Development of Shallow Machine Learning Models for Classification Problems

M.Sc. Thesis

by

Rahul Kumar Sharma

DEPARTMENT OF MATHEMATICS

INDIAN INSTITUTE OF TECHNOLOGY INDORE

MAY 2024

Development of Shallow Machine Learning Models for Classification Problems

A THESIS

Submitted in partial fulfillment of the

requirements for the award of the degree

of

Master of Science

by

Rahul Kumar Sharma

(Roll No. **2203141001**)

Under the guidance of

Dr. M. Tanveer

DEPARTMENT OF MATHEMATICS

INDIAN INSTITUTE OF TECHNOLOGY INDORE

MAY 2024

INDIAN INSTITUTE OF TECHNOLOGY INDORE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled **Development of Shallow Machine Learning Models for Classification Problems** in the partial fulfillment of the requirements for the award of the degree of **Master of Science** and submitted in the **Department of Mathematics**, **Indian Institute of Technology Indore**, is an authentic record of my own work carried out during the time period from July 2023 to May 2024 under the supervision of **Dr. M. Tanveer**, Associate Professor, Department of Mathematics, IIT Indore.

The matter presented in this thesis by me has not been submitted for the award of any other degree of this or any other institute.

Rahul 30/05/2024

Signature of the student with date

(**Rahul Kumar Sharma**)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

30/05/2024

Signature of Thesis Supervisor with date

(**Dr. M. Tanveer**)

Rahul Kumar Sharma has successfully given his M.Sc. Oral Examination held on May 27th, 2024.

V.K. Shenni

Signature of supervisor of M.Sc Thesis
Date:

Signature of Convener, DPGC
Date: 29/05/2024

“Dedicated to My Parents for Everything”

Acknowledgements

I extend my deepest gratitude to my Master's Thesis supervisor, **Dr. M. Tanveer**, Associate Professor in the Department of Mathematics at IIT Indore, for his invaluable guidance, unwavering support, and profound mentorship throughout this memorable journey. Dr. Tanveer's expertise in the domain of machine learning has been instrumental in shaping the trajectory of our project. His profound insights, meticulous attention to detail, and unwavering commitment to excellence have been a constant source of inspiration for me. Beyond academia, Dr. Tanveer's mentorship transcended conventional boundaries. He embraced the role of a parent figure on campus, offering not only academic guidance but also a nurturing environment where I could thrive and grow both personally and professionally. Thanks to his unwavering support, I am able to secure an offer at the Georgia Institute of Technology, USA, one of the premier institutions in STEM education.

Furthermore, I offer my thanks to Prof. Swadesh Kumar Sahoo, Prof. V. Antony Vijesh, Dr. Md Arshad, Dr. Sourav Mitra, and Dr. Bibekananda Maji for their persistent assistance, motivation, and guidance during my stay at IIT Indore.

I would like to extend my heartfelt gratitude to Prof. Niraj Kumar Shukla, Head of the Department of Mathematics, Dr. Vijay Kumar Sohani, DPGC, Department of Mathematics, and the entire Department of Mathematics for providing excellent facilities and support during my stay at IIT Indore.

I express my sincere appreciation to Mr. Abdul Quadir and Mr. Md Sajid members of OPTIMAL Lab, for their significant contributions to this endeavor. As mentors, collaborators, and, most importantly, as elder brothers, their consistent backing, inspiration, and direction were crucial in the development of this research project. Their readiness to exchange thoughts, knowledge, and assets has been priceless. They have undeniably served as the cornerstones of my academic progress.

I would like to extend my thanks to the individuals at the OPTIMization for MAchine Learning (OPTIMAL) Lab, specifically Dr. Ashwani Kumar Malik, Ms. Anuradha Kumari, Mr. Mushir Akhtar, Mr. Ritik Mishra, and Ms. Madhu, for their ongoing support and priceless advice. My journey in the OPTIMAL Lab has been both exciting and transformative, profoundly enriching my academic and professional development. The motivation and support I have received here are unparalleled, making my time in the OPTIMAL Lab a pivotal turning point in my academic progress.

I would like to express my sincere appreciation to Mr. Debdeep Roy for his unwavering support and guidance throughout my master's program at IIT Indore. His continuous encouragement has been instrumental in my academic achievements and personal growth during this period. Additionally, I would like to express my appreciation to the esteemed Ph.D. scholars of the Mathematics Department, including Mr. Aniruddha Deshmukh, Ms. Pinki Khatun, Ms. Manasa Bhat, Mr. Pranav Kumar, Mr. Dipendu Pramanik, Ms. Sheetal Wankhede, and Mr. Swaroop Georgy Zachariah, for their invaluable mentorship, assistance, and for creating a memorable experience during my time at IIT Indore. Their support has been crucial in shaping my academic journey and fostering a conducive learning environment.

I am deeply grateful to my parents for their constant love, unwavering support, and belief in my abilities throughout my academic journey. Their encouragement has been a steadfast pillar, providing me with the strength and motivation to pursue my goals. Additionally, I extend my heartfelt thanks to the special persons I met here, Nilesh, Pragya, Deepak, Harsh, and all my friends, classmates, and juniors whose fellowship and encouragement have been a continuous source of inspiration. Their support and shared enthusiasm for learning have made this journey not only successful but also memorable and enriching.

Finally, I would like to express my utmost appreciation to the Divine for the countless blessings and opportunities that have been granted to me. It is only through His benevolence that I have been able to achieve all that I have.

Rahul Kumar Sharma

Publications

1. *M. Tanveer,* ***R. K. Sharma****, A. Quadir, and M. Sajid. "Enhancing Robustness and Efficiency of Least Square Twin SVM via Granular Computing."* - **Under Review**

2. *M. Tanveer,* ***R. K. Sharma****, M. Sajid, and A. Quadir. "GRVFL-2V: Graph Random Vector Functional Link Neural Network Based on Two-View Learning."* - **Under Review**

Abstract

The shallow learning nature of hyperplane-based classifiers and randomized neural networks has played a crucial role in effectively tackling classification problems. These approaches have made significant strides in addressing the challenges associated with classifying data by utilizing simple decision boundaries and randomization techniques. Researchers have introduced various variants of hyperplane-based classifiers and randomized neural networks (RNNs) to improve classification performance by employing diverse machine-learning algorithms.

The least-square twin support vector machine (LSTSVM) is a hyperplane-based classifier that stands out as one of the state-of-the-art models. However, LSTSVM encounters several challenges, including sensitivity to noise and outliers, overlooking the SRM principle, and instability in resampling. Moreover, its computational complexity and reliance on matrix inversions hinder the efficient processing of large datasets. As a remedy to the aforementioned challenges of LSTSVM, we incorporate the concept of granular computing into LSTSVM, and in Chapter 3, we propose the novel granular ball least square twin support vector machine (GBLSTSVM). GBLSTSVM is trained using granular balls instead of original training data points. The granular balls are defined by their center and radius. The core of a granular ball is found at its center, where it encapsulates all the pertinent information of the data points that lie within the ball of a specified radius. GBLSTSVM has improved robustness against the effects of resampling and reduced vulnerability to noise and outliers. Further, we propose the novel large-scale granular ball least square twin support vector machine (LS-GBLSTSVM) to incorporate the SRM principle in GBLSTSVM through the inclusion of regularization terms. The proposed LS-GBLSTSVM model demonstrates exceptional efficiency, scalability for large datasets, and resilience against label noise and outliers.

The random vector functional link (RVFL) is a randomized neural network that has been extensively studied in recent times. However, due to its shallow learning nature, RVFL often fails to consider all the relevant information available in a dataset. Additionally, it overlooks the geometrical properties of the dataset. To address the limitations of RVFL, in Chapter 4, we propose a novel graph random vector functional link based on two-view learning (GRVFL-2V) model. The proposed GRVFL-2V model is trained on multiple views, incorporating the concept of multiview learning (MVL), and it also incorporates the geometrical properties of all the views using the graph embedding (GE) framework. The synergy between RVFL networks, MVL, and GE framework enables our proposed model to achieve the following: i) efficient learning: by leveraging the topology of RVFL, our proposed model can efficiently capture complex nonlinear relationships within the multi-view data, facilitating efficient and accurate predictions; ii) comprehensive representation: combining features from diverse perspectives enhances the proposed model's ability to capture complex patterns and relationships within the data, thereby improving the model's overall generalization performance; and iii) structural awareness: by employing the GE framework, our proposed model leverages the original data distribution of the dataset by naturally exploiting both intrinsic and penalty subspace learning criteria.

