An Approach for Constructing Sparse Kernel Classifier

Zejian Yuan, Yanyun Qu, Yang Yang, and Nanning Zheng
Institute of Artificial Intelligence and Robotics
Xi’an Jiaotong University
Xianning Road 28, Xi’an 710049, P.R.China
{zjyuan, yyqu}@aiar.xjtu.edu.cn

Abstract

This paper presents a new approach for constructing sparse kernel classifier with large margin. Firstly, we propose a kernel function pursuit strategy for selecting a small number of kernel functions which are used for expanding final classifier. And then an added constraint controls the sparseness of the final classifier and an approach is provided to solve the optimization problem with $L^2$ loss function and complexity measure. The experiment results show that sparse kernel classifier can achieved higher efficiency for both training and testing without sacrificing prediction accuracy.

1. Introduction

Support vector machines are powerful pattern recognition techniques and have been successfully applied to many machine learning tasks [7]. Its decision surface is linearly combined with many kernel functions. So runtime of SVMs is proportional to the number of kernel function. If that number is large, the classifier is considerably slower in testing.

To improve this disadvantage, many methods have been proposed. Burges and Scholkopf proposed a method for computing an approximation to the decision surface in terms of a reduced set of vectors [1][2]. The reduced set (RS) method chooses the smallest set of examples such that the resulting generalization may be lost, and the reduced set method use conjugate gradient computation which is very computationally expensive. Tom Downs [4] present an algorithm that allows unnecessary support vectors to be recognized and eliminated while leaving the solution unchanged, but the sparseness of classifier is not enough. In [6], the reduced support vector machine algorithm is proposed too, which randomly selects some vectors from the training samples to construct classifier, and then computes the expansion coefficients. As the vectors are chosen randomly and may not be good representatives of the training data, good classification performance is not obtained. Frauke proposes an evolution strategy to determine the kernel function from a parameterized kernel space and to control the regularization[5]. In fact, how to construct sparse classifier involves combinatorial search RKHS. We will resort to some suboptimal search methods. Some search strategy have appeared in the literature in different guises, for example, basis selection methods, basis pursuit, matching pursuits [3]. These algorithms are the basic of our method for selecting kernel function. Furthermore, given kernel functions, we will focus on how to construct a classifier with large margin.

In this paper, we presents a new approach for constructing sparse kernel classifier with large margin. The classifier consists of a little number of kernel functions, and this constraint is added into optimization problem to control the sparseness of the classifier. Meanwhile, an approach is provided to solve the optimization problem with $L^2$ loss function and complexity measure. Compared to other methods, the sparse kernel classifier trained by the proposed constructing method is faster in runtime and little computationally expensive in training phase, and it has better generalization.

In section 2, we briefly overview support vector classification and Reproducing Kernel Hilbert Space (RKHS). Sectoin 3 presents a residual minimization technique and sequential kernel pursuit strategy in RKHS. Section 4 provides an approach to solve optimization problem with $L^2$ loss function and an added constraint. Experimental results are collected in section 5.

2. Support vector classification using kernel technology

2.1. Support vector analysis and RKHS

SVM algorithms may be broadly interpreted as regularization algorithms with hinge loss functions and complexity
measures in an appropriately chosen Reproducing Kernel Hilbert Space (RKHS). Consequently those algorithms can obtain a solution from RKHS, which has minimal empirical error.

Given a Mercer kernel $K : X \times X \rightarrow R$, there is an associated RKHS $H_K$ of function $f : X \rightarrow R$ with a corresponding norm $\|f\|_K$. For a set of labeled samples $\{(x_i, y_i)\}, i = 1, 2, \cdots, l$, the standard kernel method can estimate an unknown function $f^* \in H_K$ by minimizing

$$f^* = \arg \min_{f \in H_K} \left\{ \frac{1}{l} \sum_{i=1}^{l} Loss(y_i, f(x_i)) + \beta \|f\|_K^2 \right\}$$

(1)

where $Loss(y_i, f(x_i))$ is the $L^2$ loss function $Loss(y_i, f(x_i)) = (y_i - f(x_i))^2$ in this paper. We call this method as minimum square loss classifier (MSLC). Penalizing the RKHS norm imposes smoothness conditions on possible solutions, and $\beta$ is a regularization parameter which is used for balancing empirical error and complexity measures. The classical representation theorem states that the solution to this minimization problem exists in $H_K$ and can be represented as

$$f^* = \sum_{i=1}^{l} \alpha_i K(x_i, \cdot)$$

(2)

