www.cad-journal.net
Computer-Aided Design AND APPLICATIONS™

Improved BFGS Scheme Incorporated with Exponential Penalty Function for Support Vector Machine

Jingyi Li[1], Ruojun Zhao[2]

[1]University of Sydney, jili4090@uni.sydney.edu.au
[2]Xidian University, rjzhao2023@outlook.com

Corresponding author: Jingyi Li, jili4090@uni.sydney.edu.au

Abstract. This study introduces an improved Broyden-Fletcher-Goldfarb-Shanno (I-BFGS) scheme, meticulously designed for support vector machines (SVMs), laying the fundamental basis for the development of intelligence and automation in computer-aided design (CAD). Through the incorporation of a strategically devised exponential penalty function, we have transformed a quadratic programming (QP) under both equality and inequality constraints of SVM into an unconstrained optimization paradigm. This refinement not only simplifies the computational framework but also extends the algorithm's adaptability, making the I-BFGS scheme an effective choice for SVM applications, notably in CAD, where it can significantly expedite the design process and curtail computational burden. Experimental results show that the proposed I-BFGS scheme exhibits excellent performance when combined with radial basis function (RBF) kernel, and great robustness and consistency when using sigmoid kernel. Comparative analyses with Pegasos, LIBSVM and logistics regression further accentuate the I-BFGS scheme's distinctive benefits in terms of efficiency and precision, highlighting its potential for practical applications. In essence, this research unveils a new strategy for SVM optimization, paving the way for innovative applications in diverse fields, including CAD, by enabling more streamlined and accurate design methodologies.

Keywords: SVM, I-BFGS, CAD, Pegasos, QP
DOI: https://doi.org/10.14733/cadaps.2024.610-624

1 INTRODUCTION

Support vector machines (SVMs) have emerged as versatile tools across a spectrum of scientific disciplines, catalyzing groundbreaking advancements and substantially influencing research trajectories. Their versatility is evident in bioinformatics for protein classification and genomics studies [1], in image processing for object detection and facial recognition [2], and in neuroscience for brain imaging and neural activity prediction [3, 4]. They have also significantly impacted natural language processing [5] and cybersecurity [6], underlining their adaptability and indispensability in contemporary scientific investigations.

In computer-aided design (CAD), the ability to quickly and effectively attain high-quality solutions is pivotal. Given the inherent complexity and multifaceted nature of design problems, achieving globally optimal solutions is often impractical and computationally intensive. Hence, methods that can provide reliable and high-quality optimum solutions are of immense value [7, 8]. SVMs, with their adaptability and robustness, can be invaluable in influencing design trajectories, aiding in model generation, shape recognition, and the optimization of design parameters, thereby enhancing the overall efficiency and efficacy of the design process [9].

One of the foundational advantages of SVMs is their formulation of their dual form as quadratic programming (QP) under both equality and inequality constraints, which offers pathways to more efficient and trustworthy computational solutions [10]. However, compared with unconstrained problems, the constrained QP problems are significantly more complex, especially for large scale datasets, which is precisely the challenge faced in SVMs. In response to this problem, many efforts and contributions have been made in different directions. On the one hand, some researchers endeavor to find an alternative way to avoid this complex form, instead of optimizing the original SVM problem directly. Notably, Shalev-Shwartz *et al.* present the Pegasos algorithm, which harnesses subgradient methods and stochastic gradient descent (SGD) to achieve solutions, demonstrating impressive performance and ensuring fast convergence even for voluminous datasets [11]. Their main breakthrough lies in solving the problem that the hinge loss function in the original SVM problem has non-differentiable points and combining this with an efficient algorithm SGD, which greatly improves the computing speed, even if loses a certain amount of accuracy and stability compared to gradient descent (GD) or Batch-GD algorithm [12, 13]. Similarly, Chapelle designs an alternative approach by replacing the hinge loss with a smooth loss function, thus also making both GD and SGD feasible [14].

On the other hand, historically, due to the existence of non-differentiable points in hinge loss, optimizing the original problem is relatively difficult and requires a lot of calculations, therefore early work still mainly focuses on the optimization of the dual problem. By iteratively incorporating the constraints with the greatest deviations from the required conditions into the model, Joachims' cutting-plane algorithm effectively solves this complex problem but typically requires a fine control in terms of working set size and constraint scale to reach a trade-off between efficiency and accuracy [15]. Platt's sequential minimal optimization (SMO) method strategically breaks down the dual QP problem into smaller, more manageable sub-problems, achieving precise and efficient computation, and has become the most common solution for SVMs today [16]. The well-known commercial packages LIBSVM and SVM-Light are based on improvements to this algorithm [17, 18]. However, the SMO algorithm still has certain limitations, such as sensitivity to the choice of kernel function, and difficulty in handling large datasets or sparse data. Therefore, the exploration of new methods that can address these challenges is still ongoing.

In light of these limitations, there has been a growing interest in revisiting conventional optimization strategies such as GD, SGD, Newton's method, and quasi-Newton methods, which are underpinned by solid theoretical foundations and are universally applicable [19, 20]. Especially for quasi-Newton methods such as BFGS, they have been further enhanced with some advanced features such as memory optimization [21], momentum terms [22] and adaptive learning rates [23]. However, these algorithms often meet with obstacles in this direction for SVMs as they cannot be directly applied to constrained optimization problems. They typically necessitate integrating other methods first such as the interior-point method to transform the constraints [24]. For example, Suykens and Vandewalle present the least squares support vector machine (LSSVM), which changes the constrained optimization problems to linear equations, and provides the possibility for the application of the BFGS algorithm in this direction [25]. Then, Chen *et al.* design the sparse accelerated limited memory BFGS algorithm specifically for training the LSSVM classifier, which eliminates redundancy and noise in large-scale, high-dimensional data by exploiting sparsity, and maintains good performance and efficiency when dealing with large sparse datasets.

