Distance between Two Classes: A Novel Kernel Class Separability Criterion

Jiancheng SUN††, Member, Chongxun ZHENG†††, Nonmember, and Xiaohe LI†, Student Member

SUMMARY  With a Gaussian kernel function, we find that the distance between two classes (DBTC) can be used as a class separability criterion in feature space since the between-class separation and the within-class data distribution are taken into account implicitly. To test the validity of DBTC, we develop a method of tuning the kernel parameters in support vector machine (SVM) algorithm by maximizing the DBTC in feature space. Experimental results on the real-world data show that the proposed method consistently outperforms corresponding hyperparameters tuning methods.

Key words: kernel parameter, data classification, class separability, support vector machine

1. Introduction

Class separability is a classic concept in pattern recognition [1], [2]. A widely used separability measure is based on the scatter matrices of data. For example, Fisher scalar is used as a separability measure which considers the between-class separation and the within-class data distribution synchronously. Thus it has been used elegantly to solve the problems of pattern recognition such as tuning the kernel parameter [3], feature extraction [4] and kernel matrix learning [5], [6]. However, the main drawback of Fisher scalar is the singularity problem introduced by within-class scatter matrices when the number of training samples is not adequate.

Support vector machine (SVM) [7], is gaining popularity due to many attractive features and promising empirical performance. The selection of kernel functions is an important researching branch in the area of SVM. Though the performance of SVM algorithm mainly depends on choice of parameters such as Gaussian kernel parameter $\sigma$ and penalty parameter $C$, tuning these hyperparameters is a hard work.

A straightforward way to select the hyperparameters is the grid search algorithm [11], [12]. Though good results can be obtained, this parameter selection method trains SVM for parameter combinations and thus they are time consuming. At present, the Gaussian kernel is the most common one due to its good features [8]. Since the Gaussian kernel parameter $\sigma$ is closely associated with generalization performance of SVM, how to choose an appropriate $\sigma$ is worth pursuing.

In the past several decades, application of the distance between two classes (DBTC) in feature space was neglected since the main criticism is that it seems to take into account only the between-class separation. However, based on the analysis of DBTC, in this paper we find DBTC reflects not only the between-class separation, but also within-class data distribution implicitly provided that the Gaussian kernel function is used. Thus the DBTC can be used as a class separability measure. To test the validity of DBTC, we present a method to tune the kernel parameter $\sigma$ in SVM algorithm by maximizing DBTC in feature space. Employing the DBTC as objective function, we develop a gradient-based algorithm to search the optimal kernel parameter.

2. Distance between Two Classes

2.1 Definition of DBTC

Given the training data $(x_i, y_i)$, $i = 1, 2, \ldots, l$, where $x_i$ is the $i$th input vector and $y_i$ is the target value. $y_i = 1$ denotes that $x_i$ is in class 1 and $y_i = -1$ denotes that $x_i$ is in class 2. In the case of kernel method, two classes data in the feature space are given as $\varphi(x_i), i = 1, \ldots, n_1, \varphi(x_j), j = 1, \ldots, n_2$, where $\varphi : \mathbb{R}^d \rightarrow \mathbb{F}$ is a nonlinear map that maps $x_i$ from input space to a high dimensional feature space and $n_1 + n_2 = l$. The mapping $\varphi$ is not given explicitly in practice. Instead, a kernel trick $k(x_i, x_j) = \langle \varphi(x_i), \varphi(x_j) \rangle$ is used to give the inner product value of $x_i$ and $x_j$ in the feature space. The means of two classes are defined as

$$m_1^T = \frac{1}{n_1} \sum_{i=1}^{n_1} \varphi(x_i), \quad m_2^T = \frac{1}{n_2} \sum_{j=1}^{n_2} \varphi(x_j)$$

Therefore the distance between two classes (DBTC) is defined as

$$D(\sigma^2) = \|m_1^T - m_2^T\|^2 = \frac{1}{n_1^2} 1^T K_{1,1} 1 + \frac{1}{n_2^2} 1^T K_{2,2} 1 - \frac{2}{n_1 n_2} 1^T K_{1,2} 1$$

(1)
where \( \mathbf{1} = (1, \ldots, 1)^T \), \( [\mathbf{K}_{1,1}]_{i,j} = k(\mathbf{x}_{i,1}, \mathbf{x}_{j,1}) \), \( [\mathbf{K}_{2,2}]_{i,j} = k(\mathbf{x}_{i,2}, \mathbf{x}_{j,2}) \), \( [\mathbf{K}_{1,2}]_{i,j} = k(\mathbf{x}_{i,1}, \mathbf{x}_{j,2}) \) and \( k(\cdot) \) is the popular Gaussian kernel function kernel and is defined as

\[
k(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{||\mathbf{x} - \mathbf{y}||^2}{2\sigma^2}\right), \quad \sigma > 0 \tag{2}\]

where the kernel parameter is the Gaussian width \( \sigma \).

