Differential Privacy Protection for Support Vector Machines for Nonlinear Classification

Yuxian Huang, Geng Yang, Yunlu Bai, and Hua Dai

1 College of Computer Science and Software, Nanjing University of Posts and Telecommunications, Nanjing, Jiangsu 210023, China
2 Jiangsu Key Laboratory of Big Data Security and Intelligent Processing, Nanjing, Jiangsu 210023, China

Correspondence should be addressed to Yuxian Huang; 2021040402@njupt.edu.cn

Received 23 December 2021; Accepted 31 August 2022; Published 23 September 2022

Academic Editor: Xiaohui Yuan

Copyright © 2022 Yuxian Huang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Currently, private data leakage and nonlinear classification are two challenges encountered in big data mining. In particular, few studies focus on these issues in support vector machines (SVMs). In this paper, to effectively solve them, we propose a novel framework based on the concepts of differential privacy (DP) and kernel functions. This framework can allocate privacy budgets and add artificial noise to different SVM locations simultaneously, which makes the perturbation process freer and more delicate. In addition, under this framework, we propose three algorithms, DP SVMs that perturb the training data set, perturb the kernel function, and utilize mixed perturbation (DPSVM-TDP, DPSVM-KFP, and DPSVM-MP, respectively), all of which can realize accurate classification while ensuring that the users’ privacy is not violated. Moreover, we conduct privacy analysis on these algorithms and prove that they all satisfy \((\varepsilon, 0)\)-DP. Finally, we conduct experiments to evaluate the algorithms in terms of different aspects and compare them with the DPSVM with dual-variable perturbation (DVP) algorithm (DPSVM-DVP) to determine the optimal perturbation method. The results show that DPSVM-KFP can achieve the highest data utility and strictest privacy protection with the shortest running time.

1. Introduction

With the development of information technology, a tremendous amount of data has been collected, which makes the extraction of useful information an imperative and challenging issue. As an effective way to retrieve potentially valuable information hidden in massive data, big data classification [1] has penetrated many aspects of society, such as medical prediction, video semantic analysis [2], and human activity recognition [3, 4].

However, most of the existing classification algorithms still face two challenges [5, 6]. First, these algorithms generally put personal information at risk because the attacker may infer the user’s private information from the classification results. At the same time, people are becoming increasingly concerned with the use of their sensitive data. To guarantee the fundamental right of individuals to privacy, appropriate measures to protect data against disclosure risks must be implemented before classification results are published. Unfortunately, traditional privacy protection technologies, including k-anonymity [7], l-diversity [8], and t-closeness [9], are all built on the premise that attackers own no background knowledge. In fact, attackers can infer user privacy from published classification results by referring to some additional user information. Therefore, these technologies cannot provide adequate security. Differential privacy (DP) [10], an emerging and provable privacy protection model, exhibits many substantial advantages over previous privacy protection methods. DP is based on solid mathematical theory and quantifies privacy levels through privacy budgets, making privacy protection a comparable concept. Second, most classification algorithms do not perform well when dealing with nonlinearly separable data sets. Under such conditions, samples with different labels may become clustered, making it difficult to directly identify a classification plane to distinguish them in the sample space.
Kernel method [11] is an excellent solution to deal with nonlinear classification because it can project samples to a high-dimensional feature space, where these samples can be separated by a hyperplane.

As a kind of predominant supervised learning model and data classification algorithm, support vector machines (SVMs) [12] are widely deployed in many applications. Recently, in order to prevent sensitive information leakage, several studies have been conducted on DP protection in SVM algorithms. These studies realize DP protection by adding artificial noise to SVMs. The effort in [13] obfuscates an input data set systematically through independently and identically distributed noise, thereby providing privacy protection for SVMs. Without disturbing the given data set, the research in [14, 15] perturbs the output of an SVM by adding random noise to the normal vector classification hyperplane. In addition, other researchers are mainly focused on perturbing some SVM parameters. For instance, the research in [16] perturbs the dual variables of the hyperplane, while the work of [17] perturbs the kernel function.

According to the above works, we find that although researchers have made many contributions to achieving DP protection for SVMs, there are still some shortcomings. First, the DP protection approaches of SVMs lack a general theoretical framework. Second, most studies in this area focus only on privacy protection and ignore the problem that most data sets in real tasks are nonlinearly separable. Third, the work of [17] complete classification and privacy protection for nonlinearly separable data by adding noise to the kernel function, but in this paper, the sensitivity of the kernel function and the scale of the added noise are not analysed. In addition, they only consider the radial basis function (RBF) as the kernel function and do not take the linear kernel and sigmoid kernel into account.

To effectively solve information leakage and nonlinear classification in SVMs, this paper proposes a novel framework based on the concepts of DP and the kernel method to perturb training data sets and kernel functions by purposely adding noise. A literature review finds that this approach is the first attempt to develop a general DP protection framework for SVMs. Furthermore, under this framework, DPSVM algorithms that perturb the training dataset, perturb the kernel function, and utilize mixed perturbation (DPSVM-TDP, DPSVM-KFP and DPSVM-MP, respectively) are proposed, all of which are able to classify nonlinearly separable datasets and satisfy \((\varepsilon, 0)\)-DP. Furthermore, in DPSVM-KFP, we consider perturbing different kernel functions and complete an adequate privacy analysis on them. The main contributions of this paper are summarized as follows:

(i) We propose a novel and general framework that organically combines DP, the kernel method, and an SVM. This framework can realize the classification of nonlinear separable data sets while limiting the risk of privacy leakage by purposely perturbing the training data set and kernel function.

(ii) We propose three algorithms based on the developed framework that utilize the Laplace mechanism to implement \((\varepsilon, 0)\)-DP protection for SVM. The DPSVM-TDP and DPSVM-KFP algorithms add Laplace noise to the training data set and kernel function, respectively. The DPSVM-MP algorithm adopts a mixed perturbation method to disturb both the training data set and the kernel function. Moreover, these algorithms can be applied to the classification of nonlinearly separable data sets.

(iii) We compute the global sensitivities of both the training data set and the kernel function, which is not an easy task in DP since it requires the consideration of all possible data sets differing by one record. Then, we prove that these algorithms satisfy \((\varepsilon, 0)\)-DP based on global sensitivities under a certain noise perturbation level with the Laplace mechanism. In addition, we conduct extensive experiments on three University of California, Irvine (UCI) data sets to evaluate the performance of our algorithms. Furthermore, via a comparison with the DPSVM algorithm with dual-variable perturbation (DPSVM-DVP), we verify the efficiency of our algorithms.