In this paper, we ingeniously transform the standard quadratic programming problem with equality and inequality constraints in the dual form of SVMs into an unconstrained convex optimization problem by employ-

ing an exponential penalty function [26]. Subsequently, we employ the improved Broyden-Fletcher-Goldfarb-Shanno (I-BFGS) scheme to train the SVM classifier. BFGS, an esteemed iterative optimization technique, is particularly adept at handling unconstrained nonlinear minimization problems [27, 28, 29, 30]. Evolving from Newton's method, BFGS is characterized as a quasi-Newton or a second-order convergence technique. Its hallmark is the iterative approximation of the inverse Hessian matrix, obviating the need for its direct inversion, a requirement in the traditional Newton's method, which confers a distinct computational edge [31]. In convex optimization problems, it can quickly approach the global optimal value with a small number of iterations. This scheme is particularly beneficial for CAD applications, where precise and computationally efficient outcomes are crucial, enabling the development of advanced and optimized design models [9].

This article is structured as follows: Section II delineates the datasets and their preprocessing methodologies. Section III elucidates the framework of this scheme and its derivation. Section IV presents the experimental results and a convergence analysis. Section V encapsulates our conclusions. A block diagram encapsulating the core thrust of this paper is depicted in Fig. 1.

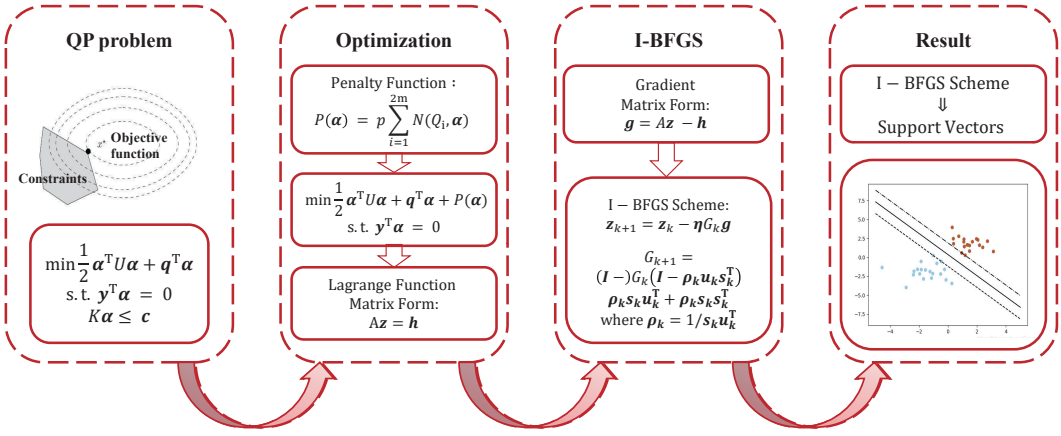

Figure 1: Block diagram of this paper.

Our principal contributions include:

1. We introduce an improved BFGS scheme for SVMs, which innovatively integrates an exponential penalty function to eliminate the equality and inequality constraints complicating SVM solutions. Furthermore, an approximation matrix supplants the inverse of the Hessian matrix in the BFGS, further mitigating computational demands.

2. Experimental evaluations across diverse datasets reveal that the SVM classification efficacy of our proposed I-BFGS approach either matches or surpasses the compared methods, especially when combined with sigmoid kernel, showing robustness to kernel selection.

3. The potential foundational approach for further integration within CAD systems, contributing to advancements in computer-aided design and its applications by providing fast and reliable approximate global optimum solutions to complex design problems.

2 DATASETS

To rigorously assess the efficacy of the proposed methodology, we employ six publicly accessible, structured classification datasets in our experimental evaluations. All these datasets are geared towards binary classifica-

tion, with class labels designated as -1 and 1. Each dataset possesses a distinct number of features, thereby offering a varied dimensional landscape for comprehensive evaluation. The datasets and their respective dimensions are as follows:

1. Wine [32]: 13 features.
2. Iris-setosa [33]: 5 features.
3. Miners or Rocks [34]: 60 features.
4. Breast Cancer [35]: 30 features.
5. Age-Related Condition [36]: 56 features.
6. Smoke [37]: 25 features.

To ensure a holistic evaluation across balanced datasets, we meticulously adjust three datasets ("Breast Cancer", "Age-Related Condition", "Smoke") through undersampling. This maneuver ensures a near-equivalent number of instances across both classes. Conversely, the other trio of datasets ("Wine", "Iris-setosa", "Miners or Rocks") are preserved in their inherent imbalanced configurations, reflecting pronounced disparities in class distribution. The "Wine" dataset, originally multi-class, was transformed into a binary dataset by amalgamating wines from categories 2 and 3. The "Iris-setosa" dataset was derived by isolating the "setosa" class from the comprehensive "Iris" dataset, labelling it as 1. Simultaneously, the remaining non-"setosa" categories were consolidated into a singular class, labelled as -1. Crucially, all datasets are devoid of missing values and have undergone normalization. This refinement not only expedites computational processes but also curtails potential numerical instabilities, ensuring the optimization process remains unaffected [38].

3 FRAMEWORK

This section first establishes the original SVM model and its transformation into an unconstrained optimization problem, and then solves it with the I-BFGS scheme to obtain the predicted value of SVM. For clarity, Algorithm 1 summarizes this process.

3.1 SVM Model

The basic idea of SVM is to classify data by constructing a hyperplane, or linear decision function, which is determined by a coefficients matrix $\boldsymbol{w}$ and a deviation coefficient b. The function takes the form

$$f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b, \tag{1}$$

where $\boldsymbol{w} \in \mathbb{R}^{n\times 1}$, b is a scalar and $\boldsymbol{x} \in \mathbb{R}^{m\times n}$ is the feature matrix. For example, dataset D can be expressed as

$$D = \begin{bmatrix} (x_1^{\mathrm{T}}, y_1) \\ (x_2^{\mathrm{T}}, y_2) \\ \vdots \\ (x_m^{\mathrm{T}}, y_m) \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} & y_1 \\ x_{21} & x_{22} & \cdots & x_{2n} & y_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & y_m \end{bmatrix}, \tag{2}$$

where feature vector $\boldsymbol{x}_i \in \mathbb{R}^{n\times 1}$ and feature label $y_i \in \{1, -1\}$. The positive or negative of the value of $f(\boldsymbol{x})$ decides which class the unlabeled data $\boldsymbol{x}$ belongs to.