Therefore, the problem is reduced to optimizing over the finite dimensional space of coefficients $\{\alpha_i\}, i = 1, 2, \cdots, l$. To solve the optimization problem (1), the coefficients $\alpha_i$ have close form solution

$$\alpha = [\gamma I + K]^{-1} Y$$

(3)

where $I$ is a unit matrix, and $K$ is the kernel matrix, i.e $K_{ij} = K(x_i, x_j), Y = [y_1, y_2, \cdots, y_l]$.

In order to decrease the computation complexity, we need a sparse classifier model in testing. In other words, we will present a kernel pursuit algorithm to choose a small number of kernel function.

3. Kernel pursuit for sparse classifier

3.1. Sparse approximation in RKHS

Let $f \in H_K$ has many good properties including lower training error and good generalization performance. But the $f$ equation (2) is not a sparse model, which is linearly expansion over a set of kernel function form feature space $H_K$. Given $f$ is an approximation to $f$, which consists of a small number of kernel functions

$$\hat{f}(x) = \sum_{j=1}^{k} \beta_j K(z_j, x)$$

(4)

where $z_j \in X, K(z, \cdot) \in H_K$. Obviously, $\hat{f} \in H_K$. In RKHS, the approximation error can be measured by norm $\|\hat{f} - f\|_K$, that is

$$E(\hat{f}, f) = \left\| \hat{f} - f \right\|_K^2$$

This is done by successive approximations of $f$ with projections on a kernel function from $H_K$.

3.2. Sequential kernel pursuit in RKHS

According to sequential forward greedy algorithms [3], Kernel pursuit approach is an iterative algorithm. This procedure is repeated each time on the following residue obtained.

Let the first kernel function $K(z, \cdot) \in H_K$, then $f$ can be written as

$$f = \beta K(z, \cdot) + R_f$$

(6)

where $R_f$ is the residual after approximating $f$ in the direction of kernel function $K(z, \cdot)$. Considering the reproducing property of kernel $K$, the norm of residual $R_f$ can be represented

$$\|R_f\|_K^2 = \langle f - \beta K(z, \cdot), f - \beta K(z, \cdot) \rangle$$

$$= \|f\|_K^2 - 2\beta \langle f, K(z, \cdot) \rangle + \beta^2 \|K(z, \cdot)\|_K^2$$

(7)

Given the kernel function $K(z, \cdot)$, let $\frac{\partial \|R_f\|_K^2}{\partial \beta} = 0$, and then optimal projection coefficient $\beta$ is

$$\beta = \frac{\langle f, K(z, \cdot) \rangle}{\langle K(z, \cdot), K(z, \cdot) \rangle}$$

(8)

Substituting approximation error (7) using (8),

$$\|R_f\|_K^2 = \langle f, f \rangle - \left( \frac{\langle f, K(z, \cdot) \rangle}{K(z, z)} \right)^2$$

(9)

Obviously, to minimize $\|R_f\|_K^2$, we must choose a kernel function $K(z, \cdot) \in H_K$ such that $|\langle f, K(z, \cdot) \rangle|$ is maximum. The optimal kernel function can be obtain by solving the following optimization problem

$$K(z, \cdot) = \arg \max_{K(z, \cdot) \in H_K} \left\{ \left| \langle f, K(z, \cdot) \rangle \right|^2 \right\}$$

(10)

Let $R_0 f = f$. We suppose that we have computed the $n^{th}$ order residue $R^n f$, for $n \geq 0$. Considering Gaussian kernel function, $K(z, z) = 1$, we choose the kernel function

$$K(z, \cdot) = \arg \max_{K(z, \cdot) \in H_K} \left\{ \left| \langle R^n f, K(z, \cdot) \rangle \right| \right\}$$

(11)
The residue $R^nf$ is sub-decomposed into

$$R^nf = \langle R^nf, K(z^{n+1}, \cdot) \rangle K(z^{n+1}, \cdot) + R^{n+1}f$$  \hfill (12)