The remainder of this paper is organized as follows. In Section 2, we show some related work and existing shortcomings. Section 3 introduces SVMs, the kernel method, and DP. Section 4 proposes a novel framework and three specific algorithms. In Section 5, we calculate the sensitivity levels of the algorithms and provide proofs of their \((\varepsilon, 0)\)-DP. We provide the experimental results and related analysis in Section 6. Section 7 concludes this paper.

2. Related Work

In recent years, some researchers have begun to utilize DP to realize privacy protection for SVMs. The ways in which such studies implement privacy protection are mainly divided into input perturbation, output perturbation, and algorithm perturbation methods [18].

Input perturbation usually disturbs the training data set. There is no doubt that this method is relatively straightforward, but the conditions of privacy protection are serious, and sometimes the classification effect is unsatisfactory. Liu [10] considers the problem of publicly releasing a data set for SVM classification while not infringing on the privacy of data subjects. An input data set is systematically obfuscated to protect privacy through noise, whose optimal distribution is determined by maximizing the weighted sum of privacy and utility metrics. Senekane [19] implements a privacy-preserving image classification scheme by adding noise to every pixel in the image. By controlling the scale of noise, he makes the scheme meet the requirements of differential privacy. In contrast to input perturbation, output perturbation is the most commonly employed method by researchers to accomplish DP protection for SVMs. The key to this method is to perturb the classification hyperplane of the given SVM, thus achieving reliable data classification under the condition of privacy protection. For instance, Xu et al. [14] use the Laplace mechanism to add random noise to the
classification hyperplane so that the attackers cannot recover the data set through the model parameters. Cai et al. [15] consider the problem of unbalanced distribution of customer data and implements a weighted DP-SVM algorithm by adding noises of different scales to the output. The last method is algorithm perturbation, which adds noise to an intermediate value. For example, Zhang et al. [16] propose the DPSVM-DVP algorithm, which does not directly perturb the normal vector of the classification hyperplane but perturbs the dual variable of the support vector. The algorithm first solves the dual problem of an SVM by the SMO method, and the difference between the estimated value and real value of each support vector is recorded. Then, it calculates the ratio of each support vector to the sum of all support vectors and adds different levels of Laplace noise to dual variables corresponding to these ratios. Wang and Li [17] propose a privacy-preserving SVM algorithm under DP for multiclassification. The algorithm disturbs the kernel function in three different ways: direct Laplace noise injection, Taylor formula replacement, and a combination of the previous two methods. The purpose of the algorithm is to protect small sample data while not interfering with the overall classification effect of the model.

In summary, such studies implement DP protection by adding noise to certain positions in SVMs. However, the existing studies fail to consider a general DP protection framework for SVMs, which forms the major task of our study. Compared with those of conventional works, the framework of our study can allocate privacy budgets and add artificial noise to different locations of SVMs simultaneously. In other words, this general framework can help to choose the locations and scales of perturbation more freely and delicately to deliver a more accurate classification result while ensuring the security of individuals’ privacy. Moreover, previous studies seldom take nonlinearly separable data sets into account, a shortcoming that is addressed in the framework of this study.

3. Preliminaries and Definitions

In this section, we introduce the related concepts of SVM, the kernel method and DP.

3.1. Support Vector Machines. SVM [20] is a binary classification model that defines a classifier that maximizes the minimum margin in the feature space. Specifically, for a data set \( D = \{(x_i, y_i)\}_{i=1}^n \), the number of samples is \( n \), and each sample has \( d \) features. In addition, \( x_i \) and \( y_i \) are the features and labels of the \( i \)th sample, respectively, where \( y_i \in \{-1, 1\} \).

Then, the classification decision function can be defined as

\[
f(x) = w^T x + b,
\]

where \( w \) and \( b \) are the normal vector and intercept of the classification hyperplane, respectively.

According to the idea of maximizing the classification hyperplane with the minimum margin, the original problem can be transformed into a constrained SVM optimization problem, which is usually described as follows:

\[
\begin{align*}
\min_{w, b} & \quad \frac{1}{2} \|w\|^2, \\
\text{s.t.} & \quad y_i (w^T x + b) \geq 1, i = 1, 2, \ldots, n.
\end{align*}
\]

Notably, the objective function is a convex function. Therefore, the Lagrange multiplier method [21] can be used to solve the problem so that we can obtain the following formula:

\[
\begin{align*}
\min_{w, b} & \quad \frac{1}{2} \|w\|^2 - \sum_{i=1}^n \alpha_i (y_i (w^T x + b) - 1), \\
\text{s.t.} & \quad \alpha_i \geq 0, i = 1, 2, \ldots, n.
\end{align*}
\]

\( \alpha_i \) is the Lagrange multiplier.

Then, we can calculate the derivatives of \( w \) and \( b \) and set the derivative functions equal to 0.

\[
\begin{align*}
w & = \sum_{i=1}^n \alpha_i y_i x_i, \\
0 & = \sum_{i=1}^n \alpha_i y_i.
\end{align*}
\]

Substituting the result into (1), we can obtain the following formula:

\[
f(x) = \sum_{i=1}^n \alpha_i y_i x_i^T x + b.
\]

3.2. Kernel Method for SVMs. The kernel method [22] is a significant way to solve nonlinear classification issues, especially in SVMs, because it can project samples to a higher dimensional feature space, making the sample separable. Therefore, if we assume that \( \phi(x) \) is the feature vector projected by \( x \), then according to (4), the model corresponding to the classification hyperplane can be expressed as

\[
f(x) = \sum_{i=1}^n \alpha_i y_i \phi(x_i)^T \phi(x) + b.
\]

Next, we can assume that there is a function that equals the inner product of \( x_i \) and \( x_j \) in the feature space; that is,

\[
K(x_i, x_j) = \langle \phi(x_i)^T \phi(x_j) \rangle.
\]

The function \( K \) is called the kernel function. Furthermore, for SVMs, there are three commonly used kernel functions: linear kernel, Gaussian kernel, and sigmoid kernel functions [23, 24].