To find the hyperplane that has maximum margin, that is, to find $\boldsymbol{w}$ and b that satisfy certain constraints, and it is equivalent to minimizing the square paradigm of margin. Therefore, solving the decision function can be transformed into a QP problem, which is also the basic form of SVM

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{3}$$

$$\begin{aligned} &\text{s.t.} \quad y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \geq 1 \\ &\text{with } i = 1, 2, \ldots, m. \end{aligned} \tag{4}$$

To avoid several training data from excessively influencing the dividing hyperplane, constraint (4) can be adjusted slightly by introducing a relaxation variable $\boldsymbol{\xi}_i$ to construct "soft-margin" SVM

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}\boldsymbol{\xi}_i \tag{5}$$

$$\begin{aligned} &\text{s.t.} \quad y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \geq 1 - \xi_i \\ &\qquad \xi_i \geq 0 \\ &\text{with } i = 1, 2, \ldots, m. \end{aligned} \tag{6}$$

Defining two Lagrangian multiplier α and μ, QP problem (5) and (6) can be transformed as a Lagrangian function, i.e., a min-max problem

$$\begin{aligned} L \equiv \min_{\boldsymbol{w},b}\max_{\substack{\alpha_i\geq 0\\ \mu_i\geq 0}} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\boldsymbol{\xi}_i \\ + \sum_{i=1}^{m}\boldsymbol{\alpha}_i(1 - \boldsymbol{\xi}_i - y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b)) - \sum_{i=1}^{m}\boldsymbol{\mu}_i\boldsymbol{\xi}_i. \end{aligned} \tag{7}$$

Take the partial derivative with respect to $\boldsymbol{w}$, b and $\boldsymbol{\xi}_i$ be 0, and we can get

$$\boldsymbol{w} = \sum_{i=1}^{m}\alpha_i y_i \boldsymbol{x}_i. \tag{8}$$

$$0 = \sum_{i=1}^{m}\alpha_i y_i. \tag{9}$$

$$C = \alpha_i + \mu_i. \tag{10}$$

In this regard, dual problem of (5) can be obtained by substituting (8)-(10) into (7)

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j \tag{11}$$

$$\begin{aligned} &\text{s.t.} \sum_{i=1}^{m}\alpha_i y_i = 0 \\ &0 \leq \alpha_i \leq C \\ &\text{with } i = 1, 2, \ldots, m. \end{aligned} \tag{12}$$

Furthermore, kernel function can be applied in formula (11) for the case of linear indivisibility

$$\min_{\boldsymbol{\alpha}} \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j K(\boldsymbol{x}_i,\boldsymbol{x}_j) - \sum_{i=1}^{m}\alpha_i \tag{13}$$

$$\begin{aligned} &\text{s.t.}\sum_{i=1}^{m}\alpha_i y_i = 0 \\ &0 \le \alpha_i \le C \\ &\text{with } i = 1, 2, \ldots, m, \end{aligned} \tag{14}$$

where for RBF kernel:

$$K(\boldsymbol{x}_i,\boldsymbol{x}_j) = \exp\left(-\gamma\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2\right), \tag{15}$$

and for sigmoid kernel:

$$K(\boldsymbol{x}_i,\boldsymbol{x}_j) = \tanh\left(\beta\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j + \theta\right). \tag{16}$$

3.2 Standard QP Problem

By defining

$$\boldsymbol{\alpha} = \begin{bmatrix}\alpha_1\\ \alpha_2\\ \vdots\\ \alpha_m\end{bmatrix}, \tag{17}$$

$$\boldsymbol{q} = \begin{bmatrix}-1\\ -1\\ \vdots\\ -1\end{bmatrix}, \tag{18}$$

$$\boldsymbol{y} = \begin{bmatrix}y_1\\ y_2\\ \vdots\\ y_m\end{bmatrix}, \tag{19}$$

$$\boldsymbol{x} = \begin{bmatrix}x_1\\ x_2\\ \vdots\\ x_m\end{bmatrix}, \tag{20}$$

$$U = \boldsymbol{y}\boldsymbol{y}^{\mathrm{T}} \odot \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}. \tag{21}$$

Equations (13)-(14) can be expressed as

$$\min \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}U\boldsymbol{\alpha} + \boldsymbol{q}^{\mathrm{T}}\boldsymbol{\alpha} \tag{22}$$

$$\begin{aligned} \text{s.t.} \quad & \boldsymbol{y}^{\mathrm{T}}\boldsymbol{\alpha} = 0 \\ & 0 \le \boldsymbol{\alpha} \le C. \end{aligned} \tag{23}$$

It can be expressed as a standard QP problem form

$$\min \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}} U \boldsymbol{\alpha} + \boldsymbol{q}^{\mathrm{T}}\boldsymbol{\alpha} \tag{24}$$

$$\text{s.t.} \quad \boldsymbol{y}^{\mathrm{T}}\boldsymbol{\alpha} = 0 \tag{25}$$

$$K\boldsymbol{\alpha} \le \boldsymbol{c}, \tag{26}$$

where

$$K = \begin{bmatrix} I \\ -I \end{bmatrix} \in \mathbb{R}^{2m \times m}, \tag{27}$$

$$\boldsymbol{c} = \begin{bmatrix} C \\ 0 \end{bmatrix} \in \mathbb{R}^{2m \times 1}, \tag{28}$$

and I is identity matrix.