Contents

1 Introduction **3**

1.1 Background 4

1.2 Motivation 7

1.3 Objectives 11

1.4 Contributions of the Thesis 11

1.5 Organization of the Thesis 13

2 Literature Review **15**

2.1 Granular Computing 15

2.2 Least Square Twin Support Vector Machine 17

2.3 Granular Ball Support Vector Machine 18

2.4 Random Vector Functional Link Neural Network 19

2.5 Graph Embedding 20

2.6 Literature Review on Multiview Learning 21

3 Enhancing Robustness and Efficiency of Least Square Twin SVM via Granular Computing **23**

3.1 The Proposed Granular Ball Least Square Twin Support Vector Machine 23

3.1.1 Linear GBLSTSVM 24

3.1.2 Nonlinear GBLSTSVM 26

3.2 The Proposed Large Scale Granular Ball Least Square Twin Support Vector Machine 27

3.2.1 Linear LS-GBLSTSVM 27

3.2.2 Nonlinear LS-GBLSTSVM 29

3.3 Computational Complexity 31

3.4 Experimental Results and Discussions 32

3.4.1 Experimental Setup 32

3.4.2 Experiments on Real World UCI and KEEL Datasets on Linear Kernel . . . 32
3.4.3 Experiments on Real World UCI and KEEL Datasets on Gaussian Kernel . . . 36
3.4.4 Experiments on Real World UCI and KEEL Datasets with Added Label Noise on Linear and Gaussian Kernel 37
3.4.5 Experiments on NDC Datasets . . . 38
3.4.6 Sensitivity Analysis of Hyperparameters . . . 39
3.4.7 Sensitivity Analysis of Granular Parameters . . . 40

4 GRVFL-2V: Graph Random Vector Functional Link Based on Two-View Learning **53**
4.1 The proposed Graph Random Vector Functional Link Based on Two-View Learning model . . . 53
4.1.1 LFDA Under the GE Framework . . . 56
4.2 Computational Complexity . . . 57
4.3 Experiments, Results, and Discussions . . . 57
4.3.1 Experimental Setup . . . 58
4.3.2 Experiments on UCI and KEEL Datasets . . . 58
4.3.3 Experiments on Corel5k Datasets . . . 62
4.3.4 Experiments on Animal with Attributes (AwA) Datasets 65
4.3.5 Sensitivity Analysis of Hyperparameters c_1 and c_2 . . . 65
4.3.6 Sensitivity Analysis of Coupling Parameter ρ . . . 67
4.3.7 Sensitivity Analysis of Graph Regularization Parameter θ 67

5 Conclusions and Future Directions **69**
5.1 Conclusions . . . 69
5.2 Future Directions . . . 71

List of Figures

2.1 Pictorial representation of the process of granular ball generation. 16
2.2 Inseperable GBSVM model having support granular balls GB_j, support centers o_j, and support radii d_j. 18
2.3 The architecture of RVFL model. 19

3.1 Granular ball least square twin support vector machine model having two non-parallel hyperplanes f_1 and f_2. 24
3.2 The effect of hyperparameter (c_1, c_2) tuning on the accuracy (ACC) of some UCI and KEEL datasets on the performance of linear GBLSTSVM. 40
3.3 The effect of granular parameter (num, pur) tuning on the accuracy (ACC) of some UCI and KEEL datasets on the performance of linear GBLSTSVM. 41

4.1 The effect of hyperparameter (c_1, c_2) tuning on the accuracy (ACC) of some UCI and KEEL datasets on the performance of proposed GRVFL-2V model. 66
4.2 The effect of coupling parameter ρ tuning on the accuracy (ACC) of UCI and KEEl, AwA, and Corel5k datasets on the performance of proposed GRVFL-2V model. 67
4.3 The effect of graph embedding parameter θ tuning on the accuracy (ACC) of Corel5k and AwA datasets on the performance of proposed GRVFL-2V model. 68

List of Tables

3.1 Average accuracy and average rank of the baseline models and the proposed models over UCI and KEEL datasets with Linear kernel. 33

3.2 Wilcoxon-signed rank test of the baseline models w.r.t. the proposed GBLSTSVM over UCI and KEEL data with the Linear kernel. 34

3.3 Wilcoxon-signed rank test of the baseline models w.r.t. the proposed LS-GBLSTSVM over UCI and KEEL datasets with Linear kernel. 34

3.4 Pairwise win-tie-loss test of proposed GBLSTSVM and baseline models on UCI and KEEL datasets with linear kernel 35

3.5 Average accuracy and average rank of the baseline models and the proposed models over UCI and KEEL datasets with Gaussian kernel. 36

3.6 Wilcoxon-signed rank test of the baseline models w.r.t. the proposed GBLSTSVM over UCI and KEEL datasets with Gaussian kernel. 36

3.7 Wilcoxon-signed rank test of the baseline models w.r.t. the proposed LS-GBLSTSVM over UCI and KEEL datasets with Gaussian kernel. 37

3.8 Pairwise win-tie-loss test of proposed and baseline models on UCI and KEEL datasets with Gaussian kernel 37

3.9 Accuracy and time of the proposed GBLSTSVM and LS-GBLSTSVM with baseline models on NDC datasets with Linear kernel. . . 39

3.10 Performance of the proposed Linear GBLSTSVM across varying purities, showcasing the relationship between the number of Granular Balls and the resulting accuracies. 42

3.11 Performance of the proposed Linear LS-GBLSTSVM across varying purities, showcasing the relationship between the number of Granular Balls and the resulting accuracies. 42
3.12 Performance comparison of the proposed GBLSTSVM and LS-GBLSTSVM along with the baseline models based on classification accuracy using linear kernel for UCI and KEEL datasets. 44
3.13 Performance comparison of the proposed GBLSTSVM and LS-GBLSTSVM along with the baseline models based on classification accuracy using Gaussian kernel for UCI and KEEL datasets. 48

4.1 Performance comparison of the proposed GRVFL-2V along with the baseline models based on classification accuracy for UCI and KEEL datasets. 59
4.2 The statistical comparison of the proposed GRVFL-2V model with the baseline models on UCI and KEEL datasets using the Nemenyi post hoc test. 61
4.3 Pairwise win-tie-loss-sign test of proposed GRVFL-2V model and baseline models on UCI and KEEL datasets. 61
4.4 Performance comparison of the proposed GRVFL-2V along with the baseline models based on classification accuracy for Corel5k datasets. 63
4.5 Performance comparison of the proposed GRVFL-2V along with the baseline models based on classification accuracy for AwA datasets. 64

List of Abbreviations

ANN Artificial Neural Network

ELM Extreme Learning Machine

GBLSTSVM Granular Ball Least Square Twin Support Vector Machine

GBSVM Granular Ball Support Vector Machine

GD Gradient Descent

GE Graph Embedding

GRVFL-2V Graph Random Vector Functional Link based on Two-View Learning

LFDA Local Fisher Discriminant Analysis

LS-GBLSTSVM Large Scale Granular Ball Least Square Twin Support Vector Machine

LSTSVM Least Square Twin Support Vector Machine

MVL Multi View Learning

QPP Quadratic Programming Problem

RNN Randomized Neural Network

RVFL Random Vector Functional Link Network

SMO Sequential Minimal Optimization

SRM Structural Risk Minimization

SVM Support Vector Machine

TSVM Twin Support Vector Machine

Chapter 1

Introduction

The continuous advancements in technology and the availability of powerful computing resources has played a significant role in the development of advanced classification algorithms. This has enabled researchers to experiment with complex models and techniques, leading to the emergence of new models that offer improved performance and scalability in handling diverse data types and structures. The classification problems lie at the core of various applications across diverse domains, including image recognition [1], spam detection [2], medical diagnosis [3], sentiment analysis [4], and more.

At its essence, classification involves building models that can learn from labeled training data to accurately predict the class labels of unseen or new data instances. The goal is to develop algorithms that can generalize well, meaning they can make accurate predictions on unseen data beyond the training set. Mathematically, the classification problem can be expressed as:

$$y = f(x, w) \in \mathbb{Y}, \tag{1.1}$$

where the learning algorithm f assigns a label y to a new observation x and w is the parameter of the learning algorithm. $\mathbb{Y}$ signifies the set of categories or class labels. Depending on the cardinality of $\mathbb{Y}$, the classification problem can be categorized as either binary or multiclass. In binary class problems, there are two distinct categories, whereas in multiclass problems, more than two categories are present.

In this thesis, we discuss two state-of-the-art classification models: support vector machine (SVM) [5], which is a hyperplane-based classifier, and random vector functional link (RVFL) [6, 7] network, which is a randomized neural network (RNN) [8]. In the following section, we provide a concise overview of these models. We also outline the motivations and objectives that drive this thesis. Additionally, we highlight the contributions made within this research and conclude with a structured outline of the thesis content.

1.1 Background

Shallow machine learning models have become indispensable tools for classification tasks due to their simple architecture, effectiveness, and computational efficiency. In contrast to deep learning models with complex architectures and numerous hidden layers, shallow algorithms offer a more efficient learning paradigm for pattern recognition and classification. Shallow machine-learning models that have been extensively studied and proven effective include hyperplane-based classifiers like SVM, twin support vector machine (TSVM) [9], and least square twin support vector machine (LSTSVM) [10]. Additionally, randomized neural networks such as RVFL, extreme learning machine (ELM) [11], and broad learning systems (BLS) [12], and their variants are also widely recognized as successful shallow machine learning models.

SVM is a powerful model in machine learning that uses kernels to determine the best hyperplane between classes in classification tasks precisely. SVM provides a deterministic classification result. Hence, its application can be found in various domains such as health care [13], anomaly detection [14], web mining [15], electroencephalogram (EEG) signal classification [16], Alzheimer's disease diagnosis [17], and so on. SVM implements the structural risk minimization (SRM) principle and, hence, shows better generalization performance. SVM solves a convex quadratic programming problem (QPP) to find the optimal separating hyperplane.

However, the effectiveness and efficiency of SVM are limited when dealing with large datasets due to the increase in computational complexity. Further, SVM is sensitive to noise, especially along the decision boundary, and is unstable to resampling. To mitigate the effects of noise and outliers in data points, fuzzy SVM (FSVM) [18] was proposed. The incorporation of pinball loss in SVM [19] led to a better classifier that has the same computational complexity as SVM but is insensitive to noise and stable to resampling. Furthermore, in [20], the authors proposed a novel general TSVM with pinball loss (Pin-GTSVM) for solving classification problems. To overcome the issue of the computational complexity of SVM, some non-parallel hyperplane-based classifiers have been proposed, such as generalized eigenvalue proximal SVM (GEPSVM) [21] and twin SVM (TSVM) [9].