The expression of the linear kernel function is

\[
K(x_i, x_j) = x_i^T x_j.
\]

The expression of the Gaussian kernel function is

\[
K(x_i, x_j) = \exp\left(-\gamma\|x_i - x_j\|^2\right), \gamma > 0.
\]

\( \gamma \) indicates the influence of a single sample on the entire classification hyperplane.
The expression of the sigmoid kernel function is
\[ K(x_i, x_j) = \tanh(y x_i^T x_j + c), \quad y > 0, c \geq 0. \] (10)

\( \tanh \) is the hyperbolic tangent function. \( y \) indicates the influence of a single sample on the entire classification hyperplane. \( c \) indicates the independent item in the sigmoid kernel.

3.3. Differential Privacy

**Definition 1** (DP [25]) A randomized mechanism \( M \) is considered to satisfy \((\varepsilon, 0)\)-DP if for any two neighboring data sets \( D \) and \( D' \) (with at most one different sample) and for all outputs \( O \) \((\text{O} \subseteq \text{Range}(M))\),
\[
\Pr[M(D) \in O] \leq e^\varepsilon \times \Pr[M(D') \in O],
\] (11)
where \( \varepsilon > 0 \) is the privacy budget controlling the strength of the privacy protection. A lower \( \varepsilon \) ensures a greater privacy guarantee.

**Definition 2** (Global Sensitivity [26]) For a function \( f : D \rightarrow \mathbb{R}^d \) and any two neighboring data sets \( D \) and \( D' \), the global sensitivity of \( f \) is defined as
\[
\Delta f = \max_{D, D'} \| f(D) - f(D') \|_1.
\] (12)

In DP, the sensitivity \( \Delta f \) can be understood as the approximate adjacency degree of the two data sets for the function \( f \).

**Definition 3** (Laplace Mechanism [27]). Given a data set \( D \), for a function \( f : D \rightarrow \mathbb{R}^d \) with sensitivity \( \Delta f \), the mechanism \( M \) provides \((\varepsilon, 0)\)-DP satisfying:
\[
M(D) = f(D) + \text{lap}\left( \frac{\Delta f}{\varepsilon} \right).
\] (13)

\( \text{lap}(\bullet) \) represents the noise sampled from the Laplace distribution with a scaling parameter \( b \), and its probability density function is
\[
g(x) = \frac{1}{2b}e^{-|x|/b}. \] (14)

The Laplace mechanism is a common method for implementing DP protection. It is realized by adding random noise that satisfies the Laplace distribution. Furthermore, the scale of the noise is determined by the global sensitivity \( \Delta f \) and the privacy budget. Therefore, when Laplace noise is added, the result of the user query is not a fixed value but rather an uncertain random number. The probability density function of the random number obeys the Laplace distribution.

**Definition 4** (Composability of DP [28]). Suppose that the privacy budgets of algorithms \( M_1, M_2, \ldots, M_n \) are \( \varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n \); then, for the same data set \( D \), the combined algorithm \( M(M_1(D), M_2(D), \ldots, M_n(D)) \) satisfies \((\sum_{i=1}^n \varepsilon_i)\)-DP.

<table>
<thead>
<tr>
<th>Notations</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>( D )</td>
<td>A set of samples</td>
</tr>
<tr>
<td>( x_i )</td>
<td>Feature of ( i )th sample</td>
</tr>
<tr>
<td>( y_i )</td>
<td>Label of ( i )th sample</td>
</tr>
<tr>
<td>( n )</td>
<td>The number of samples</td>
</tr>
<tr>
<td>( d )</td>
<td>The number of attributes</td>
</tr>
<tr>
<td>( \varepsilon )</td>
<td>The privacy budget</td>
</tr>
<tr>
<td>( p )</td>
<td>Proportional parameter</td>
</tr>
<tr>
<td>( \Delta D )</td>
<td>Global sensitivity of the training data set</td>
</tr>
<tr>
<td>( \mathcal{D} )</td>
<td>Training data set after adding noise</td>
</tr>
<tr>
<td>( K )</td>
<td>Kernel function</td>
</tr>
<tr>
<td>( \Delta K )</td>
<td>Global sensitivity of the kernel function</td>
</tr>
<tr>
<td>( \mathcal{K} )</td>
<td>Kernel function after adding noise</td>
</tr>
<tr>
<td>( \gamma )</td>
<td>Parameter of Gaussian kernel and sigmoid kernel</td>
</tr>
<tr>
<td>( c )</td>
<td>Independent item</td>
</tr>
<tr>
<td>( N )</td>
<td>Laplace noise matrix</td>
</tr>
</tbody>
</table>

This property indicates that an algorithm composed of a sequence of DP protection algorithms has a privacy budget that is equal to the sum of all individual privacy budgets.

4. Framework and Algorithms

In this section, we propose an SVM framework that satisfies DP protection for non-linearly separable data. Moreover, we propose three algorithms based on the developed framework. The DPSVM-TDP and DPSVM-KFP algorithms perturb the training data set and kernel function, respectively. The DPSVM-MP algorithm uses a mixed perturbation method to disturb both the training data set and the kernel function. Table 1 shows the definitions of the notations that will be used in this paper.

4.1. Proposed Framework. Given that nonlinear classification and privacy leakage are two urgent challenges faced by SVMs, this study utilizes a kernel function and DP to address them by proposing a general framework, as shown in Figure 1.

First, an input data set is preprocessed to obtain the model’s input, which is represented as \( \{(x_i, y_i)\}_{i=1}^n \); this means that there are \( n \) samples in total, the feature of the \( i \)th sample is \( x_i \) and its label is \( y_i \). For most data sets, the samples may be nonlinearly separable. In other words, in the current dimension, it is impossible to determine a plane to distinguish samples with different labels. Therefore, these samples must be projected into a higher dimensional space to find a hyperplane in the high-dimensional space and distinguish the samples. However, this process greatly increases the incurred computational costs, so the kernel method is adopted to simplify the calculation.

According to formula (6), the kernel function is equal to the inner product of \( x_i \) and \( x_j \) in a high-dimensional space. As a result, the kernel function \( K \) between different features is calculated and used as the input for model training. Furthermore, to protect the data from being leaked, this study adds noise to the features of the training data set or kernel function. It is also necessary to select a reasonable noise-adding mechanism and noise scale parameters for the
Algorithm to meet the protection conditions of DP. Common noise-adding mechanisms include Gaussian mechanisms, Laplace mechanisms, exponential mechanisms, and so on. The selection of the noise scale parameters depends on the available privacy budget and the required global sensitivity. Finally, the classification hyperplane is acquired after training the model.