2.2 Data Distribution Property of DBTC in Feature Space

The first criticism of DBTC is that it does not seem to take into account the within-class data distribution, but only the between-class separation. We might want to consider a criterion that balanced both of these two factors. However, DBTC actually reflects this balance impliedly provided that the Gaussian kernel function is used. To explain the distribution of DBTC, we define the sum of squared distances between two classes \( C_p, C_q \) in feature space as follows

\[
d(C_p, C_q) = \sum_{\mathbf{x}_p \in C_p} \sum_{\mathbf{x}_q \in C_q} \left( \varphi(\mathbf{x}_p) - \varphi(\mathbf{x}_q) \right)^2
\]

\[= \mathbf{1}^T \mathbf{K}_{pq} \mathbf{1} + \mathbf{1}^T \mathbf{K}_{pp} \mathbf{1} - 2 \cdot \mathbf{1}^T \mathbf{K}_{pq} \mathbf{1} \tag{3}\]

where \( [\mathbf{K}_{p,q}]_{i,j} = k(\mathbf{x}_{p,i}, \mathbf{x}_{q,j}) \), \( [\mathbf{K}_{p,p}]_{i,j} = k(\mathbf{x}_{p,i}, \mathbf{x}_{p,j}) \), and \( [\mathbf{K}_{p,q}]_{i,j} = k(\mathbf{x}_{p,i}, \mathbf{x}_{q,j}) \). When \( p = q \), \( d(C_p, C_q) \) represents the sum of the within-class squared distances. By introducing the Gaussian kernel function and considering \( k(x, x) = 1 \), Eq. (1) can be rewritten as

\[
D(\sigma^2) = \left( 2 - \frac{2}{n_1n_2} \mathbf{1}^T \mathbf{K}_{1,1} \mathbf{1} \right) - \left( 1 - \frac{2}{n_1^2} \mathbf{1}^T \mathbf{K}_{1,2} \mathbf{1} \right) - \left( 1 - \frac{2}{n_2^2} \mathbf{1}^T \mathbf{K}_{2,2} \mathbf{1} \right)
\]

\[= \frac{d(C_1, C_2)}{n_1n_2} - \frac{d(C_1, C_1)}{2n_1^2} - \frac{d(C_2, C_2)}{2n_2^2} \tag{4}\]

From (4), it is easy to see that the first term represents average class-distance distances, while the second and third terms represent the average within-class distances. Thus, a large \( D(\sigma^2) \) indicates that patterns are close to each other if they are from the same class but far from each other if they are from different classes. That is to say \( D(\sigma^2) \) reflects the combination of between-class and within-class data distribution. This is a very important feature to be used as a separability measure. More importantly, we find that \( D(\sigma^2) \) has a global maximum with respect to \( \sigma^2 \) on all the test data sets and the details will be presented in the experimental section.

Based on the definition of DBTC, the benefit of using it as a class separability measure can be summarized as follows

1. To serve as a class separability measure, it is necessary to take into account within-class data distribution and the between-class separation simultaneously. Similar to Fisher scalar, DBTC is such a criterion to balance these two factors.

2. Comparing with Fisher scalar, DBTC can avoid the singularity problem introduced by within-class scatter matrices that is the main drawback of Fisher scalar. In addition, computational load for calculating DBTC is low since calculating the inverse of within-class scatter matrix is not necessary.

3. Hyperparameters Tuning Based on DBTC

To test the validity of DBTC, this section we develop a method of tuning the hyperparameters for SVM. Tuning the kernel parameter \( \sigma \) and tuning \( C \) are different for SVM. The kernel function and its parameters only determine the geometry structure of the data and \( C \) is relation to the optimization algorithm. In fact, \( C \) has no intuitive meaning of the geometry. The variation of \( C \) can only affect the range of \( D(\sigma^2) \) and have no effect on its shape. Therefore the geometry of data in feature space cannot provide any information to tune \( C \).

Based on above mentioned point, we think tuning kernel parameter \( \sigma \) and \( C \) should divide into two independent processes. First process is to tune the \( \sigma \) using geometry information of the data in feature space. DBTC is introduced here as a criteria to tune \( \sigma \). Based on (1), we think the optimal \( \sigma \) can be obtained when \( D(\sigma^2) \) reach its maximum. This means that the optimal \( \sigma \) is corresponding to optimal class separability. In the second process, some classical methodology, such as cross validation error [11] and generalization error [9], can be used to tune the \( C \). Here we utilized the cross validation error to tune the \( C \) for comparing with the traditional methods.

