Expert Systems with Applications 37 (2010) 6618–6628

Contents lists available at ScienceDirect

Expert Systems with Applications

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

An ACO-based algorithm for parameter optimization of support vector machines

XiaoLi Zhang *, XueFeng Chen, ZhengJia He

State Key Lab. for Manufacturing Systems Engineering, Xi'an Jiaotong University, Xi'an 710049, PR China

ARTICLE INFO

Keywords:
Ant colony optimization (ACO) algorithm
Support vector machines (SVM)
Parameter optimization
ACO-SVM model

ABSTRACT

One of the significant research problems in support vector machines (SVM) is the selection of optimal parameters that can establish an efficient SVM so as to attain desired output with an acceptable level of accuracy. The present study adopts ant colony optimization (ACO) algorithm to develop a novel ACO-SVM model to solve this problem. The proposed algorithm is applied on some real world benchmark datasets to validate the feasibility and efficiency, which shows that the new ACO-SVM model can yield promising results.

1. Introduction

Support vector machines (SVM), a powerful machine learning methods for classification and regression problems of small samples and high dimensions, was initially presented by Vapnik in the last decade of the 20th century based on statistical learning theory and structural risk minimization principle (Vapnik, 1998, 1999). SVM has found application in tackling many pattern recognition problems in different fields, such as machinery condition monitoring and fault diagnosis (Hu, He, Zhang, & Zi, 2007; Yuan & Chu, 2006; Yuan & Chu, 2007), bioinformatics (Byvatov & Schneider, 2003), handwriting recognition (Bahlmann, Haasdonk, & Burkhardt, 2002), digital image processing (Cheng & Wang, 2006; Li, Fevens, & Krzyżak, 2006), time series forecasting (Lau & Wu, 2008; Thissen, van Brakel, & Weijer, 2003), financial and bankruptcy prediction (Kim, 2003; Wu, Tzeng, & Goo, 2007). It is noteworthy that the kernel parameters have influence on the generalization performance of SVM. The regularization constant C determines the trade-off between minimizing the training error and minimizing model complexity. The parameters of the kernel function implicitly define the non-linear mapping from input space to high-dimensional feature space (Duan, Keerthi, & Poo, 2003). As the performance of SVM will be weakened if these parameters are not properly chosen, it is an indispensable step to optimize the parameters of SVM for a good performance in handling a learning task. It needs either an exhaustive search over the parameter space or an optimized procedure that explores only a finite subset of the possible values (Imbault & Lebart, 2004). The rationale for optimizing the parameters of SVM makes it necessary to estimate the generalization error and find its global minimum over the parameter space. Several techniques have been developed so far: trial and error procedures, grid algorithm, cross validation method, generalization error estimation and gradient descent methods, and evolutionary algorithm.

In real applications, many practitioners select empirically by trying a finite number of parameter values and keeping those that bring the least test error. Apart from consuming enormous time, such trial and error procedures for selecting the parameters of SVM may not obtain the best performance because it is imprecise and the result is unreliable (Imbault & Lebart, 2004).

Grid search method is another common way to find proper parameters for SVM. This procedure requires a grid search over the parameter space. The parameters vary with the fixed step-size through a wide range of values. The performance of each parameter combination is assessed by some performance measure. Grid search is only suitable for adjustment of very few parameters and does not perform well in practice because it is complex in computation and time consuming (Friedrichs & Igel, 2005).

The leave-one-out (LOO), an unbiased estimate of the generalization error, has the shortcoming of computational complexity. The most commonly used method is cross validation, which also requires long and complicated calculation.

The most elaborate systematic techniques for parameter optimization are based on the generalization error estimation and gradient descent. In 2001, Chapelle, Vapnik, Bousquet, and Mukherjee (2002) proposed an automatic method by minimizing some estimates of the generalization error of SVM with a gradient descent algorithm over the set of parameters. Keerthi (2002) proposed a method for obtaining both the leave one-out (LOO) error and its gradient with respect to parameters for L2 soft margin of SVM. Adankon and Cheriet (2007) developed an improved method based on the empirical error by using two techniques: (1) Approximation of the gradient of error, which enables determination of the gradient without inverting the Gram–Schmidt matrix, thus reducing computational complexity of the gradient; (2) incremental

* Corresponding author. Tel./fax: +86 29 82663689.
E-mail address: lilyzhang83@hotmail.com (X. Zhang).

0957-4174/$ - see front matter
doi:10.1016/j.eswa.2010.03.067

learning strategy, which makes it possible to optimize the parameters of SVM. These iterative gradient-based algorithms usually rely on the smoothed approximation function to assess the performance of the parameters, and might not properly solve the parameter optimization problem if the starting point is not proper.

In recent years, the development of parameter optimization for SVM is supported by artificial intelligent (AI) techniques and evolutionary strategy. Friedrichs and Igel (2005) proposed a covariance matrix adaption evolution strategy (CMA-ES) to determine multiple parameters for SVM. The objective functions in SVM optimization can follow a global trend that is superposed by local minima, thus the number of objective parameters and the optimization accuracy can be increased when an evolutionary algorithm is used (Friedrichs & Igel, 2005). Wu, Tzeng, and Lin (2009) developed a novel hybrid genetic algorithm for parameter optimization in support vector regression, which is then applied to forecast the maximum electrical daily load.