The GEPSVM and TSVM generate non-parallel hyperplanes that position themselves closer to the samples of one class while maximizing their distance from samples belonging to the other class. GEPSVM solves two generalized eigenvalue problems, and its solutions are determined by selecting the eigenvectors corresponding to the smallest eigenvalue. However, TSVM solves two smaller QPPs, which makes its learning process approximately four times faster than SVM [9]. Also, TSVM shows better generalization performance than GEPSVM [9]. The effectiveness of TSVM may decrease due to its vulnerability to noise and outliers, potentially leading to unsatisfactory results. Moreover, its substantially high computational complexity and reliance on matrix inversions pose significant challenges, especially when dealing with large datasets, thereby impeding its real-time applications. Also, TSVM does not adhere to the SRM principle, making the model susceptible to overfitting.

Kumar and Gopal [10] proposed least squares TSVM (LSTSVM). Unlike TSVM, LSTSVM incorporates an equality constraint in the primal formulation instead of an inequality constraint. This modification allows LSTSVM to train much faster than TSVM, as it solves a system of linear equations to determine the optimal nonparallel separating hyperplanes. However, despite its success in reducing training time, the methodology of LSTSVM involves the computation of matrix inverses, which limits its applicability to large datasets. Additionally, the LSTSVM's ability to learn decision boundaries can be significantly affected by the presence of noisy data and outliers.

Some recent advancements in TSVM and LSTSVM include capped $l_{2,p}$-norm metric-based robust LSTSVM for pattern classification [22], the Laplacian l_p norm LSTSVM [23], sparse solution of least- squares twin multi-class support vector machine using l_0 and l_p-norm for classification and feature selection [24], the inverse free reduced universum TSVM for imbalanced data classification [25], symmetric LINEX loss TSVM for robust classification and its fast iterative algorithm [26], and intuitionistic fuzzy weighted least squares TSVM (IFW-LSTSVM) [27]. Recent advancements in loss function theory for SVM and TSVM include the integration of a novel RoBoSS loss function in [28] and the incorporation of an asymmetric wave loss function in [29]. These improvements aim to enhance the robustness, sparsity, and smoothness of the loss functions used in SVM and TSVM models. These advancements contribute to the ongoing progress in SVM, TSVM, and LSTSVM techniques. A comprehensive overview of the various versions of TSVM can be found in [30]. Furthermore, in [31], the authors have conducted a thorough assessment of twin SVM-based classifiers on UCI datasets. They have examined the effectiveness of 8 different variants of TSVM-based classifiers, evaluating a total of 187 classifiers across 90 UCI benchmark datasets.

The artificial neural networks (ANNs) belong to the class of non-parametric learning methods that are used for estimating or approximating functions that may depend on a large number of inputs and outputs [32]. Inspired by the topology of biological neural networks, ANNs have found applications in diverse domains, such as image recognition [33], Alzheimer's disease diagnosis [34], stock price prediction [35], brain age prediction [36], and so on. The efficacy of ANNs depends upon several factors, including the quality and quantity of data, computational resources, and the effectiveness of underlying algorithms [37]. The ANNs are trained by optimizing a cost function, which quantifies the disparity between model predictions and actual observations. The adjustment of the network's weights and biases is facilitated through backpropagation, employing an iterative technique known as the gradient descent (GD) method. To enhance the agreement between predicted outcomes and actual observations, the optimization process requires fine-tuning the network's weights and biases. Nevertheless, GD-based algorithms come with inherent limitations, including slow convergence [38], difficulty in achieving global minima [39] and heightened sensitivity to the selection of learning rate and the point of initialization.

Randomized neural networks (RNNs) [40] have emerged as a unique solution to overcome the difficulties faced in training ANNs using gradient descent (GD) methods. Unlike conventional ANNs, RNNs adopt a distinct approach where the network's weights and biases are randomly selected from a specified range and remain constant throughout the training process. The determination of output layer parameters in RNNs is achieved through a closed-form solution [41], as opposed to iterative-based optimization techniques commonly employed in traditional ANNs. The prominent examples of RNNs include the random vector functional link (RVFL) network [6, 42] and the extreme learning machines (ELM) [11]. RVFL has garnered significant attention within the realm of RNNs due to its distinctive features, including direct connections between input and output layers that improve generalization performance, a straightforward architecture aligned with Occam's Razor principle and PAC learning theory [43], a reduced number of parameters, and the universal approximation capability [44]. Furthermore, the direct connections help to regulate the randomness in RVFL [32, 45], while the output parameters are computed analytically through methods like pseudo-inverse or least-square techniques.

Various enhancements have been done to optimize the learning capabilities of RVFL, with notable advancements such as the sparse pre-trained random vector functional link (SP-RVFL) network [46] and the discriminative manifold RVFL neural network (DM-RVFLNN) [47]. The SP-RVFL network utilizes diverse weight initialization methods to improve the generalization performance of standard RVFL models, while the DM-RVFLNN integrates manifold learning with a soft label matrix to enhance the model's discriminative capacity by increasing the margin between samples from different classes. In addition, techniques such as manifold learning based on an in-class similarity graph have been employed to enhance the compactness and similarity of samples within the same class. To address noise and outliers, Cui et al. [48] proposed an RVFL-based approach that incorporates a novel feature selection (FS) technique, enhancing the efficiency and robustness of RVFL through the augmented Lagrangian method. Furthermore, the integration of a kernel function into RVFL has led to the development of the kernel exponentially extended RVFL network (KERVFLN) [49], demonstrating the adaptability and potential of RVFL in various machine learning applications. Recent advancements in RVFL theory include RVFL+, kernel RVFL+ (KRVFL+) [50], incremental learning paradigm with privileged information for RVFL (IRVFL+), and improved fuzziness-based RVFL [51, 52, 53, 54], showcasing the ongoing evolution and versatility of RVFL models in addressing diverse machine-learning challenges, including multilabel classification tasks using RNNs. Some recent advancements in RVFL theory include neuro-fuzzy RVFL for classification and regression problems [45], kernel ridge regression-based randomized network for brain age classification and estimation [55], etc.

1.2 Motivation

The concept of "large-scale priority" aligns with the natural information-processing mechanism of the human brain [56]. Granulation, or breaking down information into smaller, more manageable parts, is a fundamental aspect of the learning process. Our brains are wired to absorb and process information in layers, starting with a broader concept and then delving into the specifics as needed. This approach allows us to grasp the big picture first and then gradually fill in the details, leading to a more comprehensive understanding. Drawing inspiration from the brain's functioning, granular computing explores problems at various levels of detail. Coarser granularity emphasizes important components, thereby enhancing the effectiveness of learning and resistance to noise. Conversely, finer granularity offers intricate insights that deepen knowledge.

In contrast, a majority of machine learning models rely on pixels or data points for training at the lowest possible resolution. Consequently, they are often more susceptible to outliers and noise. This approach lacks the efficiency and scalability of the brain's adaptable granulating capabilities. The novel classifier based on granular computing and SVM [57] was developed to incorporate the concept of granular balls. This classifier utilizes hyper-balls to partition datasets into different sizes of granular balls [58]. As highlighted in [59], this approach, which imitates cognitive processes observed in the human brain, offers a scalable, dependable, and efficient solution within the realm of granular computing by introducing larger granularity sizes. However, this transition may compromise the accuracy of fine details. Conversely, finer granularity enhances the focus on specific features, potentially improving precision but also introducing challenges in terms of robustness and efficiency in noisy scenarios. Consequently, striking the right balance between granularity and size becomes imperative. Researchers persistently explore novel applications, refine existing methodologies [60, 61, 62], and bridge interdisciplinary gaps to harness the potential of granular computing across diverse domains.

Recently, the granular ball SVM (GBSVM) [63] has been proposed, which integrates the concepts of SVM and granular computing. GBSVM exhibits good performance in effectively managing datasets that are contaminated with noise and outliers. Motivated by the robustness and efficiency demonstrated by the GBSVM, in Chapter 3, we incorporate the concept of granular computing into the LSTSVM and propose a novel model called the granular ball least square twin support vector machine (GBLSTSVM). This integration aims to address the inherent drawbacks and complexities associated with LSTSVM. GBLSTSVM uses granular balls as input to construct non-parallel separating hyperplanes by solving a system of linear equations like in the case of LSTSVM. The granular balls are characterized by their center and radius. The essence of a granular ball lies in its center, which encapsulates all the relevant information of the data points that lie within the ball. In comparision to LSTSVM, GBLSTSVM provides enhanced efficiency, a heightened resistance to noise and outliers, robustness to resampling, and is trained using a substantially reduced number of training instances, thereby significantly reducing the training time. However, GBLSTSVM lacks the SRM principle, which can lead to the potential risk of overfitting. To address this, we further propose the novel large-scale GBLSTSVM (LS-GBLSTSVM) model. LS-GBLSTSVM incorporates the regularization terms in its primal form of the optimization problem which eliminates the need for matrix inversions and also helps to mitigate the risk of overfitting.