To maximize \( D(\sigma^2) \), we follow the standard gradient approach. It is easy to calculate the gradient of the \( D(\sigma^2) \) as

\[
\frac{\partial D}{\partial \sigma^2} = \frac{1}{n_1^2} \sum_{i=1}^{n_1} \sum_{j=1}^{n_1} \frac{\partial k(\mathbf{x}_{i,1}, \mathbf{x}_{j,1})}{\partial \sigma^2} + \frac{1}{n_2^2} \sum_{i=1}^{n_2} \sum_{j=1}^{n_2} \frac{\partial k(\mathbf{x}_{i,2}, \mathbf{x}_{j,2})}{\partial \sigma^2}
\]

\[= \frac{2}{n_1n_2} \sum_{i=1}^{n_1} \sum_{j=1}^{n_1} \frac{\partial k(\mathbf{x}_{i,1}, \mathbf{x}_{j,2})}{\partial \sigma^2} \tag{5}\]

The derivatives of kernel with respect to \( \sigma^2 \) can be derived easily as

\[
\frac{\partial k(\mathbf{x}, \mathbf{x})}{\partial \sigma^2} = k(\mathbf{x}, \mathbf{x}) \frac{||\mathbf{x} - \mathbf{x}||^2}{2\sigma^4} \tag{6}\]

Since \( D(\sigma^2) \) is smooth, the optimal \( \sigma^2 \) can be easily obtained by general gradient-based optimal algorithms. Here we use the quasi-newton method to solve this maximal problem. When the optimal \( \sigma^2 \) is obtained we employ the cross validation techniques [11] to find the optimal \( C \).

4. Computational Experiments

In this section, we describe the experimental results in order to investigate the performance of utilizing the selected
parameters in SVM. For comparison, cross validation techniques (CV), span of support vectors (SP) and radius / margin bound (R/M) are used to tune the hyperparameters in this paper. SP and R/M are proposed based on the leave-one-out (LOO) error rate for the SVM [9]. R/M is defined as

\[ f_{R/M} = \frac{1}{l} R^2 \| w \|^2 \]

where \( R \) is the radius of the smallest sphere that contains all \( \varphi(x_i) \) vectors, \( l \) is the number of training examples, and \( \| w \|^2 \) is the reciprocal of the margin in SVM. The concept of SP is defined as

\[ f_{SP} = \frac{1}{k} \sum_{p=1}^{k} \phi \left( \alpha_p S_p^2 - 1 \right) \]

where \( S_p \) is the distance between the mapped point \( \varphi(x_p) \) of the support vector \( x_p \), \( \alpha_p \) can be obtained by solving the SVM, \( k \) is the number of support vectors, and \( \phi \) is the step function: \( \phi(x) = 1 \) when \( x > 0 \) and \( \phi(x) = 0 \) otherwise.

A set of 6 real-world benchmark data sets from the Gunnar Raetsch [10] is used to evaluate the generalisation properties. Table 1 describes some basic information about these data sets. These data sets have zero mean and unit variance, and 100 different random training and test splits are defined.

In common cases, it is not easy to choose good initial values for \( \sigma^2 \) and \( C \) unless we have sufficient knowledge about the data set. For the DBTC, different initial values are selected randomly to test the algorithm convergence. We find that same optimal \( \sigma^2 \) can always be obtained with all the real-world data sets. These results show that the DBTC has a unique maximum for these benchmark datasets. To confirm this point, we plot the variation of DBTC with respect to \( \ln(\sigma^2) \) in Fig. 1. From the Fig. 1, we can see that a unique maximum can be found for all the data sets. Therefore, selection of initial values for our method becomes easy.

However, too small value for \( \sigma^2 \) is not advisable since the derivatives of Gaussian kernel is very small in this case. This problem may lead to the optimization algorithm stop in initial stages for the small derivative. We choose the initial value is \( \sigma^2 = 0.4 \) for all data sets. After obtaining the optimal \( \sigma^2 \), \( C \) is selected from \{10^{-1}, 1, 5, 10, 10^2, 10^3\} using the cross validation technique which is a grid search method [11].

Using the parameters tuned by our DBTC-CV Methods and a range of state-of-the-art statistical pattern recognition algorithms, Table 2 shows the outcome of a comparison of the classification error of SVM algorithm. Here SP and R/M denote the Chapelle’s methods utilized span of support vectors and radius / margin bound [9]. CV represent the cross validation techniques and the results come form Mika et al [10].

Comparing with SP and R/M, we see that for the DBTC-CV there is a remarkable improvement in the performance for all the data sets. Obviously, choosing \( \sigma^2 \) based on DBTC is an appropriate strategy provided that an optimal \( C \) is used. In other words, DBTC reflect the data geometry structure in feature space. A best separability can be obtained when the DBTC reach its maximum. More importantly, the significance of proposing DBTC lies in not only tuning the kernel parameter, but other issues in pattern recognition such as feature selection, feature extraction and kernel matrix learning.

![Fig. 1 Variation of DBTC with respect to \( \ln(\sigma^2) \).](image-url)
5. Conclusion

We have proposed in this paper a new method for selecting the hyperparameters for SVM based on the distance between two classes (DBTC). From the point of view of DBTC definition, it can be seen that DBTC does not take into account the within-class data distribution, but only the between-class separation. However, we have found in this paper that DBTC actually reflects within-class data distribution impliedly provided that the Gaussian kernel function is used. By maximizing DBTC, we have developed a gradient-based algorithm to find the optimal kernel parameter. The experimental results demonstrate that the DBTC-based methods outperform the SP, R/M on all the benchmark datasets.

References