Based on this general framework, we propose three specific algorithms by adjusting the artificial noise. The DPSVM-TDP algorithm disturbs the training data set, the DPSVM-KFP algorithm disturbs the kernel function, and the DPSVM-MP algorithm adds noise to both.

4.2. DPSVM-TDP Algorithm. The DPSVM-TDP algorithm is a method of input perturbation that realizes $(\varepsilon, 0)$-DP by adding noise to the training data set. In DPSVM-TDP, the data set is preprocessed mainly to normalize the features so that they are in the interval $[0, 1]$. We assume that $x_{ij}$ is the $j$th feature of the $i$th sample in the data set $D$, and $x'_{ij}$ is the feature after normalization. The normalization formula can be expressed as follows:

$$x'_{ij} = \frac{x_{ij} - \min_{i=1,2,...,n} x_{ij}}{\max_{i=1,2,...,n} x_{ij} - \min_{i=1,2,...,n} x_{ij}} \quad (15)$$

Subsequently, we adopt the Laplace mechanism to add artificial noise to the training data set. According to formula (12), the scale parameter of the noise is determined by both the global sensitivity of the training data set and the privacy budget. The process of adding artificial noise can be expressed by the following formula:

$$\mathcal{D} = D + lap\left(\frac{\Delta D}{\varepsilon}\right), \quad (16)$$

where the scale parameter of the Laplace noise is $\Delta D/\varepsilon$. $\Delta D$ is the global sensitivity of the training data set, and its calculation is shown in Section 5. $\mathcal{D}$ is used to represent the training data set after adding noise.

Next, we project the features of the samples in $D$ to the high-dimensional space and use the kernel function to replace the inner product between the features in the high-dimensional space. Therefore, we can obtain the expression of the classification hyperplane as follows:

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b. \quad (17)$$

Finally, we train the model and input the test data set to achieve classification accuracy.

The DPSVM-TDP Algorithm 1 is described as follows.

4.3. DPSVM-KFP Algorithm. In DPSVM-KFP, the method of preprocessing the data set is the same as that in DPSVM-TDP. In addition, we do not need to add noise to the training data set during the process of perturbation. According to Section 3.2, we find that the kernel function can be used to replace the inner product of two high-dimensional vectors in an SVM to solve the issue of nonlinear classification. On this basis, to implement $(\varepsilon, 0)$-DP, Laplace noise is added to the kernel function. The scale of the noise is determined by both the global sensitivity of the kernel function and the privacy budget, and we obtain the following formula:

$$K = K + lap\left(\frac{\Delta K}{\varepsilon}\right), \quad (18)$$

where the scale parameter of the Laplace noise is $\Delta K/\varepsilon$. $\Delta K$ is the global sensitivity of the kernel function, and its calculation is also shown in Section 5. $K$ is used to represent the kernel function after adding noise. Then, we can obtain the formula of the perturbed classification hyperplane, as shown below:

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i K(x_i, \bar{x}) + b. \quad (19)$$
Input: data set $D = \{(x_i, y_i)\}_{i=1}^n$, samples $n$, attributes $d$, privacy budget $\epsilon$.
Output: classification accuracy with Laplace noise added.

1. Normalize the data set so that the features of any sample $x_i$ belong to the interval $[0, 1]$;
2. Add Laplace noise to the training data set $D$. The scale parameter of noise is $\Delta D/\epsilon$. $\Delta D$ is the global sensitivity of the training data set $D$ used to represent the training data set after adding noise;
3. Project the low-dimensional data $D$ to the high-dimensional feature space, and the mapped feature vector $\varphi(x_i)$ is obtained;
4. Describe the classification decision function as $f(x) = \sum_{i=1}^n a_i y_i \varphi(x_i) + b$;
5. Substitute the inner product of vector with kernel function and obtain the function $f(x) = \sum_{i=1}^n a_i y_i K(x_i, x) + b$;
6. Train the SVM model and compute the classification decision function;
7. Input test data and achieve the classification accuracy with Laplace noise added.

Algorithm 1: DPSVM-TDP.

Input: data set $D = \{(x_i, y_i)\}_{i=1}^n$, samples $n$, attributes $d$, privacy budget $\epsilon$.
Output: classification accuracy with Laplace noise added.

1. Normalize the data set so that the features of any sample $x_i$ belong to the interval $[0, 1]$;
2. Project the low-dimensional data $D$ to the high-dimensional feature space, and the mapped feature vector $\varphi(x_i)$ is obtained;
3. Describe the classification decision function as $f(x) = \sum_{i=1}^n a_i y_i \varphi(x_i) + b$;
4. Substitute the inner product of vector with kernel function and obtain the function $f(x) = \sum_{i=1}^n a_i y_i K(x_i, x) + b$;
5. Add Laplace noise to the kernel function $K(x_i, x)$. The scale parameter of noise is $\Delta K/\epsilon$. $\Delta K$ is the global sensitivity of the kernel function. $K(x_i, x)$ is used to represent the kernel function after adding noise;
6. Train the SVM model and compute the classification decision function;
7. Input test data and achieve the classification accuracy with Laplace noise added.

Algorithm 2: DPSVM-KFP.

Finally, we train the model and obtain the classification accuracy on the test data set. The DPSVM-KFP algorithm is described as follows.

Notably, Algorithm 2 realizes DP protection by perturbing the kernel function. This method is classified as a perturbation algorithm because it adds noise to the parameters of the given SVM.

4.4. DPSVM-KFP Algorithm. In Sections 4.2 and 4.3, we perturb the training data set and the kernel function, respectively, to achieve DP protection. In this part, according to the composability of DP, we decide to allocate the available privacy budget to the training data set and the kernel function proportionally and realize DP protection by adding noise to both of them simultaneously.

The way in which we perturb the training data set is shown in

$$\bar{D} = D + lap\left(\frac{\Delta D}{pe}\right),$$

where the scale parameter of the Laplace noise is $\Delta D/pe$. $p$ is the proportional parameter, which denotes the proportion of the privacy budget that we allocate to the training data set.