In various real-world applications, a multitude of characteristics is often observed, requiring representation through multiple feature sets. This leads to the prevalence of multiview data, where information from various measurement methods is collected to comprehensively capture the nuances of each example rather than relying solely on a single feature set. For instance, a picture can be described by color or texture features, and a person can be identified by face or fingerprints [64]. Web pages serve as a quintessential example of multimodal data, where one feature vector encapsulates the words within the webpage text, while another feature vector captures the words present in the links pointing to the webpage from other pages. While individual views may suffice for specific learning tasks, there is potential for enhancement by amalgamating insights from multiple data representations [65]. Multiview learning (MVL), a well-established collection of techniques, holds significant potential as multi-modal datasets become increasingly accessible [66]. MVL models are often developed under the supervision of the consensus or complimentary principles [67] to ensure the effectiveness of an algorithm. The consensus principle aims to enhance the performance of classifiers for each view by maximizing consistency across multiple viewpoints. Conversely, the complementarity principle emphasizes the importance of providing complementary data from diverse perspectives to offer a comprehensive and accurate description of the object.

The integration of the graph embedding (GE) [68] framework into the RVFL model has played a crucial role in enhancing the learning process of RVFL [53, 54]. By incorporating the GE framework, the RVFL model is able to capture the geometric relationships present within the dataset, which were previously overlooked [42]. To incorporate the geometrical relationships within a dataset, Malik et al. [53] introduced the graph-embedded intuitionistic fuzzy weighted RVFL (GE-IFWRVFL) model. This model not only integrates the intuitionistic fuzzy (IF) membership scheme to handle noisy data and outliers but also preserves the geometric characteristics through the GE framework.

The incorporation of IF and GE has proven to enhance the resilience and performance of the RVFL model significantly. Recent advancements in RVFL have seen the utilization of GE framework in extended graph-embedded RVFL (EGERVFL) and graph-embedded intuitionistic fuzzy RVFL network for class imbalance learning (GE-IFRVFL-CIL) as proposed in [54] and [69] respectively. The experimental results from [53, 54, 69], along with studies in [42], strongly support the notion that the performance of RVFL is greatly improved through the integration of the GE framework. This emphasizes the importance of considering the geometric attributes of a dataset in the learning process to enhance the overall performance of RVFL.

RVFL is categorized as a shallow learning algorithm due to its single hidden layer, which may potentially hinder its ability to fully grasp complex patterns and subtle nuances in a dataset. While RVFL may excel in specific classification tasks, its shallow architecture could limit its capability to accurately interpret intricate patterns or extract detailed features from the dataset. The simplicity of RVFL's architecture offers advantages in terms of computational efficiency and ease of implementation. However, when confronted with datasets containing intricate patterns or nuanced relationships among variables, more sophisticated learning frameworks are necessary to effectively capture the underlying complexities inherent in the dataset. To mitigate the limitations imposed by the shallow learning nature of RVFL, presenting the same information from multiple perspectives can offer a solution. By training RVFL on multiple views of the data, it becomes possible to compensate for its inherent shallow learning structure and enhance its ability to comprehend intricate patterns and subtle nuances within the dataset. Hence, leveraging the multiview learning (MVL) framework can significantly enhance the learning efficiency of RVFL.

Recognizing the importance of a dataset's inherent geometrical characteristics and MVL, in Chapter 4, we propose the novel graph random vector functional link based on two-view learning (GRVFL-2V). The GRVFL-2V method incorporates the intrinsic and penalty graphical representations of multiview datasets within the GE framework, thereby capturing the geometrical properties of the multiview data. By integrating the MVL and GE framework in RVFL, the learning capabilities of RVFL are greatly enhanced, allowing it to effectively tackle classification challenges. In order to ensure simplicity and effectiveness, the proposed GRVFL-2V model incorporates two views at a time. This strategy allows for a balance between complexity and performance, as it utilizes the complementary information from these two views to enhance effectiveness while still maintaining simplicity. By leveraging information from multiple views, GRVFL-2V significantly improves the classification performance of RVFL by integrating both geometric and discriminative information from both views using intrinsic and penalty-based subspace learning criteria within the GE framework.

1.3 Objectives

The following are the objectives of this Thesis:

[1] To present a literature review on granular computing and multiview learning and to provide a brief theory of LSTSVM, GBSVM, RVFL, and the graph embedding framework.

[2] To develop a novel classification model incorporating granular computing in LSTSVM and a novel RVFL model for classification based on multiview learning with a graph embedding framework.

[3] To evaluate the computational complexity and statistical significance of the proposed models.

1.4 Contributions of the Thesis

Here, we give a summary of the contribution of the thesis. The main contributions of the thesis are as follows:

[1] We propose the novel GBLSTSVM by incorporating granular computing in LSTSVM. The GBLSTSVM is trained by feeding granular balls as input instead of data points for constructing optimal non-parallel separating hyperplanes. The use of granular balls reduces the training time by a substantial amount, amplifies the model's performance, and elevates the robustness against noise and outliers.

[2] We propose the novel LS-GBLSTSVM by implementing the SRM principle through the inclusion of regularization terms in the primal formulation of GBLSTSVM. LS-GBLSTSVM does not require matrix inversion, making it suitable for large-scale problems. In addition, LS-GBLSTSVM offers robust overfitting control, noise and outlier resilience, and improved generalization performance.

[3] We present the meticulous mathematical frameworks for both GBLSTSVM and LS-GBLSTSVM on linear and Gaussian kernel spaces. The formulation integrates the centers and radii of all granular balls used in training into the LSTSVM model. These models excel in capturing complex data patterns and relationships through sophisticated nonlinear transformations.

[4] We conducted the experiments of our proposed GBLSTSVM and LS-GBLSTSVM models using 34 UCI [70] and KEEL [71] datasets with and without label noise. Our comprehensive statistical analyses demonstrate the significantly superior generalization abilities of our proposed models compared to LSTSVM and the other baseline models. Further, we performed experiments on NDC [72] datasets of sample sizes ranging from 10,000 to 5 million to determine scalability. The results demonstrated that our proposed models surpass the baseline models in terms of accuracy, efficiency, robustness, and scalability.

[5] Furthermore, we present a generic framework that integrates random vector functional link (RVFL) with multiview learning (MVL) and graph embedding (GE) [73] framework. This novel model is called the graph random vector functional neural network based on two-view learning (GRVFL-2V).

[6] The proposed GRVFL-2V model is developed upon the foundation of the RVFL architecture, which is known for its simplicity and efficiency. However, it goes beyond the limitations of shallow learning of RVFL by integrating the concept of multiview learning (MVL). By leveraging multiple views, the model aims to enhance classification performance using multiple perspectives of the dataset.

[7] The proposed GRVFL-2V model integrates geometrical information from multiview data by embedding intrinsic and penalty subspace learning (SL) criteria within the GE framework. It employs local Fisher discriminant analysis (LFDA) [74] and graph regularization parameters to effectively utilize the GE framework.

[8] To trade off the information from multiple views, our proposed mathematical formulation of GRVFL-2V includes a coupling term. This coupling term helps to mitigate errors between the views, resulting in improved generalization performance.

[9] Our experiments for the proposed GRVFL-2V encompassed 27 UCI [70] and KEEL [71] datasets, 50 datasets from Corel5k[1], and 45 datasets from AwA[2]. Through comprehensive statistical analyses, we demonstrate the superior generalization performance of our proposed model when compared to baseline models.

1.5 Organization of the Thesis

The work in this thesis has been divided into five main chapters. The brief description of each chapter is given below:

[1] In Chapter 2, we provide a comprehensive literature review on granular computing and multiview learning. Additionally, we delve into the mathematical formulations of several key models: the least squares twin support vector machine (LSTSVM), the granular ball support vector machine (GBSVM), the random vector functional (RVFL) neural network, and the graph embedding (GE) framework.

[2] In Chapter 3, we introduce the granular ball least squares twin support vector machine (GBLSTSVM) and the large-scale least squares twin support vector machine (LS-GBLSTSVM). We also present experimental results and conduct statistical analyses of the proposed models.

[3] In Chapter 4, we propose the graph random vector functional link model based on two-view learning (GRVFL-2V). Additionally, we provide comprehensive experimental results and statistical analyses of the proposed model.

[4] In Chapter 5, we summarize the contributions of this thesis and outline potential future research directions.

[1] https://wang.ist.psu.edu/docs/related/

[2] http://attributes.kyb.tuebingen.mpg.de

Chapter 2

Literature Review

Within this chapter, a thorough examination of granular computing and multi-view learning (MVL) is provided, offering a comprehensive literature overview. Furthermore, a concise and sufficient mathematical depiction is presented for the least square twin support vector machine (LSTSVM), granular ball support vector machine (GBSVM), random vector functional link (RVFL) neural network, and graph embedding (GE) framework.