Once the kernel $K(z^n, \cdot)$ is selected, we compute the inner product of new residue $R^{n+1}f$ with any kernel function $K(z, \cdot) \in H_K$, with an updating the following formula

$$\langle R^{n+1}f, K(z, \cdot) \rangle = \langle R^nf, K(z, \cdot) \rangle - \langle R^nf, K(z^n, \cdot) \rangle \cdot \langle K(z^n, \cdot), K(z, \cdot) \rangle$$  \hfill (13)

The $(n+1)^{th}$ optimal kernel function $K(z^{n+1}, \cdot)$ is selected according to the following formula

$$K(z^{n+1}, \cdot) = \arg \max_{K(z, \cdot) \in H_K} \{|\langle R^{n+1}f, K(z, \cdot) \rangle|\}$$  \hfill (14)

Each kernel function can be obtained by using gradient based optimization algorithm. The number of iteration we sub-decompose the residues of a given classification function surface $f$ depends upon the desired precision.

3.3. Pursuit kernel functions associated with training samples

The surface of classifier is highly irregular, therefore, it is very difficult to select optimal kernel function in RKHS according to equation (11). The kernel function selected is not a good solution because of random initialization. Here we present a kernel pursuit scheme that is firstly and search kernel functions from a small dictionary $\Delta = \{K(x_i, \cdot)\}_{i=1,2,\cdots,l}$, which is consisted of kernel functions centered on training samples set.

Firstly, we need to obtain the projection of $R^0f$ on kernel function $K(x_i, \cdot), i = 1, 2, \cdots, l$, that is $\langle R^0f, K(x_i, \cdot) \rangle$. And then the kernel function with maximum projection will be selected.

$$K(z^*, \cdot) = \arg \max_{K(z, \cdot) \in \Delta} \{|\langle R^0f, K(z, \cdot) \rangle|\}$$

$$= \arg \max_{K(x_i, \cdot) \in \Delta} \left\{ \sum_{j} \alpha_j K(x_j, x_i) \right\}$$  \hfill (15)

According to equation (13), the project of $n^{th}$ residual $R^nf$ on other kernel functions can be obtained, that is $\langle R^nf, K(x_i, \cdot) \rangle, i = 1, 2, \cdots, l$. The $n^{th}$ kernel function is selected by using formula (15), which is regard as an initialization for gradient based algorithm (14).

Given the kernel matrix $K_{ij} = K(x_i, x_j)$, we need only a little computation cost to find an initialization value for problem (14). The kernel pursuit strategy which searches in the small dictionary associated with training samples can find better kernel function than random initialization method.

4. Constructing sparse kernel classifier with large margin

Above approximation sparse classifier may be loss generalization. In order to obtain a sparse kernel classifier with good prediction ability, we need to add a constrain condition into the original optimization problem (1). Suppose kernel functions $K(z_j, \cdot)$ are found by using above kernel pursuit strategy, the classifier with large margin can be obtained by solving the following optimization problem

$$\min_{\beta} \left\{ \frac{1}{2} \sum_{i=1}^{l} (y_i - f(x_i))^2 + \gamma_2 \|f\|_K^2 \right\}$$

$$\text{s.t.} \quad f(x) = \sum_{j=1}^{k} \beta_j K(z_j, x)$$  \hfill (16)

It can be seen that the above problem is standard optimization problem (1) with the constrain condition. According to equation (16), objection function of optimization problem is given as follow

$$L(\beta) = Y^T Y - 2Y^T K_{xz} \beta + \beta^T \left[ \gamma_2 K_z + K_{xz} K_{xz} \right] \beta$$  \hfill (17)

where the matrix $K_{xz}$ is defined as $K_{ij}^{xz} = K(x_i, z_j)$, and kernel matrix $K_z$ is defined as $K_{ij}^{z} = K(z_i, z_j)$. Obviously, equation (17) is a quadratic objective function, the coefficients $\beta$ has global optimal solution

$$\beta = \left[ \gamma_2 K_z + K_{xz} K_{xz} \right]^{-1} K_{xz} Y$$  \hfill (18)

While $k = l$, $z_j = x_j, j = 1, 2, \cdots, l$, this is a standard margin algorithm with $L^2$ loss function, i.e. original optimization problem (1).

Constructing a sparse kernel classifier with large margin algorithm:

1. Construct initial classifier according to equation (2).
2. Find the kernel function $K(z_j, \cdot)$ by using gradient based optimization algorithm (14), each kernel function is initialized by using search method (15).
3. Given kernel function $K(z_j, \cdot), j = 1, 2, \cdots, k$, find sparse kernel classifier with large margin by solving optimization problem (16). That is to compute the coefficients by using close form representation (18).