Moreover, the way in which we perturb the kernel function is shown in

$$\bar{K} = K + lap\left(\frac{\Delta K}{(1-p)\epsilon}\right),$$

where the scale parameter of the Laplace noise is $\Delta K/(1-p)e$ and $(1-p)e$ is the proportion of the privacy budget that we allocate to the kernel function. The remaining steps are similar to those in the DPSVM-TDP and DPSVM-KFP algorithms. The DPSVM-MP algorithm is described as follows.

In Algorithm 3, we assign a privacy budget of $pe$ to the training data set and assign a privacy budget of $(1-p)e$ to the kernel function. According to Definition 4, we know that Algorithm 3 can finally accomplish $(\epsilon, 0)$–DP. The specific proof is elaborated in Section 5.

5. Privacy Analysis

In this section, we conduct a privacy analysis on the above three algorithms. We first calculate the global sensitivity levels of the training data set and the kernel function and then prove that these algorithms all satisfy DP.

5.1. Global Sensitivity of the Training Data set. According to the Laplace mechanism of DP, the scale of the added noise is related to the privacy budget and global sensitivity. Since the privacy budget is adjustable, we also need to calculate the global sensitivity to determine the amount of noise added. From Section 4.1, we know that there are two disturbed objects: the training data set and the kernel function, so we should calculate their global sensitivity levels.

Lemma 1. The global sensitivity of training data set equals to $d$. 

Input: data set $D = \{(x_i, y_i)\}_{i=1}^n$, samples $n$, attributes $d$, privacy budget $\epsilon$, proportional parameter $p$.
Output: classification accuracy with Laplace noise added.

(1) Normalize the data set so that the features of any sample $x_i$ belong to the interval $[0, 1]$.
(2) Add Laplace noise to the training data set $D$. The scale parameter of noise is $\Delta D/p$. $\Delta D$ is the global sensitivity of the training data set. $\mathcal{D}$ is used to represent the training data set after adding noise.
(3) Project the low-dimensional data $\mathcal{D}$ to the high-dimensional feature space, and the mapped feature vector $\varphi(x_i)$ is obtained.
(4) Describe the classification decision function as $f(x) = \sum_{i=1}^n a_i y_i \varphi(x_i) + b$.
(5) Substitute the inner product of vector with kernel function and obtain the function $f(x) = \sum_{i=1}^n a_i y_i K(x_i, x) + b$.
(6) Add Laplace noise to the kernel function $K(x_i, x)$. The scale parameter of noise is $\Delta K/(1-p)\epsilon$. $\Delta K$ is the global sensitivity of the kernel function. $K(x_i, x)$ is used to represent the kernel function after adding noise.
(7) Train the SVM model and compute the classification decision function.
(8) Input test data and achieve the classification accuracy with Laplace noise added.

**Algorithm 3: DPSVM-MP.**

Proof. Suppose there are two adjacent data sets $D_1 = \{x_1, x_2, \ldots, x_m, x_n\}^T$ and $D_2 = \{x_1, x_2, \ldots, x_m, x_n'\}^T$, $x_i$ is the $i$-th record is different. If the global sensitivity of the training data set is denoted as $\Delta D$, according to the definition of global sensitivity,

$$\Delta D = \max \|D_1 - D_2\|_1.$$  

(22)

For matrix $C \in \mathbb{R}^{m \times d}$, $\|C\|_1 = \max_j \sum_{i=1}^m |c_{ij}| (1 < j < d)$, So $\Delta D$ can be expressed as

$$\Delta D = \max \|x_m - x_m'\|_1.$$  

(23)

Because we normalize the data set, the values of all features belong to the interval $[0, 1]$. Therefore, we can obtain the maximum value of $\Delta D$

$$\Delta D = d.$$  

(24)

5.2. Global Sensitivity of the Kernel Function. Regarding the kernel function, we have three different options, namely, a linear kernel, Gaussian kernel, and sigmoid kernel. According to Section 3.2, we know that these kernel functions are calculated in different ways. Therefore, their global sensitivities are also different.

Suppose there are adjacent data sets $D_1 = \{x_1, x_2, \ldots, x_m, x_n\}^T$ and $D_2 = \{x_1, x_2, \ldots, x_m, x_n'\}^T$, and the $i$-th record is different. $K_1$ is the kernel function of any two records in the data set $D_1$, while $K_2$ is the kernel function of any two records in the data set $D_2$. If we set the sensitivity of the kernel function as $\Delta K$, according to the definition of sensitivity,

$$\Delta K = \max \|K_1 - K_2\|_1.$$  

(25)

**Lemma 2.** The global sensitivity of linear kernel function equals to $d$.

Proof. According to Section 3.2, the linear kernel function $K$ can be expressed as

$$K(x_i, x_j) = x_i^T x_j.$$  

(26)

Thus, we can obtain the sensitivity of the linear kernel function.

$$K = \max \|K_1 - K_2\|_1 = \max \|x_i^T x_j - x_i'^T x_j'\|_1.$$  

(27)

Because the data set has been normalized, all the features of any sample $x_i$ belong to the interval $[0, 1]$. Therefore, when all the features in $x_i$ and $x_j$ take the value 1, and all the features in $x_i'$and $x_j'$ take the value 0, $\|K_1 - K_2\|_1$ achieves the maximized. Thus, the sensitivity of linear kernel function is

$$\Delta K = d.$$  

(28)

**Lemma 3.** The sensitivity of Gaussian kernel function equals to $(1 - \exp(-\gamma \cdot d))$.

Proof. According to Section 3.2, the Gaussian kernel function $K$ can be expressed as

$$K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|_2^2).$$  

(29)

Thus, we can obtain the global sensitivity of the Gaussian kernel function.

$$\Delta K = \max \|K_1 - K_2\|_1 = \max \{\exp(-\gamma \|x_i - x_j\|_2^2) - \exp(-\gamma \|x'_i - x'_j\|_2^2)\}_1 = \max \{\exp(-\gamma \|x_i - x_j\|_2^2) - \min \{\exp(-\gamma \|x'_i - x'_j\|_2^2)\}\}.$$  

(30)

Because the data set has been normalized, all the features of any sample $x_i$ belong to the interval $[0, 1]$. Therefore, when all the features in $x_i$ and $x_j$ have the same value, $\exp(-\gamma \|x_i - x_j\|_2^2)$ is the maximized.