Ant colony optimization (ACO) algorithm was first introduced by Dorigo and his colleagues as a novel nature-inspired method for the solution of combinatorial optimization (CO) problems in the early 1990s (Dorigo & Blum, 2005). From then on, researchers have successfully applied ACO to many optimization problems such as job shop scheduling (Blum, 2005), vehicle routing (Bell & McMullen, 2004), continuous optimization problems (Socha & Dorigo, 2008), global optimum function (Toksari, 2006, 2007), feature selection (Kanan, Faez, & Taheri, 2007; Sivagaminathan & Ramakrishnan, 2007). The ACO algorithm is easy to realize, which only involves basic mathematic operation. The most important is that the parallelism and distributional characteristics have no demanding requirements for CPU and memory, which ensures the capability of processing massive data. The existing researches have proved the effectiveness of ACO as a new method to obtain satisfactory global optimization results, which facilitates the selection of optimal parameters for SVM.

This study adopts ACO approach to present a novel ACO-SVM model for parameter optimization problem of SVM. In the following, the basic idea of SVM is introduced in Section 2. A brief introduction of ACO algorithm is presented in Section 3. The proposed novel ACO-SVM model for parameter optimization problem of SVM is explained in Section 4. In Section 5 the established ACO-SVM model algorithm is conducted on several real word datasets and the experimental results are discussed. In Section 6 we draw a general conclusion.

2. Support vector machines (SVM)

The basic idea of SVM is mapping the training samples from the input space into a higher dimensional feature space via a mapping function ϕ. Given a training set $S = \{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in H, y_i \in \{\pm 1\}, i = 1,2,\ldots,l\}$, where $\mathbf{x}_i$ are the input vectors and y_i the labels of the $\mathbf{x}_i$, the target function is

$$\begin{cases} \min \quad \Phi(\mathbf{w}) = \frac{1}{2}\langle \mathbf{w}\cdot\mathbf{w}\rangle + C\sum_{i=1}^{l}\xi_i \\ s.t. \quad y_i(\langle \mathbf{w}\cdot\phi(\mathbf{x}_i)\rangle + b) \geq 1-\xi_i, \quad \xi_i \geq 0 \quad i=1,2,\ldots,l, \end{cases} \tag{1}$$

where C is a penalty parameter, ξ_i are non-negative slack variables.

So the problem of constructing the optimal hyperplane is transformed into the following quadratic programming problem:

$$\begin{cases} \max \quad L(\alpha) = \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i,\mathbf{x}_j) \\ s.t. \quad \sum_{i=1}^{l}\alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C, \quad i=1,2,\ldots,l. \end{cases} \tag{2}$$

The decision function can be shown as:

$$f(\mathbf{x}) = \text{sign}\left[\sum_{i=1}^{l} y_i\alpha_i K(\mathbf{x}_i\cdot\mathbf{x}) + b\right]. \tag{3}$$

The most common kernel functions used in SVM are shown as follows:

Linear kernel

$$\text{K}(\mathbf{x},\mathbf{x}_i) = \langle \mathbf{x}\cdot\mathbf{x}_i\rangle. \tag{4}$$

Polynomial kernel

$$\text{K}(\mathbf{x},\mathbf{x}_i) = (\langle \mathbf{x}\cdot\mathbf{x}_i\rangle + c)^{\text{d}}. \tag{5}$$

RBF kernel

$$\text{K}(\mathbf{x},\mathbf{x}_i) = \exp(-\|\mathbf{x}-\mathbf{x}_i\|^2/2\sigma^2). \tag{6}$$

In this study we focus on the radial basis function kernel for its good performance and universal application.

3. Basic ideas of ant colony optimization (ACO) algorithms

Ant algorithms are optimization algorithms inspired by the foraging behavior of real ants in the wild. When searching for food, ants initially explore the area surrounding their nest in a random manner. As soon as an ant finds a food source, it evaluates the quantity and quality of the food and carries some of it back to the nest. During the return trip, the ant deposits a chemical pheromone trail on the ground. The quantity of pheromone deposited, which may depend on the quantity and quality of the food, will guide other ants to the food source. The indirect communication between the ants via pheromone trails enables them to find the shortest paths between the nest and food sources. This characteristic of real ant colonies is exploited in artificial ant colonies in order to solve difficult combinatorial optimization problems. In ant colony algorithm, artificial ants probabilistically build solutions by taking into account dynamical artificial pheromone trails. The central component of ACO algorithm is the pheromone model including the state transition rule and updating rule, which is used to probabilistically sample the search space. The ACO problem can be defined as follows:

Definition 1. A model $P = (S, \Omega, T)$ of an ACO problem consists of Dorigo and Blum (2005):

- a search (or solution) space S defined over a finite set of discrete decision variables and a set Ω of constraints among the variables;
- an objective function $T{:}S \to R^+$ to be minimized.

The search space S is defined as follows: Given a set of n discrete variables $\mathbf{U}_i$ with values $a_i^j \in D_i = \{a_i^1,\ldots,a_i^{|D_i|}\}, i=1,\ldots,n$, a variable instantiation, that is, the assignment of a value a_i^j to a variable U_i, is denoted by $U_i = a_i^j$. A feasible solution $s \in S$ is a complete assignment (an assignment in which each decision variable has a domain value assigned) that satisfies the constraints. If the set of constraints Ω is empty, then each decision variable can take any value from its domain independently of the values of the other decision variables. In this case, we call P an unconstrained problem model, otherwise a constrained problem model. A feasible solution $s^* \in S$ is called a globally optimal solution (or global optimum), if $T(s^*) \leqslant T(s)\ \forall s \in S$. The set of globally optimal solutions is denoted by $S^* \subseteq S$. To solve an ACO problem, one has to find a solution $s^* \in S^*$.

The framework of a basic ACO algorithm is shown in Table 1. At the start of the algorithm, the ACO problem model is inputted, and some variants are initialized. The basic ingredient of any ACO algorithm is a constructive heuristic for probabilistically constructing solutions. At each iteration, ants exploit a given pheromone model