3.3 Improved-BFGS Scheme Design

Quasi Newton methods are often used to solve unconstrained optimization problems. In this paper, a penalty function is introduced to convert the standard QP form of SVM into an unconstrained problem. First, the penalty part is

$$P(\boldsymbol{\alpha}) = p \sum_{i=1}^{2m} N(Q_i, \boldsymbol{\alpha}) \tag{29}$$

$$\text{with} \quad Q_i = \boldsymbol{c}_i - K_i\boldsymbol{\alpha} \tag{30}$$

$$N(\cdot) = e^{-\sigma Q_i} \tag{31}$$

where $\sigma > 0$, $p > 0$. Then, the above penalty function is used to replace the inequality constraints (27), and it satisfies

$$\begin{cases} P(\boldsymbol{\alpha}) \approx 0, \text{if } Q_i \ge 0 \\ P(\boldsymbol{\alpha}) \gg 0, \text{if } Q_i < 0 \end{cases}. \tag{32}$$

When the inequality constraint (27) is met, $P(\boldsymbol{\alpha})$ yields a value close to zero, indicating a small result. However, if the inequality constraint is not met, $P(\boldsymbol{\alpha})$ produces a large value as a substantial penalty. Therefore, the inequality (27) constraint can be substituted by the penalty function, and the standard QP problem (25)-(27) can be rewritten as

$$\min \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}} U \boldsymbol{\alpha} + \boldsymbol{q}^{\mathrm{T}}\boldsymbol{\alpha} + P(\boldsymbol{\alpha}) \tag{33}$$

$$\text{s.t.} \quad \boldsymbol{y}^{\mathrm{T}}\boldsymbol{\alpha} = 0. \tag{34}$$

By utilizing the Lagrange multiplier method, we can obtain

$$L(\boldsymbol{\alpha}, \boldsymbol{\lambda}) = \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}} U \boldsymbol{\alpha} + \boldsymbol{q}^{\mathrm{T}}\boldsymbol{\alpha} + P(\boldsymbol{\alpha}) + \boldsymbol{\lambda}^{\mathrm{T}}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{\alpha}. \tag{35}$$

Subsequently, take the partial derivative with respect to $\boldsymbol{\alpha}$ and $\boldsymbol{\lambda}$, and one has

$$\begin{cases} \dfrac{\partial L(\boldsymbol{\alpha},\boldsymbol{\lambda})}{\partial \boldsymbol{\alpha}} = U\boldsymbol{\alpha} + \boldsymbol{q} + \boldsymbol{y}\boldsymbol{\lambda} + p\sigma \sum_{i=1}^{m} (e^{-\sigma Q_i} \cdot K_i^{\mathrm{T}}) = 0 \\ \dfrac{\partial L(\boldsymbol{\alpha},\boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}} = \boldsymbol{y}^{\mathrm{T}}\boldsymbol{\alpha} = 0 \end{cases}. \tag{36}$$

Define

$$A = \begin{bmatrix} U & \boldsymbol{y} \\ \boldsymbol{y}^{\mathrm{T}} & 0 \end{bmatrix} \in \mathbb{R}^{(m+1)\times(m+1)}, \tag{37}$$

$$\boldsymbol{z} = \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\lambda} \end{bmatrix} \in \mathbb{R}^{(m+1)\times 1}, \tag{38}$$

$$\boldsymbol{h} = \begin{bmatrix} -\boldsymbol{q} - p\sigma \sum_{i=1}^{2m} (e^{-\sigma Q_i} \cdot K_i^{\mathrm{T}}) \\ 0 \end{bmatrix} \in \mathbb{R}^{(m+1)\times(m+1)}. \tag{39}$$

In this sense, equation (36) can be rewritten as a compact form:

$$\boldsymbol{g} = A\boldsymbol{z} - \boldsymbol{h}. \tag{40}$$

Newton's method, a second-order optimization algorithm, builds on the principles of gradient descent by incorporating second-order derivative information via the Hessian matrix. By accounting for the function's curvature and implementing a local quadratic approximation, this method refines the search direction utilizing both the Hessian matrix and the gradient. Although this technique offers a more nuanced and precise approach compared to basic gradient descent, its real-world deployment is often stymied by the hefty computational demands of calculating, storing, and inverting the Hessian matrix, especially for expansive problems. Moreover, in the context of non-convex problems, a non-positive definite Hessian matrix can compromise the descent direction of the search.

Given these challenges, the BFGS algorithm emerges as a viable solution. As a Quasi-Newton method, it mimics the prowess of Newton's method without necessitating the direct computation and storage of the Hessian matrix. Instead, the BFGS algorithm leverages an approximation matrix, B, typically initialized as an identity matrix, to estimate the inverse of the Hessian matrix. This matrix is iteratively updated using data from both the current and preceding gradient and position. As a result, the BFGS algorithm significantly reduces computational complexity compared to Newton's method. Moreover, the continuously updated matrix B remains positive-definite, ensuring that the search direction consistently aligns with the descent direction. Specifically, the update rule for the conventional BFGS is

$$\boldsymbol{z}_{k+1} = \boldsymbol{z}_k - \boldsymbol{\eta}_k B_k^{-1} \boldsymbol{g}_k, \tag{41}$$

where $\boldsymbol{\eta}_k$ is the step size. By denoting $\boldsymbol{s}_k = \boldsymbol{z}_k - \boldsymbol{z}_{k-1}$ and $\boldsymbol{u}_k = \boldsymbol{g}_k - \boldsymbol{g}_{k-1}$ the BFGS can approximates the inverse of the Hessian matrix by the solution of secant equation $B_k \boldsymbol{s}_k = \boldsymbol{u}_k$. Then, the update rule of B_k can be defined as

$$B_{k+1} = B_k + \frac{\boldsymbol{u}_k \boldsymbol{u}_k^{\mathrm{T}}}{\boldsymbol{u}_k^{\mathrm{T}} \boldsymbol{s}_k} - \frac{B_k \boldsymbol{s}_k \boldsymbol{s}_k^{\mathrm{T}} B_k}{\boldsymbol{s}_k^{\mathrm{T}} B_k \boldsymbol{s}_k}. \tag{42}$$