2.1 Granular Computing

In 1996, Lin and Zadeh proposed the concept of "granular computing". It becomes computationally expensive to process every data point in the space when dealing with large datasets. The objective of granular computing is to reduce the number of training data points required for machine learning models. The core idea behind granular computing is to use granular balls to completely or partially cover the sample space. This captures the spirit of data simplification while maintaining representativeness during the learning process. Granular balls, characterized by two parametric simple representations, a center o and a radius d, are the most appropriate choice for effectively handling high-dimensional data. Given a granular ball (GB) containing the datapoints $\{x_1, x_2, \ldots, x_p\}$, where $x_i \in \mathbb{R}^{1\times N}$, the center o of a GB is the center of gravity for all sample points in the ball, and d is equal to the average distance from o to all other points in GB. Mathematically, they can be calculated as: $o = \frac{1}{p}\sum_{i=1}^{p} x_i$ and $d = \frac{1}{p}\sum_{i=1}^{p} ||x_i - o||$. The average distance is utilized to calculate the radius d as it remains unaffected by outliers and aligns appropriately with the distribution of the data. The label assigned to a granular ball is determined by the labels of the data points that have the maximum

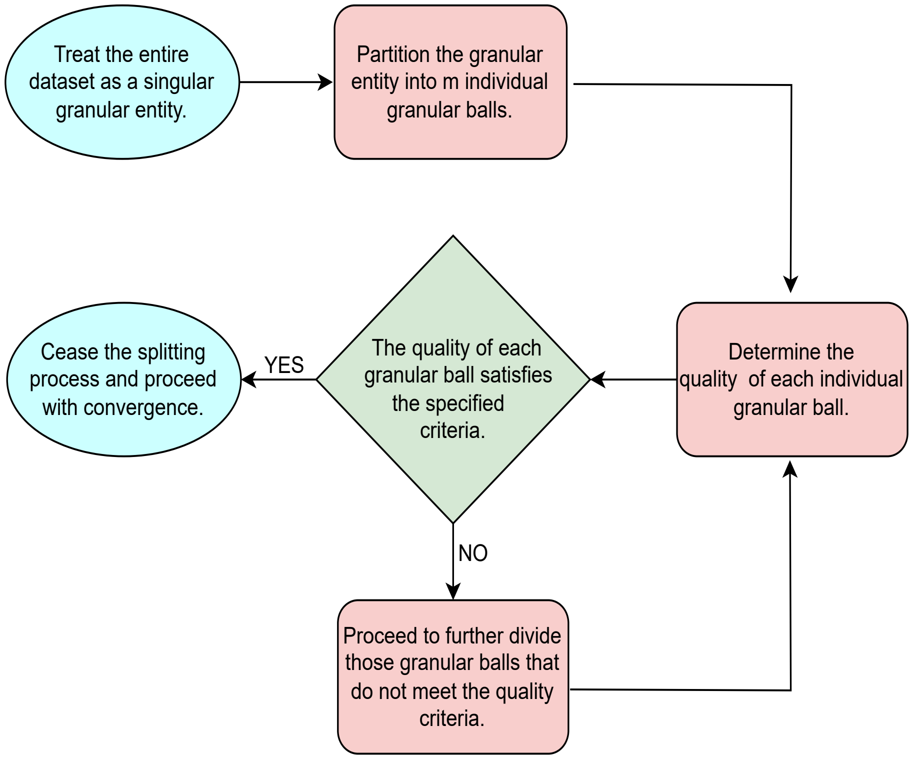


Figure 2.1: Pictorial representation of the process of granular ball generation.

frequency within the ball. To quantitatively assess the amount of splitting within a granular ball, the concept of "threshold purity" is introduced. This threshold purity represents the percentage of the majority of samples within the granular ball that possess the same label. The number of granular balls generated on T is given by the following optimization problem:

$$\begin{aligned} \min \ & \gamma_1 \times \frac{m}{\sum_{j=1}^{k} |GB_j|} + \gamma_2 * k, \\ \text{s.t.} \ & quality(GB_j) \geq \rho, \end{aligned}$$

where γ_1 and γ_2 are weight coefficients. ρ is the threshold purity. $|.|$ represents the cardinality of a granular ball, and m and k represent the number of samples in T and the number of granular balls generated on T, respectively. The quality of each granular ball is adaptive [75]. Initially, the whole dataset is considered as a single granular ball, which fails to represent the dataset's distribution accurately and exhibits the lowest level of purity. In the cases where the purity of this granular ball falls below the specified threshold, it is necessary to divide it multiple times until all sub-granular balls achieve a purity level equal to or higher than the threshold purity. As the purity of the granular balls increases, their alignment with the original dataset's data distribution improves. Fig. 1 depicts the procedure of granular ball generation.

2.2 Least Square Twin Support Vector Machine

Suppose matrix $A \in \mathbb{R}^{m_1 \times N}$ and $B \in \mathbb{R}^{m_2 \times N}$ contains all the training data points belonging to the $+1$ and -1 class, respectively. The primal problem of LSTSVM [10] can be expressed as:

$$\begin{aligned} &\min_{w_1, b_1} \ \frac{1}{2}||Aw_1 + e_1 b_1||^2 + \frac{c_1}{2}||q_1||^2, \\ &\text{s.t.} \quad -(Bw_1 + e_2 b_1) + q_1 = e_2, \end{aligned}$$

and

$$\begin{aligned} &\min_{w_2, b_2} \ \frac{1}{2}||Bw_2 + e_2 b_2||^2 + \frac{c_2}{2}||q_2||^2, \\ &\text{s.t.} \ (Aw_2 + e_1 b_2) + q_2 = e_1, \end{aligned}$$

where q_1 and q_2 are slack variables. Substituting the equality constraint into the primal problem, we get

$$\min_{w_1, b_1} \ \frac{1}{2}||Aw_1 + e_1 b_1||^2 + \frac{c_1}{2}||Bw_1 + e_2 b_1 + e_2||^2, \tag{2.1}$$

and

$$\min_{w_2, b_2} \ \frac{1}{2}||Bw_2 + e_2 b_2||^2 + \frac{c_2}{2}||Aw_2 + e_1 b_2 - e_1||^2. \tag{2.2}$$

Taking gradient of (2.1) with respect to w_1 and b_1 and solving, we get

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = -\left[F'F + \tfrac{1}{c_1}E'E\right]^{-1} F'e_2. \tag{2.3}$$

Similarly, for (2.2), we get

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = \left[E'E + \tfrac{1}{c_2}F'F\right]^{-1} E'e_1, \tag{2.4}$$

where $E = \begin{bmatrix} A & e_1 \end{bmatrix}$ and $F = \begin{bmatrix} B & e_2 \end{bmatrix}$.

Once the optimal values of (w_1, b_1) and (w_2, b_2) are obtained using (2.3) and (2.4) respectively. The categorization of a new input data point $\mathrm{x} \in \mathbb{R}^{1 \times N}$ into either the $+1$(class-1) or -1(class-2) class can be determined as follows:

$$\text{Class}(\mathrm{x}) = \underset{i \in \{1,2\}}{\text{argmin}} \Big(\frac{\|w_i \mathrm{x} + b_i\|}{\|w_i\|} \Big).$$

2.3 Granular Ball Support Vector Machine

The basic idea of GBSVM [63] is to mimic the classical SVM [5] using granular balls during the training process instead of data points. This makes GBSVM efficient and robust in comparison to SVM. The parallel hyperplanes in GBSVM are constructed using supporting granular balls GB_j having support center o_j and support radius d_j. Fig. 2 depicts the construction of inseparable GBSVM using supporting granular balls.

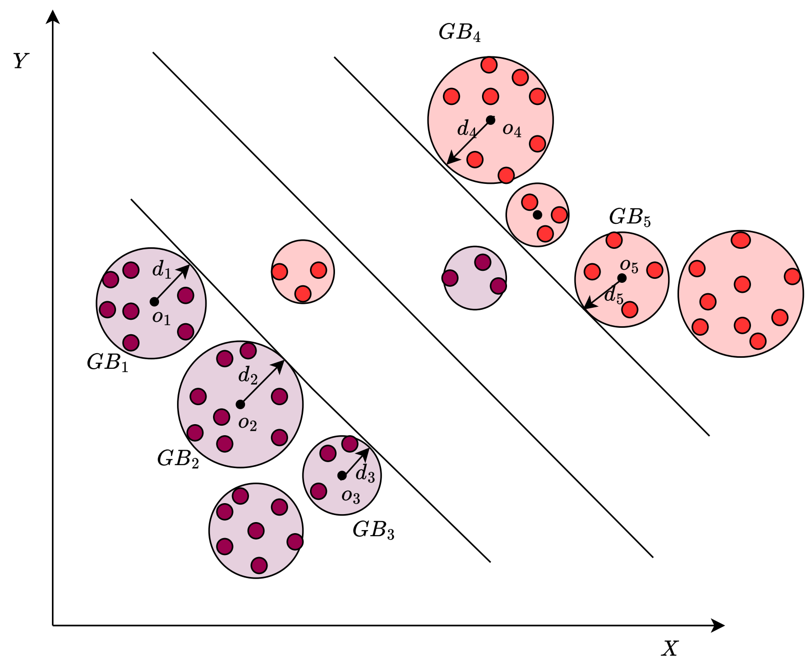


Figure 2.2: Inseperable GBSVM model having support granular balls GB_j, support centers o_j, and support radii d_j.

The GBSVM model can be expressed as:

$$\begin{aligned} \min_{w,b,\xi_j} \quad & \frac{1}{2}||w||^2 + C\sum_{j=1}^{k} \xi_j, \\ \text{s.t.} \quad & y_j(wo_j + b) - ||w||d_j \geq 1 - \xi_j, \\ & \xi_j \geq 0, \;\; j = 1, 2, 3, ..., k. \end{aligned} \tag{2.5}$$

The dual of GBSVM formulation (2.5) is:

$$\max_{\alpha} \quad -\frac{1}{2}||w||^2 + \sum_{j=1}^{k} \alpha_j,$$

$$\text{s.t.} \quad \sum_{j=1}^{k} \alpha_j y_j = 0,$$

$$0 \leq \alpha_j \leq C, \quad j = 1, 2, 3, ..., k,$$

where α_j's are Lagrange multipliers.