$$\max \{\exp(-\gamma \|x_i - x_j\|_2^2)\} = \exp(\gamma \cdot 0) = 1.$$  

(31)

When all the features in $x'_i$ take the value 1 and all the features in $x'_j$ take the value 0, $\exp(-\gamma \|x'_i - x'_j\|_2^2)$ is the minimized.
According to Section 3.2, the sigmoid kernel function \( K \) can be expressed as
\[
K(x_i, x_j) = \tanh(\gamma x_i^T x_j + c).
\] (34)

Thus, we can obtain the sensitivity of the sigmoid kernel function.

\[
\Delta K = \max\|K_1 - K_2\|_1 = \max\|\tanh(\gamma x_i^T x_j + c) - \tanh(\gamma x_i'^T x_j' + c)\|_1.
\] (35)

The hyperbolic tangent function can be transformed as follows:
\[
\tanh x = \frac{e^x - 1/e^x}{e^x + 1/e^x} = 1 - \frac{2}{e^x + 1/e^x} = 1 - \frac{2}{e^{2x} + 1}
\] (36)

Therefore, we can transform \( \Delta K \) according to formula (36):
\[
\Delta K = \max\left\| 1 - \frac{2}{e^{2x} + 1/e^x} \right\|_1
\] (37)

Then, let \( A = \gamma x_i^T x_j + c \) and \( B = \gamma x_i'^T x_j' + c \), \( \Delta K \) can be expressed as
\[
\Delta K = \max\left\| \frac{2}{e^{2x} + 1} - \frac{2}{e^{2x} + 1} \right\|_1 = \max\left\| \frac{2}{e^{2x} + 1} \right\|_1 - \min\left\| \frac{2}{e^{2x} + 1} \right\|_1
\] (38)

In addition, function \( f(x) = 2/e^{2x} + 1 \) decreases monotonically when \( x \in [-\infty, +\infty] \). Moreover, since \( y > 0, c \geq 0, d > 0 \), \( 2/e^{2x} + 1 \) is maximized when \( B = c \) \( (x_i^T x_j') = 0 \) and \( 2/e^{2x} + 1 \) achieve the minimized when \( A = y + c \) \( (x_i^T x_j) = d \).

\[
\max\left\| \frac{2}{e^{2x} + 1} \right\|_1 = \frac{2}{e^{2x} + 1}, \quad \min\left\| \frac{2}{e^{2x} + 1} \right\|_1 = \frac{2}{e^{2(y+dc)} + 1}
\] (39)

Thus, the sensitivity of the Gaussian kernel function is
\[
\Delta K = \frac{2}{e^{2x} + 1} - \frac{2}{e^{2(y+dc)} + 1}
\] (40)

5.3. Differential Privacy Proof of Algorithms

**Theorem 1.** The DPSVM-TDP algorithm satisfies \((\varepsilon, 0)\)-DP.

**Proof.** Suppose that there are two adjacent data sets \( D_1 = \{x_1, x_2, \ldots, x_m, \ldots, x_n\} \) and \( D_2 = \{x_1, x_2, \ldots, x_m', \ldots, x_n\} \). Their ith records are different. \( N_1 \) and \( N_2 \) are the Laplace noise, which is added to the training data set \( D_1 \) and \( D_2 \), respectively. In addition, the scale parameter of noise is \( \Delta D/\varepsilon \). \( \overline{D} \) is the result of adding noise to the \( D_1 \) and \( D_2 \)
\[
\text{Pr}(D_1 + N_1 = \overline{D}) \leq \text{Pr}(N_1 = \overline{D} - D_1) \leq \text{Pr}(N_2 = \overline{D} - D_2) = e^{-\varepsilon/2\Delta D} \exp\left(-\varepsilon\|\overline{D} - D_1\|/\Delta D\right)
\] (41)

\[
\leq \exp\left(\frac{\varepsilon}{\Delta D}(\|\overline{D} - D_2\| - (\overline{D} - D_1))\right)
\] (42)

Therefore, the training data set after adding Laplace noise satisfies \((\varepsilon, 0)\)-DP. In addition, due to the consistency of DP, our subsequent operations on the kernel function do not affect its privacy protection. As a result, the DPSVM-TDP algorithm satisfies \((\varepsilon, 0)\)-DP.

**Theorem 2.** The DPSVM-KFP algorithm satisfies \((\varepsilon, 0)\)-DP.

**Proof.** Suppose that there are two adjacent data sets \( D_1 = \{x_1, x_2, \ldots, x_m, \ldots, x_n\}^T \) and \( D_2 = \{x_1, x_2, \ldots, x_m', \ldots, x_n\}^T \). Their ith records are different. \( K_i \) is the kernel function of any two records in the data set \( D_1 \), while \( K_2 \) is the kernel function of any two records in the data set \( D_2 \). \( N_1 \) and \( N_2 \) are the Laplace noise, which is added to the kernel functions \( K_1 \) and \( K_2 \). Besides, the scale parameter of noise is \( \Delta K/\varepsilon \). \( \overline{K} \) is the result of adding noise to the \( K_1 \) and \( K_2 \)
\[
\text{Pr}(K_1 + N_1 = \overline{K}) \leq \text{Pr}(N_1 = \overline{K} - K_1) \leq \text{Pr}(N_2 = \overline{K} - K_2) = e^{-\varepsilon/2\Delta K} \exp\left(-\varepsilon\|\overline{K} - K_1\|/\Delta K\right)
\] (43)

\[
\leq \exp\left(\frac{\varepsilon}{\Delta K}(|\overline{K} - K_2| - |\overline{K} - K_1|)\right)
\] (44)

Therefore, the kernel function after adding Laplace noise satisfies \((\varepsilon, 0)\)-DP. In addition, due to the consistency of DP, our subsequent operations on the kernel function do not
affect its privacy protection. As a result, the DPSVM-KFP algorithm satisfies $(\varepsilon, 0)\text{-DP}$.

**Theorem 3.** The DPSVM-MP algorithm satisfies $(\varepsilon, 0)\text{-DP}$.

**Proof.** Suppose there is a dataset $D$. $K$ is the kernel function. $M_1$ is a function that perturbs the training data set, and the noise scale parameter is $\Delta D/\varepsilon$. $M_2$ is a function that perturbs the kernel function, and the noise scale parameter is $\Delta K/(1 - p)\varepsilon$.