Algorithm 1 I-BFGS for SVM Optimization

1: Input: Training data set $D = (\boldsymbol{x}_i, y_i), i = 1, ..., n$, where $\boldsymbol{x}_i$ is the input vector and y_i is the target value; parameters for penalty function C, p and σ; tolerance ε for convergence, max iteration *max_ iter*, and kernel function K;

2: Output: $\boldsymbol{\alpha}$ and λ, bias b, prediction value $\boldsymbol{y_{pred}}$;

Initialize:

3: Set objective function L of the transformed unconstrained SVM dual problem;

4: Choose initial $\boldsymbol{\alpha}_0$, λ_0 and set $\boldsymbol{z}_0 = [\boldsymbol{\alpha}_0; \lambda_0]$;

5: Set initial Hessian matrix approximation G_0 as an identity matrix;

6: Choose initial step size η_0;

7: Set $k = 0$;

Repeat:

8: Compute the designed exponential penalty function $P(\boldsymbol{\alpha})$ according to formulas (29)-(32);

9: Compute A using chosen kernel K;

10: Compute the gradient $\boldsymbol{g}_k = A \cdot \boldsymbol{z}_k - \boldsymbol{h}_k$ using A and $P(\boldsymbol{\alpha})$;

11: **if** $\|\boldsymbol{g}_k\| < \varepsilon$ **then**

12: **exit loop** {convergence criteria met};

13: **end**

14: Calculate the direction of descent $\boldsymbol{d}_k = -G_k \cdot \boldsymbol{g}_k$;

15: Perform a line search to find the step size η_k that minimizes the objective function along the direction $\boldsymbol{d}_k$;

16: Update the state $\boldsymbol{z}_{k+1} = \boldsymbol{z}_k - \boldsymbol{\eta}_k B_k^{-1} \boldsymbol{g}_k$, and $\boldsymbol{s}_k = \boldsymbol{z}_k - \boldsymbol{z}_{k-1}, \boldsymbol{u}_k = \boldsymbol{g}_k - \boldsymbol{g}_{k-1}, \boldsymbol{\rho}_k = \boldsymbol{u}_k^{\mathrm{T}} \boldsymbol{s}_k$;

17: Update the Hessian matrix approximation using the I-BFGS update rule: $G_{k+1} = (I - \boldsymbol{\rho}_k \boldsymbol{s}_k \boldsymbol{u}_k^{\mathrm{T}}) G_k (I - \boldsymbol{\rho}_k \boldsymbol{u}_k \boldsymbol{s}_k^{\mathrm{T}}) + \boldsymbol{\rho}_k \boldsymbol{s}_k \boldsymbol{s}_k^{\mathrm{T}}$;

18: $k = k + 1$;

Until convergence criteria are met

19: Choose support vectors whose α_i between 0 and C;

20: Compute bias $b = \frac{1}{N_s} \sum_{i=1}^{N_s} \left(y_i - \sum_{j=1}^{N_s} \alpha_j y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \right)$ for SVM by using y_i, α_i of support vectors and chosen kernel;

21: Compute prediction value $y_{pred,i} = \operatorname{sgn}\left(\sum_{i=1}^{N_s} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b \right)$;

22: **Return** $\boldsymbol{\alpha}$, λ, b, $\boldsymbol{y_{pred}}$.

Even so, obtaining the search direction still necessitates the computation of the inverse of the approximation matrix B_k, which may impose high computational costs and cause numerical instability due to the near singularity. This problem can be solved by employing the Sherman-Morrison formula [39] to directly update the inverse of B_k. Then, the update rule of B_k can be replaced by

$$G_{k+1} = (I - \boldsymbol{\rho}_k \boldsymbol{s}_k \boldsymbol{u}_k^{\mathrm{T}}) G_k (I - \boldsymbol{\rho}_k \boldsymbol{u}_k \boldsymbol{s}_k^{\mathrm{T}}) + \boldsymbol{\rho}_k \boldsymbol{s}_k \boldsymbol{s}_k^{\mathrm{T}}, \tag{43}$$

where

$$\boldsymbol{\rho_k} = \boldsymbol{u}_k^{\mathrm{T}} \boldsymbol{s}_k. \tag{44}$$

Finally, the improved BFGS solver for SVM has been constructed, with

$$\boldsymbol{z}_{k+1} = \boldsymbol{z}_k - \boldsymbol{\eta}_k G_k \boldsymbol{g}_k. \tag{45}$$

4 Experiment

This section presents a comprehensive evaluation of the I-BFGS scheme for SVMs, validating its effectiveness compared with the Pegasos SVM [11], LIBSVM [18] and logistic regression [40]. The experiments are conducted on six publicly available, small, structured binary classification datasets.

4.1 Experimental Setup

The datasets employed in our evaluations are structured for binary classification, where class labels are designated as -1 and 1. For enhanced computational efficiency and to mitigate potential numerical instabilities, all datasets underwent normalization, thereby optimizing the subsequent analytical processes. Rigorous preprocessing ensured each dataset was devoid of missing values. Table 1 outlines the respective sizes of these datasets, and a five-fold cross-validation method is employed to ensure the robustness of the evaluation and avoid overfitting for each scheme.

4.2 Experimental Results

As shown in Fig. 2, it is evident that the I-BFGS scheme, regardless of the kernel functions employed, exhibits steadfast convergence across all datasets. The comprehensive performance comparison is presented in Table 1, where I-BFGS, Pegasos, LIBSVM, and Logistic Regression are evaluated side by side across various datasets and kernel functions. Among them, LIBSVM and logistic regression are used as baselines for auxiliary comparison as they are the most common and mature algorithms in machine learning. We delve into these observed patterns in the ensuing sections:

Performance with RBF Kernel When using the RBF kernel, it can be observed that each of the three SVM methods excels over the other two at certain times, with not much difference in performance, indicating that they have comparable levels of performance and are suited to their respective data distributions. For instance, I-BFGS achieves the best performance on "Wine", Pegasos on "Miners or Rocks", and LIBSVM on "Age-Related Condition".