2.4 Random Vector Functional Link Neural Network

The RVFL model consists of an input layer, a hidden layer, and an output layer. The input and output layers are connected through the hidden layer, which acts as a bridge between them. Notably, the original features are also directly passed to the output layer because of the direct connections between the input and output layers. During the training process, the weights connecting the input and hidden layers, as well as the biases at the hidden layer, are randomly generated. Once generated, these parameters remain fixed and do not require any adjustments during the training phase. In terms of determining the output weight matrix connecting the input layer and the hidden layer to the output layer, the least squares or pseudo-inverse methods are employed, providing an analytical solution.

Consider a training matrix X with dimensions $l \times p$. Let W_1 be a weight

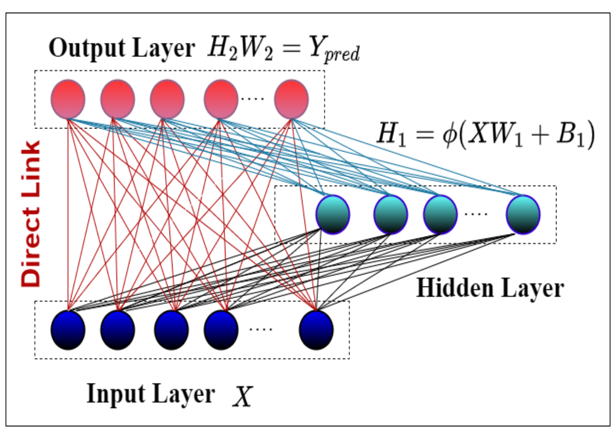


Figure 2.3: The architecture of RVFL model.

matrix with dimensions $p \times h$, where the values are randomly generated from a uniform distribution within the range of $[-1, 1]$ and $B_1 \in \mathbb{R}^{l\times h}$ (all the columns are identical) be the randomly generated bias matrix. To obtain the hidden layer matrix, also called the randomized feature layer, denoted as H_1, we apply a nonlinear activation function ϕ to the matrix $XW_1 + B_1$. Consequently, the hidden layer matrix H_1 can be expressed as:

$$H_1 = \phi(XW_1 + B_1).$$

Therefore, the matrix H_1 can be represented as:

$$H_1 = \begin{bmatrix} \phi(x_1w_1 + b_1) & \cdots & \phi(x_1w_h + b_h) \\ \vdots & \vdots & \vdots \\ \phi(x_pw_1 + b_1) & \cdots & \phi(x_pw_h + b_h) \end{bmatrix},$$

where $x_i \in \mathbb{R}^{1\times p}$ and $w_j \in \mathbb{R}^{p\times 1}$ represent the i^{th} row of X and the j^{th} column of W_1, respectively. The term b_j represents the bias of the j^{th} hidden node.

Let H_2 be a concatenated matrix of features from the input and hidden layers, represented as $H_2 = \begin{bmatrix} X & H_1 \end{bmatrix} \in \mathbb{R}^{l\times(p+h)}$. Here, X represents the input features, and H_1 represents the output of the hidden layer. Let W_2 denote the weights matrix connecting the concatenation of input (X) and hidden (H_1) layers to the output layer, with dimensions $W_2 \in \mathbb{R}^{(p+h)\times 2}$. The predicted output matrix $Y_{pred} \in \mathbb{R}^{l\times 2}$ of the RVFL is calculated using the following matrix equation:

$$H_2W_2 = Y_{pred}. \tag{2.6}$$

The optimization problem of (2.6) is given as:

$$\begin{aligned} &\min_{W_2} \ \frac{1}{2}||W_2||_2^2 + \frac{c}{2}||\xi||_2^2 \\ &\text{s.t. } \ H_2W_2 - Y_{true} = \xi, \end{aligned} \tag{2.7}$$

where c is a regularization parameter.

The solution of (2.7) is given by:

$$(W_2)_{min} = \begin{cases} ({H_2}^tH_2 + \frac{1}{c}I)^{-1}{H_2}^tY_{true}, & (p+h) \leq l \\ {H_2}^t(H_2H_2^t + \frac{1}{c}I)^{-1}Y_{true}, & l < (p+h). \end{cases}$$

2.5 Graph Embedding

The concept of graph embedding (GE) framework [73] aims to capture the underlying graphical structure of data in a vector space. In this framework, a given input matrix X is utilized to define the intrinsic graph $\mathcal{G}^{int} = \{X, \Delta^{int}\}$

and the penalty graph $\mathcal{G}^{pen} = \{X, \Delta^{pen}\}$ for the purpose of subspace learning (SL) [68]. The similarity weight matrix $\Delta^{int} \in \mathbb{R}^{l \times l}$ incorporates weights that represent the pairwise relationships between the vertices in X. Conversely, the penalty weight matrix $\Delta^{pen} \in \mathbb{R}^{l \times l}$ assigns penalties to specific relationships among the graph vertices. The optimization problem for graph embedding is formulated as follows:

$$
\begin{aligned}
v^* &= \underset{tr({v_0}^t X^t \mathbb{U} X v_0) = d}{\operatorname{argmin}} \sum_{k \neq l} ||{v_0}^t x_k - {v_0}^t x_l|| \Delta^{int}_{kl}, \\
&= \underset{tr({v_0}^t X^t \mathbb{U} X v_0) = d}{\operatorname{argmin}} \quad tr({v_0}^t X^t \mathbb{L} X v_0),
\end{aligned} \tag{2.8}
$$

where $tr(\cdot)$ is the trace operator and d is a constant value. $\mathbb{L} = \mathbb{D} - \Delta^{int} \in \mathbb{R}^{l \times l}$ is a representation of the Laplacian matrix of the intrinsic graph $\mathcal{G}^{\text{int}}$, with the diagonal elements of $\mathbb{D}$ being defined as $\mathbb{D}_{kk} = \sum_l \Delta^{int}_{kl}$. Moreover, $\mathbb{U} = \mathbb{L}^p = \mathbb{D}^p - \Delta^{pen}$ serves as the Laplacian matrix of the penalty graph $\mathcal{G}^{\text{pen}}$. The matrix v_0 is associated with the projection matrix. Equation (2.8) can be simplified to a generalized eigenvalue problem as shown in the form [76]:

$$
G_{int} s = \lambda G_{pen} s,
$$

where $G_{int} = X^t \mathbb{L} X$ and $G_{pen} = X^t \mathbb{U} X$. This simplification implies that the transformation matrix will be constructed from the eigenvectors of the matrix $G = G_{pen}^{-1} G_{int}$, where G integrates the intrinsic and penalty graph connections of the data samples.

2.6 Literature Review on Multiview Learning

Multiview learning (MVL) is an emerging area of research that holds significant promise in enhancing the generalization performance across various learning tasks. By integrating multiple feature sets, each offering unique and complementary insights, MVL has the potential to significantly improve overall model performance [77, 78, 79]. The abundance of diverse data types in practical applications has led to the development of MVL. In real-world scenarios, samples from different perspectives may reside in distinct spaces or exhibit vastly different distributions, often due to significant disparities between views [80, 81]. However, conventional approaches typically address such data by employing a cascade strategy, wherein multiview data is amalgamated into a single-view format through the concatenation of heterogeneous feature spaces into a homogeneous one. However, this cascading approach overlooks the distinctive statistical properties of each view and is plagued by the curse of dimensionality problems. One notable advantage of MVL is that it can enhance the performance of a standard single-view approach by leveraging manually generated multiple views.

In MVL, a distinct function is developed for each view, with the overall goal of constructing a unified function to optimize all individual functions jointly. This approach enhances the generalization performance of the framework across multiple views. According to Zhao et al. [77], MVL models can be categorized into three main groups: co-training style algorithms, co-regularization style algorithms, and margin consistency style algorithms. Co-training style algorithms focus on enhancing mutual agreement among diverse views. Conversely, co-regularization style algorithms aim to minimize discrepancies during the learning process. MVL learning has been successfully implemented in various hyper-plane based classifiers, such as SVM-2K [82], multiview twin support vector machine [83] (MvTSVM), etc.

MVL has demonstrated its effectiveness across various application scenarios, including enhancing image classification, annotation, and retrieval performance [84], predicting financial distress [85], forecasting multiple stages of Alzheimer's Disease progression [86], and identifying product adoption intentions from social media data [87]. MVL is now widely used in many various domains and research endeavors.

Chapter 3

Enhancing Robustness and Efficiency of Least Square Twin SVM via Granular Computing

This chapter introduces two novel models for binary classification problems in both linear and non-linear spaces: the granular ball least square twin support vector machine (GBLSTSVM) and the large-scale granular ball least square twin support vector machine (LS-GBLSTSVM). Additionally, the algorithms for these models are presented, and the computational complexity of the proposed models is thoroughly discussed. We also provide a comprehensive experimental and statistical analyses of the proposed models.

3.1 The Proposed Granular Ball Least Square Twin Support Vector Machine

In this section, we introduce a novel GBLSTSVM to tackle the binary classification problem. We propose the use of granular balls that encompass either the complete sample space or a fraction of it during the training process. These granular balls, derived from the training dataset, are coarse and represent only a small fraction of the total training data points. This coarse nature renders our proposed model less susceptible to noise and outliers.