According to Theorem 1, we know that $M_1(D)$ satisfies $(\varepsilon_1, 0)\text{-DP}$. In addition, in line with Theorem 2, $M_2(K(D))$ satisfies $((1 - p)\varepsilon, 0)\text{-DP}$.

As a result, according to Definition 4, the privacy budget of algorithm DPSVM-MP is the sum of $M_1(D)$ and $M_2(K(D))$. Therefore, algorithm DPSVM-MP satisfies $(\varepsilon, 0)\text{-DP}$. □

### 6. Experimental Results and Analysis

In this section, we evaluate the performance of the DPSVM-TDP, DPSVM-KFP, and DPSVM-MP algorithms on three UCI data sets.

First, different from DPSVM-TDP and DPSVM-KFP, DPSVM-MP contains a parameter $p$, which may affect the performance of this algorithm. Therefore, we conduct experiments to analyse the classification accuracy of DPSVM-MP by varying the values of $p$ and the privacy budget $\varepsilon$. Second, according to the results of Experiment 6.1, we can determine the value of $p$ that maximizes the classification accuracy of DPSVM-MP and then conduct experiments to compare this algorithm with DPSVM-TDP and DPSVM-KFP. Third, to characterize the performance of our algorithms, we compare the algorithm that performs best in Experiment 6.2 with the DPSVM-DVP algorithm [13] in terms of their classification accuracies and running times. Three UCI data sets are used in our experiments: DryBean, Sensorless, and Covtype. The data set information is shown in Table 2.


In the DPSVM-MP algorithm, we allocate the privacy budget to the training data set and the kernel function proportionally and implement DP protection by adding noise to both simultaneously. According to Section 4.3, the privacy budget allocated to the training data set is $\varepsilon_1\varepsilon$, and the privacy budget allocated to the kernel function is $(1 - p)\varepsilon$. To analyze the performance of the DPSVM-MP algorithm when $p$ takes different values, we design the first experiment.

This experiment is based on the DryBean data set, which has a total of 13611 records, and each record has 16 features and one label. Moreover, there are two parameters ($y$ and $c$) in the Gaussian kernel and sigmoid kernel. In scikit-learn, $y$ and $c$ default to $1/n$ and 0, respectively, where $n$ is the number of records in the data set. Therefore, in this experiment, we also set $y$ and $c$ to $1/n$ and 0, respectively.

When the parameters $y$ and $c$ are fixed, we are able to observe the classification accuracy changes with different proportional parameters $p$ by adjusting the value of the privacy budget. The results are shown in Figures 2 and 3. Notably, regardless of which kernel function is adopted, the classification accuracy of the DPSVM-MP algorithm is
higher when the parameter $p$ is closer to 0.5. In other words, when the privacy budget is relatively evenly distributed to the training data set and the kernel function, the availability of the classification results is better. In addition, we find that the classification accuracy of the DPSVM-MP algorithm with the sigmoid kernel function is best, while the accuracy achieved when using the linear kernel is relatively low.

6.2. Comparison of the DPSVM-TDP, DPSVM-KFP, and DPSVM-MP Algorithms. In this section, to verify which perturbation algorithm is most efficient, we compare the classification accuracies of the DPSVM-TDP, DPSVM-KFP, and DPSVM-MP algorithms on the Sensorless data set. According to the previous experiment, we know that when the proportional parameter $p = 0.5$, the performance of the DPSVM-MP algorithm is best. Therefore, in the second experiment, the proportional parameter $p$ is set to 0.5. In addition, the two parameters $\gamma$ and $c$ in the Gaussian kernel and sigmoid kernel are still set to $1/n$ and 0, respectively.

The results of the experiment are shown in Figure 4. First, we find that the performance of the DPSVM-KFP algorithm is generally higher than that of DPSVM-TDP. This outcome is caused by the following reason. In DP, if the privacy budget $\varepsilon$ is fixed, the scale of the noise is proportional to the sensitivity of the parameters to be disturbed. In other words, if the sensitivity increases, the noise scale is enhanced, and the classification accuracy of the data worsens. DPSVM-TDP, as an input perturbation method, adds noise to the training data. According to Sections 5.1 and 5.2, the

![Figure 4: The classification accuracies of DPSVM-TDP, DPSVM-KFP, and DPSVM-MP. (a) Linear kernel. (b) Gaussian kernel. (c) Sigmoid kernel.](image-url)
sensitivity of the training data set is $d$. In contrast, DPSVM-KFP, as an algorithm perturbation method, adds noise to the kernel function. If we choose a linear kernel, the sensitivity is also $d$. At this time, the scale of the added noise is the same, but due to the different perturbation locations, there is a slight gap between the accuracies of the DPSVM-TDP and DPSVM-KFP algorithms. For the Gaussian kernel and sigmoid kernel, the sensitivities of the two corresponding kernel functions are $(1 - \exp(-yd))$ and $(2(e^{\gamma c} + 1 - 2(e^{\gamma c} - 1))$, respectively, which are much smaller than $d$. There is no doubt that in these two cases, the DPSVM-KFP algorithm performs much better than the DPSVM-TDP algorithm, especially when the privacy budget is relatively low.

In addition, the classification accuracy of DPSVM-MP is always lower than that of DPSVM-KFP and DPSVM-DVP. In DPSVM-MP, we evenly allocate the privacy budget to the training data set and the kernel function and disturb both of them. Moreover, whether for the training data set or the kernel function, the allocated privacy budget is only half of that in DPSVM-TDP and DPSVM-KFP. Therefore, the scale of the noise added in DPSVM-MP is twice that of DPSVM-KFP and DPSVM-TDP. For instance, in DPSVM-TDP, we add noise with a scale parameter of $\Delta d/\varepsilon$ to the training data set. However, in DPSVM-MP, we only have to add noise with a scale parameter of $\Delta d/\varepsilon$ to the training data set but also add noise with a scale parameter of $\Delta \alpha/(1 - \varepsilon)$ to the kernel function. In addition, since $\Delta d/\varepsilon$ is smaller than $\Delta d/\varepsilon$, the noise added in DPSVM-MP is greater. Therefore, the performance of DPSVM-MP, as shown in the experimental results, is worse than that of DPSVM-TDP and DPSVM-KFP.