Performance with Sigmoid Kernel With the sigmoid kernel, the I-BFGS scheme and LIBSVM demonstrate consistently great performance across multiple datasets, which implies excellent generalization ability. Even occasionally not performing as well as them with the RBF kernel, their variability is not significant, suggesting a relative insensitivity to the choice of kernel. However, LIBSVM exhibits a substantial performance decline on the simplest dataset, "Iris-setosa", possibly due to the data distribution being unsuitable for this type of algorithm. In contrast, Pegasos performs significantly worse with the sigmoid kernel compared to other algorithms, with much greater performance variability when compared to its own performance with the RBF kernel. Specifically, on the "Iris-setosa" and "Miners or Rocks" datasets, the Pegasos algorithm seems completely unable to handle these types of data distribution and gets an accuracy of only 32.90% and 57.82% respectively. These contrasts in Pegasos emphasize the need for careful kernel selection. Moreover, to a certain extent, the performance comparison on the "Iris-setosa" dataset suggests that the I-BFGS scheme with the sigmoid kernel shows less dependency on parameter selection compared to the other two SVM algorithms.

Logistic Regression Benchmark Logistic Regression, often used as a baseline in classification tasks, shows its robustness with stable scores across all metrics for the different datasets. In most cases, its performance is

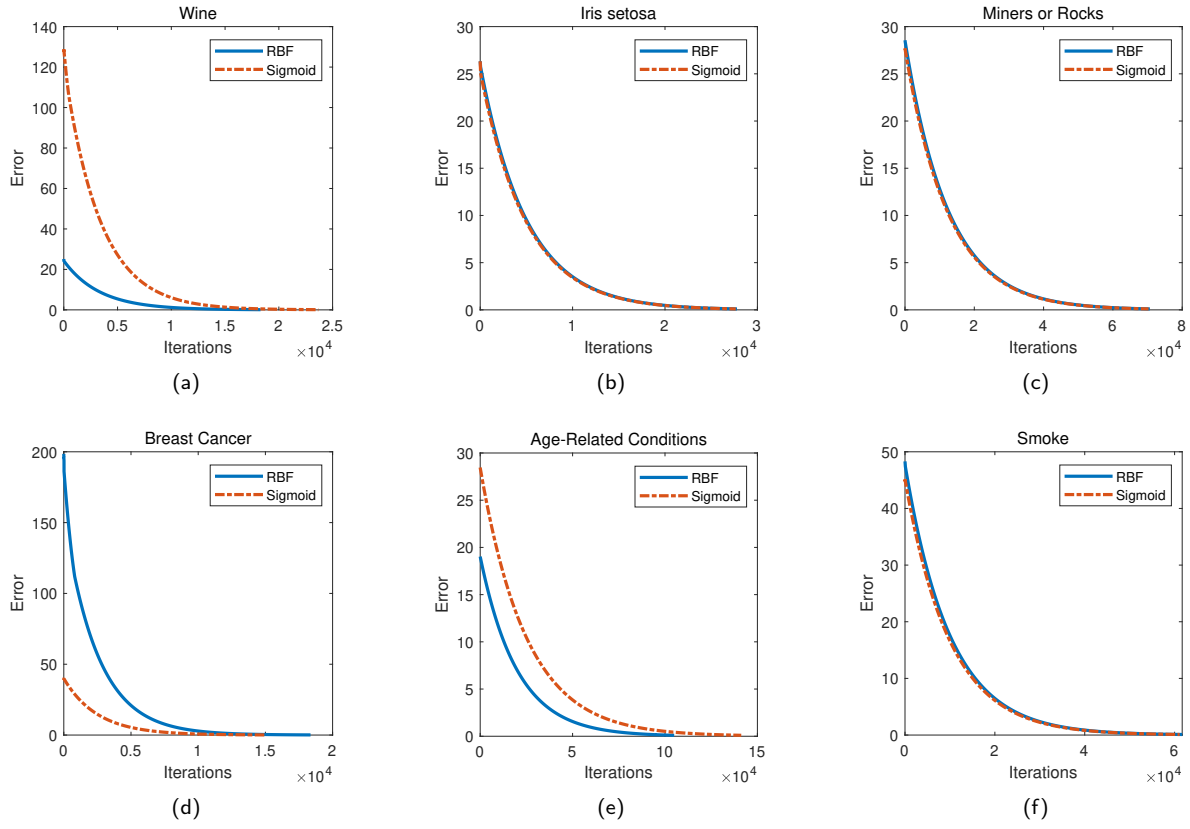

Figure 2: Gradient Convergence Curve of (a) Wine Dataset. (b) Iris Dataset. (c) Miners or Rocks Dataset. (d) Breast Cancer Dataset. (e) Age-Related Conditions Dataset. (f) Smoke Dataset.

not as superior as SVM methods. However, it significantly outperforms all SVM-based methods on the "Breast Cancer" dataset, with 97.17% accuracy, 97.12% recall and 96.86% F1-score, demonstrating the enduring value of logistic regression in binary classification tasks, particularly when the decision boundary is linear or near-linear. This result aligns with our predictions because most of the datasets faced by current machine learning exhibit nonlinear distributions.

4.3 Conclusive Insights

In essence, the I-BFGS scheme, especially when combined with the RBF kernel, consistently delivers robust performance across a spectrum of datasets. Conversely, Pegasos exhibits more performance variability, with its efficacy seemingly tethered to kernel selection, particularly underwhelming with the sigmoid kernel. LIBSVM as a business library with well-rounded improvement and mature optimization techniques, remains a strong contender and has achieved first place in most cases, slightly better than other algorithms. It can demonstrate the potential value of the innovation of the I-BFGS scheme by providing a comparison with this industry benchmark. Cumulatively, these findings accentuate the proposed I-BFGS scheme as a promising avenue for SVM optimization.