By leveraging the granular balls, we aim to generate separating hyperplanes that are nonparallel and can effectively classify the original data points. In the construction of optimal separating hyperplanes, we aim to utilize the maximum information stored in all the training data points while simultaneously decreasing the data points required to find optimal separating hyperplanes.

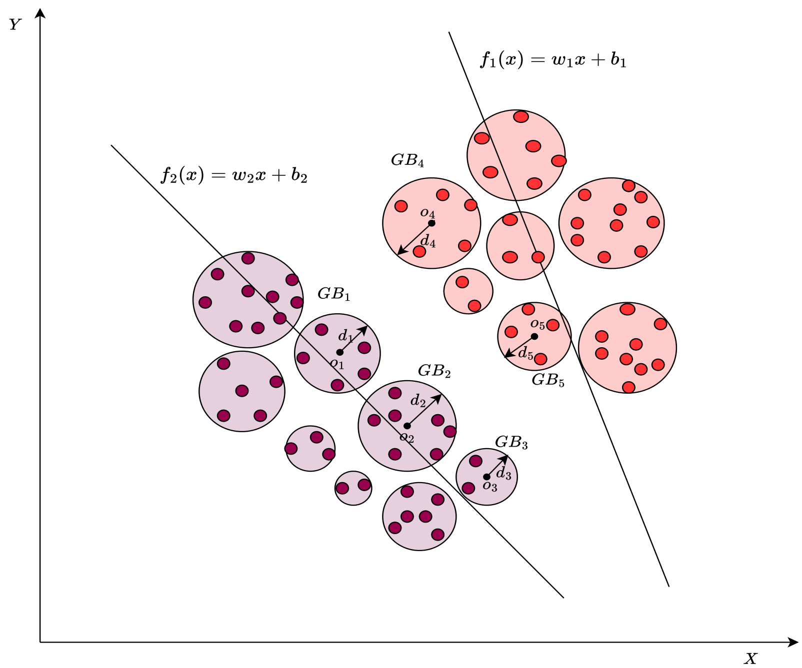


Figure 3.1: Granular ball least square twin support vector machine model having two non-parallel hyperplanes f_1 and f_2.

Hence, we incorporate both the centers and radii of all the granular balls generated through granular computing into the primal formulation of LSTSVM. The integration of granular balls in the training process not only enhances the LSTSVM's robustness against noise and outliers but also significantly reduces the training time, offering enhanced robustness and reduced computational complexity. The geometrical depiction of GBLSTSVM is shown in Fig. 3.1.

3.1.1 Linear GBLSTSVM

The proposed optimization problem of linear GBLSTSVM is given by:

$$\begin{aligned} \min_{w_1,b_1,q_1} \quad & \frac{1}{2}||Cw_1 + e_1b_1||^2 + \frac{c_1}{2}||q_1||^2 \\ \text{s.t.} \quad & -(Dw_1 + e_2b_1) + q_1 = e_2 + R^- \end{aligned} \tag{3.1}$$

and

$$\begin{aligned} \min_{w_2,b_2,q_2} \quad & \frac{1}{2}||Dw_2 + e_2b_2||^2 + \frac{c_2}{2}||q_2||^2 \\ \text{s.t.} \quad & (Cw_2 + e_1b_2) + q_2 = e_1 + R^+, \end{aligned} \tag{3.2}$$

where q_1 and q_2 are slack variables and c_1 and c_2 are tunable parameters. $\begin{bmatrix} w_1; & b_1 \end{bmatrix}$ and $\begin{bmatrix} w_2; & b_2 \end{bmatrix}$ are the hyperplane parameters. In equations (3.1) and (3.2), the incorporation of centers and radii of granular balls is represented using matrices C and D, along with column vectors R^+ and R^-, respectively. To solve (3.1), we substitute the equality constraint into the primal problem

$$\min_{w_1,b_1} \frac{1}{2}||Cw_1 + e_1b_1||^2 + \frac{c_1}{2}||Dw_1 + e_2b_1 + e_2 + R^-||^2.$$

Taking gradient with respect to w_1 and b_1 and equating to 0, we get

$$C'(Cw_1 + e_1b_1) + c_1D'(Dw_1 + e_2b_1 + e_2 + R^-) = 0$$

and

$$e_1'(Cw_1 + e_1b_1) + c_1e_2'(Dw_1 + e_2b_1 + e_2 + R^-) = 0.$$

Converting the system of linear equations into matrix form and solving for w_1 and b_1 we get

$$\begin{bmatrix} D'D & D'e_2 \\ e_2'D & m_2 \end{bmatrix} \begin{bmatrix} w_1 \\ b_1 \end{bmatrix} + \frac{1}{c_1} \begin{bmatrix} C'C & C'e_1 \\ e_1'C & m_1 \end{bmatrix} \begin{bmatrix} w_1 \\ b_1 \end{bmatrix} + \begin{bmatrix} D'e_2 + D'R^- \\ m_2 + e_2'R^- \end{bmatrix} = 0,$$

where $e_1'e_1 = m_1$ and $e_2'e_2 = m_2$.

$$\implies \begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = - \begin{bmatrix} D'D + \frac{1}{c_1}C'C & D'e_2 + \frac{1}{c_1}C'e_1 \\ e_2'D + \frac{1}{c_1}e_1'C & m_2 + \frac{1}{c_1}m_1 \end{bmatrix}^{-1} \begin{bmatrix} D'e_2 + D'R^- \\ m_2 + e_2'R^- \end{bmatrix},$$

$$\implies \begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = - \left[\begin{bmatrix} D' \\ e_2' \end{bmatrix} \begin{bmatrix} D & e_2 \end{bmatrix} + \frac{1}{c_1} \begin{bmatrix} C' \\ e_1' \end{bmatrix} \begin{bmatrix} C & e_1 \end{bmatrix} \right]^{-1} \begin{bmatrix} D' & D' \\ e_2' & e_2' \end{bmatrix} \begin{bmatrix} e_2 \\ R^- \end{bmatrix},$$

$$\implies \begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = - \left[F'F + \frac{1}{c_1}E'E \right]^{-1} \overline{F}\overline{e_2}, \tag{3.3}$$

where

$$E = \begin{bmatrix} C & e_1 \end{bmatrix}, \quad F = \begin{bmatrix} D & e_2 \end{bmatrix}, \quad \overline{F} = \begin{bmatrix} D' & D' \\ {e_2}' & {e_2}' \end{bmatrix}, \text{and } \overline{e_2} = \begin{bmatrix} e_2 \\ R^- \end{bmatrix}.$$

Solving (3.2) in a similar way, we get

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = \left[E'E + \frac{1}{c_2}F'F \right]^{-1} \begin{bmatrix} C'e_1 + C'R^+ \\ m_1 + e_1'R^+ \end{bmatrix},$$

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = \left[E'E + \frac{1}{c_2}F'F \right]^{-1} \overline{E}\overline{e_1}, \tag{3.4}$$

where $\overline{E} = \begin{bmatrix} C' & C' \\ e_1' & e_1' \end{bmatrix}$ and $\overline{e_1} = \begin{bmatrix} e_1 \\ R^+ \end{bmatrix}$.

Once the optimal values of w_1, b_1 and w_2, b_2 are obtained. The categorization of a new input data point $\mathrm{x} \in \mathbb{R}^{1\times N}$ into either the $+1$(class-1) or -1(class-2) class can be determined as follows:

$$\text{Class}(\mathrm{x}) = \underset{i\in\{1,2\}}{\operatorname{argmin}}\Big(\frac{\|w_i\mathrm{x}+b_i\|}{\|w_i\|}\Big). \tag{3.5}$$

3.1.2 Nonlinear GBLSTSVM

To generalize our proposed model to the nonlinear case, we introduce the map $x^\phi = \phi(x) : \mathbb{R}^N \to \mathbb{H}$, where $\mathbb{H}$ represents a Hilbert space. We define $T^\phi = \{x^\phi : x \in T\}$, where T denotes the training dataset. The granular balls that are generated on the set T^ϕ are denoted by $\mathbb{G}^\phi = \{((o_1{}^\phi, d_1{}^\phi), y_1), ((o_2{}^\phi, d_2{}^\phi), y_2), \cdots, ((o_k{}^\phi, d_k{}^\phi), y_k)\}$, where k represents the number of generated granular balls. Let the matrices C^ϕ and D^ϕ, along with column vectors $R_\phi{}^+$ and $R_\phi{}^-$, represent the features of the centers and radii of the granular balls belonging to the positive and negative class, respectively.

