs N SVMs models where N is the number of classes. The ith SVMs are trained with all of the examples in the ith class with positive labels, and all other examples with negative labels. Thus there are N decision functions. The test example is in the class which has the biggest value of the decision function.

Another major method is called the one-against-one method[13]. This method constructs $N*(N-1)/2$ classifiers where each one is trained on data from two classes. The test example is in the class with the largest vote. The voting method is also called the "Max Wins" strategy.

The third algorithm, DAGSVM, is an effective method to solve multiclass problem. For N class problem, its training phase is the same as one-against-one method by solving $N*(N-1)/2$ binary SVMs. However, during the testing phase, it exploits a rooted binary directed acyclic graph which has $N*(N-1)/2$ internal nodes and N terminal nodes (leaves). Each internal node is a binary SVMs of ith and jth classes. Before reaching a leaf node a path indicates the predicted class. Its generalization error depends on the number N of class and class margin in the internal nodes. In addition, its test time is less than the one-against-one method. DAGSVM classification schematic graph of four classes is demonstrated in Fig.1.

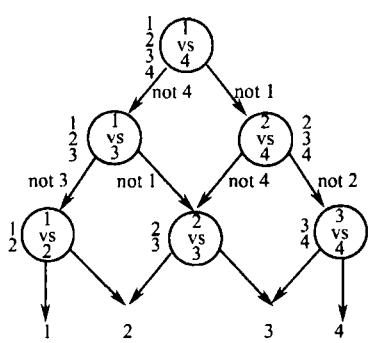

Fig.1 DAGSVM of four classes