Table 1: Comparative experiment results with different methods among various datasets

Dataset	Size	Kernel	Method	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
Wine	179	RBF	I-BFGS	98.89	100	98.00	98.95
			Pegasos	98.33	96.90	100	98.39
			LIBSVM	97.18	94.64	97.45	95.82
		Sigmoid	I-BFGS	89.35	98.33	72.67	81.29
			Pegasos	73.13	55.21	100	70.53
			LIBSVM	96.63	93.44	97.01	94.95
		Logistic Regression		96.62	95.21	95.52	95.21
Iris-setosa	149	RBF	I-BFGS	100	100	100	100
			Pegasos	100	100	100	100
			LIBSVM	100	100	100	100
		Sigmoid	I-BFGS	100	100	100	100
			Pegasos	32.90	32.90	100	49.34
			LIBSVM	84.55	53.33	53.33	53.33
		Logistic Regression		100	100	100	100
Miners or Rocks	208	RBF	I-BFGS	83.70	91.79	75.36	81.84
			Pegasos	87.03	91.67	77.61	83.94
			LIBSVM	79.51	79.58	76.35	77.44
		Sigmoid	I-BFGS	67.96	68.80	64.71	64.79
			Pegasos	57.84	53.62	67.58	59.30
			LIBSVM	78.18	77.73	75.38	76.04
		Logistic Regression		74.02	72.48	72.37	71.93
Breast Cancer	424	RBF	I-BFGS	94.34	94.77	93.47	94.04
			Pegasos	95.99	96.93	94.68	95.78
			LIBSVM	95.05	94.59	95.54	94.85
		Sigmoid	I-BFGS	92.92	95.67	90.27	92.74
			Pegasos	90.10	90.21	90.04	89.99
			LIBSVM	94.99	94.38	95.44	94.72
		Logistic Regression		97.17	96.81	97.12	96.86
Age-Related Condition	204	RBF	I-BFGS	73.01	68.95	84.52	75.26
			Pegasos	76.49	78.28	73.58	75.18
			LIBSVM	78.93	78.86	80.88	79.41
		Sigmoid	I-BFGS	71.60	79.93	64.24	68.04
			Pegasos	64.71	66.26	60.01	62.67
			LIBSVM	78.69	78.48	80.48	79.11
		Logistic Regression		77.95	79.59	77.27	77.91
Smoke	502	RBF	I-BFGS	74.10	68.56	89.29	77.41
			Pegasos	76.08	69.03	94.88	79.84
			LIBSVM	72.30	69.12	80.87	74.24
		Sigmoid	I-BFGS	74.50	68.29	91.10	78.05
			Pegasos	65.34	61.11	84.85	71.00
			LIBSVM	73.22	68.53	86.26	76.05
		Logistic Regression		73.69	69.10	85.89	76.52

5 Conclusion

Our research has presented an improved Broyden-Fletcher-Goldfarb-Shanno (I-BFGS) scheme for SVMs, converting it from a quadratic programming problem under equality and inequality constraints to an unconstrained one by introducing an exponential penalty function. The I-BFGS scheme provides efficient approximations, essential for generating intricate models and optimizing designs within practical timeframes. Compared to

Pegasos, LIBSVM, and logistic regression, I-BFGS with RBF kernel demonstrates superior performance, and when combined with sigmoid kernel, it shows consistency across datasets, implicating the insensitivity to kernel selection. Additionally, it is significantly more suitable than logistic regression for non-centralized datasets, which are more common in the field of machine learning. The proposed I-BFGS scheme streamlines CAD modelling by offering efficient design and predictive analytics, showing promise for tasks such as automated defect detection and structural optimization, and is poised to significantly enhance design and manufacturing with more precise and feasible solutions.

REFERENCES

[1] Liu, Y.; Zeng, X.; He, Z.; Zou, Q.: Inferring microrna-disease associations by random walk on a heterogeneous network with multiple data sources. IEEE/ACM Trans. Comput. Biol. Bioinform., 14(4), 905–915, 2017. http://doi.org/10.1109/TCBB.2016.2550432.

[2] Yang, J.; Zhang, D.; Frangi, A.F.; Yang, J.Y.: Two-dimensional pca: a new approach to appearance-based face representation and recognition. IEEE Trans. Pattern Anal. Mach. Intell., 26(1), 131–137, 2004. http://doi.org/10.1109/TPAMI.2004.1261097.

[3] Orrú, G.; Pettersson-Yeo, W.; Marquand, A.F.; Sartori, G.; Mechelli, A.: Using support vector machine to identify imaging biomarkers of neurological and psychiatric disease: a critical review. Neurosci. Biobehav. Rev., 36(4), 1140–1152, 2012. http://doi.org/10.1016/j.neubiorev.2012.01.004.

[4] Pereira, F.; Mitchell, T.; Botvinick, M.: Machine learning classifiers and fmri: a tutorial overview. Neuroimage, 45(1), S199–S209, 2009. http://doi.org/10.1016/j.neuroimage.2008.11.007.

[5] Joachims, T.: Learning to Classify Text Using Support Vector Machines: Methods, Theory and Algorithms. Kluwer Academic Publishers, 2002. http://doi.org/10.1007/978-1-4615-0907-3.

[6] Eskin, E.; Arnold, A.; Prerau, M.; Portnoy, L.; Stolfo, S.: A geometric framework for unsupervised anomaly detection. In Applications of Data Mining in Computer Security, 77–101. Springer, Boston, MA, 2002. http://doi.org/10.1007/978-1-4615-0953-0_4.

[7] Agarwal, D.; Robinson, T.; Armstrong, C.: A cad based framework for optimizing performance while ensuring assembly fit. In S. Wang; M. Price; M. Lim; Y. Jin; Y. Luo; R. Chen, eds., Recent Advances in Intelligent Manufacturing, vol. 923 of Communications in Computer and Information Science. Springer, Singapore, 2018. http://doi.org/10.1007/978-981-13-2396-6_7.

[8] Agarwal, D.; Robinson, T.; Armstrong, C.: Cad-based adjoint optimization using other components in a cad model assembly as constraints. Computer-Aided Design and Applications, 20(4), 749–762, 2022. ISSN 1686-4360. http://doi.org/10.14733/cadaps.2023.749-762.