In summary, to achieve DP protection for an SVM, the kernel function perturbation method has the best usability, perturbing the training data set is the second-best approach, and the method of mixed perturbation has the worst effect.

6.3. Performance Evaluation of the DPSVM-KFP and DPSVM-DVP Algorithms. In the previous experiment, we verified that the classification accuracy of DPSVM-KFP is superior to that of DPSVM-TDP and DPSVM-MP. To further evaluate the performance of DPSVM-KFP, in this section, we compare it with an SVM and DPSVM-DVP [16] on the Covtype data set.

DPSVM-DVP is an algorithm perturbation method that does not directly perturb the normal vector of the classification hyperplane but rather perturbs the dual variable of the support vector. The algorithm first solves the dual problem of SVMs by the SMO method, and the difference between the estimated value and real value of each support vector is recorded. Then, DPSVM-DVP calculates the ratio of each support vector to the sum of all support vectors and adds different levels of Laplace noise to the dual variables corresponding to these ratios.

Notably, both DPSVM-KFP and DPSVM-DVP are algorithm perturbation methods, but the disturbed parameters are quite different. The former disturbs the kernel function $K$, while the latter disturbs the dual variable $\alpha$. In the following experiment, for these two algorithms, we choose the sigmoid kernel function. In addition, the parameters $\gamma$ and $c$ are still set to $1/n$ and 0, respectively.

First, we analyse the relationship between the classification accuracy and the privacy budget. We fix the number of samples to 5000 here. The privacy budget epsilon is set to $0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1$ in succession. For each privacy budget, we conduct three experiments by using the SVM, DPSVM-KFP, and DPSVM-DVP algorithms. The results of the experiment are shown in Figure 5. Notably, since no noise is added by the SVM algorithm, it has the highest classification accuracy and is not affected by the privacy budget. In addition, when the privacy budget is greater than 0.5, the classification accuracy of the DPSVM-DVP algorithm is slightly higher than that of the DPSVM-KFP algorithm, both of which are above 0.8. However, if $\varepsilon \in [0.05, 0.1]$, the usability of DPSVM-KFP is significantly higher than that of DPSVM-DVP. Finally, when the privacy budget is less than 0.001, the classification accuracies of the two algorithms both tend to 0.5.

As a result, upon judging the usability of the algorithms, compared with that of DPSVM-DVP, the classification accuracy of DPSVM-KFP becomes more stable when the privacy budget decreases. In addition, we find that DPSVM-KFP is able to withstand more noise and maintain acceptable classification accuracy.

Second, we analyse the relationship between the classification accuracy and sample size. The privacy budget $\varepsilon$ is fixed to 1. The number of samples is set to 100, 500, 1000, 5000, 10000, 20000, and 40000 in succession. The results of the experiment are shown in Figure 6. As the number of samples increases, the classification accuracy of DPSVM-KFP and DPSVM-DVP exhibits upward trends. However, in general, DPSVM-KFP performs better, both at low and high data volumes.
Third, in addition to algorithm availability, running time is also an important criterion for measuring the pros and cons of an algorithm. Therefore, we fix the privacy budget to 1. The number of samples $n$ is set to 500, 1000, 5000, and 10000 in succession. Then, we observe the running times of the SVM, DPSVM-KFP, and DPSVM-DVP.

The results of the experiment are shown in Figure 7. It can be seen from the figure that the running times of the SVM and DPSVM-KFP are close and shorter than that of DPSVM-DVP. According to Section 4.2, in DPSVM-KFP, we only add noise to the calculated value of the kernel function. Therefore, the running time of DPSVM-KFP is close to that of the SVM. In contrast, we need to calculate the privacy budget allocated to each dual variable in the DPSVM-DVP algorithm, which undoubtedly requires a longer running time. Especially when dealing with large-scale data sets, our algorithm can save more running time. For example, when the number of samples is equal to 10,000, algorithm DPSVM-DVP takes 93.6 minutes, while our algorithm only takes 72.9 minutes, saving nearly 22% of the running time.

7. Conclusions

Currently, nonlinear classification and privacy data leakage are two major challenges. Few studies focus on these issues in SVMs.

To solve these problems, we propose a universal framework and three specific algorithms in this paper. Moreover, we conduct privacy analysis by calculating the sensitivities of the DPSVM-TDP, DPSVM-KFP, and DPSVM-MP algorithms. Then, we prove that all three algorithms satisfy $(\varepsilon, \delta)$-DP. Last, we complete three experiments and obtain the following results. (1) When we disturb the training data set and the kernel function simultaneously, the more evenly the privacy budget is allocated, the better the performance of the DPSVM-MP algorithm. (2) The method of perturbing the kernel function is more efficient than training data set perturbation and mixed perturbation, especially for the sigmoid kernel, and the classification accuracy is higher. (3) Compared with DPSVM-DVP, DPSVM-KFP has better classification accuracy and stability and a shorter running time.

In addition, there are some shortcomings in our research. For example, the linear kernel, Gaussian kernel, and sigmoid kernel are suitable for different types of data sets

![Figure 6: Classification accuracy variations with different numbers of samples (SVM, DPSVM-KFP, and DPSVM-DVP).](image1)

![Figure 7: The running times of the SVM, DPSVM-KFP, and DPSVM-DVP.](image2)
with SVM. However, on the Sensorless data set, the sigmoid kernel is able to endure a lower privacy budget than that required for the linear kernel and the Gaussian kernel. However, it is insufficient to compare the fitness levels of these three kernel functions with DP on a single data set. As a result, in the future, we will add Laplace noise to these kernel functions on different types of data sets and compare their classification accuracy.

Data Availability

The data set DryBean used to support the findings of this study is available through visiting: https://archive.ics.uci.edu/ml/datasets/Dry+Bean+Dataset/. The data set Sensorless used to support the findings of this study is available through visiting https://archive.ics.uci.edu/ml/datasets/Sensorless+for+Sensorless+Drive+Diagnosis/. The data set Covtype used to support the findings of this study is available through visiting https://archive.ics.uci.edu/ml/machine-learning-databases/covtype/.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61872197 and 61972209), the Postgraduate Research & Practice Innovation Program of Jiangsu Province of China (KYCX180891), the Natural Science Foundation of graduate Research & Practice Innovation Program of Jiangsu Province of China (61872197 and 61972209), the Postdoctoral Science Foundation Project of China (2016M601859).

References