5. Experiments

In this paper, a Gaussian kernel is used in the experiments

$$K(x_i, x_j) = \exp \left( -\frac{\|x_i - x_j\|^2}{2\sigma^2} \right)$$  \hfill (19)
Table 1. Results on the six classification benchmarks.

<table>
<thead>
<tr>
<th>Data set</th>
<th>SVM (Ns)</th>
<th>MSLC</th>
<th>SKC (Ns)</th>
<th>RVM (Ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast C.</td>
<td>27.1%(115.3)</td>
<td>25.2%</td>
<td>25.8% (8.2)</td>
<td>29.9%(6.3)</td>
</tr>
<tr>
<td>Banana</td>
<td>10.5%(153.4)</td>
<td>10.7%</td>
<td>11.9% (12.9)</td>
<td>10.8%(11.4)</td>
</tr>
<tr>
<td>Heart</td>
<td>16.3%(105.2)</td>
<td>17.4%</td>
<td>17.3% (11.8)</td>
<td>————</td>
</tr>
<tr>
<td>Titanic</td>
<td>22.3%(87.6)</td>
<td>23.1%</td>
<td>22.9% (3.2)</td>
<td>23.0%(65.3)</td>
</tr>
<tr>
<td>Waveform</td>
<td>9.9%(150.5)</td>
<td>9.8%</td>
<td>9.7% (9.9)</td>
<td>10.9%(14.6)</td>
</tr>
<tr>
<td>German</td>
<td>23.6%(412.4)</td>
<td>22.7%</td>
<td>23.6% (11.9)</td>
<td>22.2%(12.5)</td>
</tr>
</tbody>
</table>

Six classification benchmark datasets are used: Breast Cancer, Banana, Heart, Titanic, Waveform and German. These data sets are downloaded from website: http://ida.first.fraunhofer.de/projects/bench/. For each data set, there are 100 training/test splits and our results show averages over the first 50 of those.

The parameters for different approach are used: For standard SVM, we use the same parameters provided by Gunnar from the above website. We perform 5-fold cross validation on the training set to select parameters for the minimum square loss classifier (MSLC), i.e optimization problem (1), and for sparse kernel classifier (SKC) i.e optimization problem (16), the same kernel parameters as MSLC are used, and regularization parameters need adaptively adjust by using $N_s/l$, where $N_s$ is the number of kernel function selected. In the experiment, the parameters for different data sets are used: Breast Cancer $[\sigma = 1.7, \gamma = 10.0]$, Banana $[\sigma = 0.35, \gamma = 1.0]$, Heart $[\sigma = 3.0, \gamma = 1.0]$, Titanic $[\sigma = 3.5, \gamma = 5.0]$, German $[\sigma = 3.0, \gamma = 2.0]$, Waveform $[\sigma = 2.0, \gamma = 1.0]$.

The test results of each algorithm on the six classification benchmarks are presented in Table 1. The classification error rate (%) and the number of support vector (Ns) for each classifier is the averages over 50 training/test splits data sets. The support vectors for standard SVM are support vectors which lie nearly boundary of the classifier or error classification samples. The $N_s$ of MSLC are all training samples, and the $N_s$ of SKC are some representations of training samples which are from input space and not always training samples. In Table 1, benchmark examples indicate SKC has better sparseness while its predictive ability can be hold. The results of the relevance vector machine (RVM) are taken directly from [8], which is averages over the first 10 of data sets. Our algorithm (SKC) is better than RVM on Titanic and Waveform data sets.

Test errors of MSLC and SKC on 50 training/test splits Breast Cancer dataset are clearly shown in Figure 1. We see that test error of the two algorithms are almost equivalent and better than standard SVM in Table 1, but SKC has only a small number of kernel functions and it will be faster in testing.

6. Conclusion

We proposed an approach for constructing sparse kernel classifier with large margin, which essentially finds some kernel functions from RKHS to construct a sparse classifier. The added constraint can control the sparseness of the final classifier, and the provided approach is effective for solving the optimization problem with $L^2$ loss function and complexity measure. The experiment results show that the final sparse classifiers has a small number of kernel function and good generalization.

References