[9] Shah, J.J.; Mäntylä, M.: Parametric and feature-based CAD/CAM: concepts, techniques, and applications. John Wiley & Sons, Inc., 1995.

[10] Boyd, S.; Vandenberghe, L.: Convex Optimization. Cambridge University Press, 2004. http://doi.org/10.1017/CBO9780511804441.

[11] Shalev-Shwartz, S.; Singer, Y.; Srebro, N.: Pegasos: Primal estimated sub-gradient solver for svm. In Proceedings of the 24th International Conference on Machine Learning, 807–814, 2007. http://doi.org/10.1145/1273496.1273598.

[12] Amari, S.: Natural gradient works efficiently in learning. Neural Computation, 10, 251–276, 1998. http://doi.org/10.1162/089976698300017746.

[13] Bottou, L.: Online algorithms and stochastic approximations. In D. Saad, ed., Online Learning and Neural Networks. Cambridge University Press, Cambridge, UK, 1998. http://doi.org/10.1017/CBO9780511569920.003.

[14] Chapelle, O.: Training a support vector machine in the primal. Neural Computation, 19(5), 1155–1178, 2007. http://doi.org/10.1162/neco.2007.19.5.1155.

[15] Joachims, T.: Optimizing search engines using clickthrough data. In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, 133–142. Edmonton, Alberta, Canada, 2002. http://doi.org/10.1145/775047.775067.

[16] Platt, J.: Sequential minimal optimization: A fast algorithm for training support vector machines. Tech. rep., Microsoft Research, 1998.

[17] Joachims, T.: Making large-scale svm learning practical. In B. Schlkopf; C.J.C. Burges; A.J. Smola, eds., Advances in Kernel Methods. MIT Press, 1999. http://doi.org/10.17877/DE290R-5098.

[18] Chang, C.C.; Lin, C.J.: Libsvm: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2(3), 27:1–27:27, 2011. http://doi.org/10.1145/1961189.1961199.

[19] Ruder, S.: An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747, 2016. http://doi.org/10.48550/arXiv.1609.04747.

[20] Bottou, L.; Murata, N.: Stochastic approximations and efficient learning. In M. Arbib, ed., The Handbook of Brain Theory and Neural Networks. The MIT Press, Cambridge, MA, 2 ed., 2002. http://leon.bottou.org/papers/bottou-murata-2002.

[21] Moritz, P.; Nishihara, R.; Jordan, M.: A linearly-convergent stochastic l-bfgs algorithm. In Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, vol. 51 of Proceedings of Machine Learning Research, 249–258, 2016. https://proceedings.mlr.press/v51/moritz16.html.

[22] Bollapragada, R.; Nocedal, J.; Mudigere, D.; Shi, H.; Tang, P.T.P.: A progressive batching l-bfgs method for machine learning. In Proceedings of the 35th International Conference on Machine Learning, vol. 80 of Proceedings of Machine Learning Research, 620–629, 2018. https://proceedings.mlr.press/v80/bollapragada18a.html.

[23] Chang, D.; Sun, S.; Zhang, C.: An accelerated linearly convergent stochastic l-bfgs algorithm. IEEE Transactions on Neural Networks and Learning Systems, 30(1), 3338–3346, 2019. http://doi.org/10.1109/TNNLS.2019.2891088.

[24] Wright, S.: Primal-dual interior-point methods. SIAM, 1997.

[25] Suykens, J.; Vandewalle, J.: Least squares support vector machine classifiers. Neural Processing Letters, 9, 293–300, 1999. http://doi.org/10.1023/A:1018628609742.

[26] Zhang, Z.; Chen, G.; Yang, S.: Ensemble support vector recurrent neural network for brain signal detection. IEEE Transactions on Neural Networks and Learning Systems, 33(11), 6856–6866, 2021. http://doi.org/10.1109/TNNLS.2021.3083710.

[27] Broyden, C.G.: The convergence of a class of double-rank minimization algorithms 1. general considerations. IMA Journal of Applied Mathematics, 6(1), 76–90, 1970. http://doi.org/10.1093/imamat/6.1.76.

[28] Fletcher, R.: A new approach to variable metric algorithms. The Computer Journal, 13(3), 317–322, 1970. http://doi.org/10.1093/comjnl/13.3.317.

[29] Goldfarb, D.: A family of variable metric updates derived by variational means. Mathematics of Computation, 24(109), 23–26, 1970. http://doi.org/10.1090/S0025-5718-1970-0258249-6.

[30] Shanno, D.F.: Conditioning of quasi-newton methods for function minimization. Mathematics of Computation, 24(111), 647–656, 1970. http://doi.org/10.2307/2004840.

[31] Nocedal, J.; Wright, S.: Numerical optimization. Springer Science & Business Media, 2006.

[32] Aeberhard, S.; Coletti, M.: Wine, 1991. http://doi.org/10.24432/C5PC7J.

[33] Fisher, R.A.: Iris, 1988. http://doi.org/10.24432/C56C76.

[34] Sejnowski, T.; Gorman, R.: Connectionist bench (sonar, mines vs. rocks). http://doi.org/10.24432/C5T01Q.

[35] Wolberg, W.H.; Mangasarian, O.L.; Street, W.N.: Breast cancer wisconsin (diagnostic), 1995. http://doi.org/10.24432/C5DW2B.

[36] Kaggle Inc.: Icr - identifying age-related conditions, 2023. https://www.kaggle.com/competitions/icr-identify-age-related-conditions/data?select=train.csv.

[37] Kaggle Inc.: Body signal of smoking, 2022. https://www.kaggle.com/datasets/kukuroo3/body-signal-of-smoking.

[38] Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, 2006.

[39] Sherman, J.; Morrison, W.J.: Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. The Annals of Mathematical Statistics, 21(1), 124–127, 1950.

[40] Cox, D.R.: The regression analysis of binary sequences. Journal of the Royal Statistical Society. Series B (Methodological), 20(2), 215–242, 1958. ISSN 00359246. http://www.jstor.org/stable/2983890.