The optimization problem for nonlinear GBLSTSVM is given as:

$$\begin{aligned} \min_{w_1,b_1,q_1} \quad & \frac{1}{2}||C^\phi w_1 + e_1 b_1||^2 + \frac{c_1}{2}||q_1||^2, \\ \text{s.t.} \quad & -(D^\phi w_1 + e_2 b_1) + q_1 = e_2 + R_\phi{}^-, \end{aligned} \tag{3.6}$$

and

$$\begin{aligned} \min_{w_2,b_2,q_2} \quad & \frac{1}{2}||D^\phi w_2 + e_2 b_2||^2 + \frac{c_2}{2}||q_2||^2, \\ \text{s.t.} \quad & (C^\phi w_2 + e_1 b_2) + q_2 = e_1 + R_\phi{}^+. \end{aligned} \tag{3.7}$$

The solutions of (3.6) and (3.7) can be derived similarly as in the linear case. The solutions are:

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = -\left[H'H + \tfrac{1}{c_1}G'G\right]^{-1}\overline{H}\overline{e_2},$$

and

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = \left[G'G + \tfrac{1}{c_2}H'H\right]^{-1}\overline{G}\overline{e_1},$$

where

$$\begin{aligned} H &= \begin{bmatrix} D^\phi & e_2 \end{bmatrix}, & G &= \begin{bmatrix} C^\phi & e_1 \end{bmatrix}, \\ \overline{H} &= \begin{bmatrix} D^{\phi'} & D^{\phi'} \\ e_2{}' & e_2{}' \end{bmatrix}, & \overline{G} &= \begin{bmatrix} C^{\phi'} & C^{\phi'} \\ e_1{}' & e_1{}' \end{bmatrix}, \\ \overline{e_1} &= \begin{bmatrix} e_1 \\ R_\phi{}^+ \end{bmatrix}, & \overline{e_2} &= \begin{bmatrix} e_2 \\ R_\phi{}^- \end{bmatrix}. \end{aligned}$$

The classification of data points to class $+1$ or -1 is done similarly to the linear case of the GBLSTSVM model.

3.2 The Proposed Large Scale Granular Ball Least Square Twin Support Vector Machine

Granular computing significantly reduces the number of training instances, leading to a substantial reduction in computational requirements. However, the scalability of the GBLSTSVM may decrease when confronted with large datasets due to its reliance on matrix inversion for solving the system of linear equations. Additionally, like LSTSVM, GBLSTSVM lacks the SRM principle. To address these issues, we introduce a regularization term into the primal formulation of GBLSTSVM. This inclusion results in an additional equality constraint in the primal formulation, effectively eliminating the necessity for matrix inversions in obtaining optimal nonparallel hyperplanes in GBLSTSVM. This removal of matrix inversions significantly reduces the computational complexity of LS-GBLSTSVM, making it well-suited for handling large datasets. Moreover, the integration of the regularization terms implements the SRM principle in GBLSTSVM. Now, we give the formulation of linear LS-GBLSTSVM.

3.2.1 Linear LS-GBLSTSVM

The optimized primal problem of linear LS-GBLSTSVM is given by:

$$\begin{aligned}
\min_{w_1,b_1,q_1,\eta_1} \quad & \frac{c_3}{2}(||w_1||^2 + b_1^2) + \frac{1}{2}||\eta_1||^2 + \frac{c_1}{2}||q_1||^2, \\
\text{s.t.} \quad & \eta_1 = Cw_1 + e_1 b_1, \\
& -(Dw_1 + e_2 b_1) + q_1 = e_2 + R^-,
\end{aligned} \tag{3.8}$$

and

$$\begin{aligned}
\min_{w_2,b_2,q_2,\eta_1} \quad & \frac{c_4}{2}(||w_2||^2 + b_2^2) + \frac{1}{2}||\eta_2||^2 + \frac{c_2}{2}||q_2||^2, \\
\text{s.t.} \quad & \eta_2 = Dw_2 + e_2 b_2, \\
& (Cw_2 + e_1 b_2) + q_2 = e_1 + R^+.
\end{aligned} \tag{3.9}$$

Introducing Lagrange multipliers $\alpha \in \mathbb{R}^{k_1 \times 1}$ and $\beta \in \mathbb{R}^{k_2 \times 1}$ in (3.8), we get

$$L = \frac{c_3}{2}(||w_1||^2 + b_1^2) + \frac{1}{2}||\eta_1||^2 + \frac{c_1}{2}||q_1||^2 + \alpha'(\eta_1 - Cw_1 - e_1 b_1)$$
$$+ \beta'(-(Dw_1 + e_2 b_1) - e_2 - R^- + q_1). \tag{3.10}$$

Applying the K.K.T. necessary and sufficient conditions for (3.10) we obtain the following:

$$\frac{\partial L}{\partial w_1} = c_3 w_1 - C'\alpha - D'\beta = 0, \tag{3.11}$$
$$\frac{\partial L}{\partial b_1} = c_3 b_1 - e_1'\alpha - e_2'\beta = 0, \tag{3.12}$$
$$\frac{\partial L}{\partial q_1} = c_1 q_1 + e_2'\beta = 0, \tag{3.13}$$
$$\frac{\partial L}{\partial \eta_1} = \eta_1 + \alpha = 0, \tag{3.14}$$
$$\eta_1 = Cw_1 + e_1 b_1,$$
$$-(Dw_1 + e_2 b_1) + q_1 = e_2 + R^-.$$

From (3.11) and (3.12), we get

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = \frac{1}{c_3} \begin{bmatrix} C' & D' \\ e_1' & e_2' \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix}. \tag{3.15}$$

Substituting (3.13), (3.14), and (3.15) in (3.10) and simplifying, we get the dual of (3.8):

$$\max_{\alpha,\beta} -\frac{1}{2}\begin{pmatrix} \alpha' & \beta' \end{pmatrix} Q_1 \begin{pmatrix} \alpha' & \beta' \end{pmatrix}' - c_3\beta'(e_2 + R^-)$$
$$\text{where } Q_1 = \begin{bmatrix} CC' + c_3 I_1 & CD' \\ DC' & DD' + \frac{c_3}{c_1} I_2 \end{bmatrix} + E. \tag{3.16}$$

Here, E is the matrix of ones of appropriate dimensions, and I_1 and I_2 are the identity matrices. Similarly, the Wolfe Dual of (3.9) is:

$$\max_{\lambda,\theta} -\frac{1}{2}\begin{pmatrix} \lambda' & \theta' \end{pmatrix} Q_2 \begin{pmatrix} \lambda' & \theta' \end{pmatrix}' - c_4\theta'(e_1 + R^+),$$
$$\text{where } Q_2 = \begin{bmatrix} DD' + c_4 I_2 & DC' \\ CD' & CC' + \frac{c_4}{c_2} I_1 \end{bmatrix} + E. \tag{3.17}$$

Then w_2, b_2 is given by:

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = -\frac{1}{c_4} \begin{bmatrix} D' & C' \\ e_2' & e_1' \end{bmatrix} \begin{bmatrix} \lambda \\ \theta \end{bmatrix}. \tag{3.18}$$

The categorization of a new input data point $x \in \mathbb{R}^{1\times N}$ into either the $+1$(class-1) or -1(class-2) class can be determined as follows:

$$\text{Class}(x) = \underset{i\in\{1,2\}}{\text{argmin}}\Big(\frac{\|w_i x + b_i\|}{\|w_i\|}\Big). \tag{3.19}$$

3.2.2 Nonlinear LS-GBLSTSVM

The optimization problem of nonlinear LS-GBLSTSVM is given as follows:

$$\begin{aligned} \min_{w_1,b_1,q_1,\eta_1} \quad & \frac{c_3}{2}(||w_1||^2 + b_1^2) + \frac{1}{2}||\eta_1||^2 + \frac{c_1}{2}||q_1||^2, \\ \text{s.t.} \quad & \eta_1 = C^{\phi}w_1 + e_1 b_1, \\ & -(D^{\phi}w_1 + e_2 b_1) + q_1 = e_2 + {R_{\phi}}^{-}, \end{aligned} \tag{3.20}$$

and

$$\begin{aligned} \min_{w_2,b_2,q_2,\eta_1} \quad & \frac{c_4}{2}(||w_2||^2 + b_2^2) + \frac{1}{2}||\eta_2||^2 + \frac{c_2}{2}||q_2||^2, \\ \text{s.t.} \quad & \eta_2 = D^{\phi}w_2 + e_2 b_2, \\ & (C^{\phi}w_2 + e_1 b_2) + q_2 = e_1 + {R_{\phi}}^{+}. \end{aligned} \tag{3.21}$$

Calculating Lagrangian as in the Linear LS-GBLSTSVM, we get

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = \frac{1}{c_3}\begin{bmatrix} C^{\phi'} & D^{\phi'} \\ {e_1}' & {e_2}' \end{bmatrix}\begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

The dual of the optimization problem (3.20) and (3.21) are,

$$\begin{aligned} & \max_{\alpha,\beta} -\frac{1}{2}\begin{pmatrix}\alpha' & \beta'\end{pmatrix} Q_1 \begin{pmatrix}\alpha' & \beta'\end{pmatrix}' - c_3\beta'(e_2 + {R_{\phi}}^{-}), \\ & \text{where, } Q_1 = \begin{bmatrix} C^{\phi}C^{\phi'} + c_3 I_1 & C^{\phi}D^{\phi'} \\ D^{\phi}C^{\phi'} & D^{\phi}D^{\phi'} + \frac{c_3}{c_1}I_2 \end{bmatrix} + E, \end{aligned} \tag{3.22}$$

and

$$\begin{aligned} & \max_{\lambda,\theta} -\frac{1}{2}\begin{pmatrix}\lambda' & \theta'\end{pmatrix} Q_2 \begin{pmatrix}\lambda' & \theta'\end{pmatrix}' - c_4\theta'(e_1 + {R_{\phi}}^{+}), \\ & \text{where, } Q_2 = \begin{bmatrix} D^{\phi}D^{\phi'} + c_4 I_2 & D^{\phi}C^{\phi'} \\ C^{\phi}D^{\phi'} & C^{\phi}C^{\phi'} + \frac{c_4}{c_2}I_1 \end{bmatrix} + E. \end{aligned}$$

Then w_2, b_2 is given by:

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = -\frac{1}{c_4}\begin{bmatrix} D^{\phi'} & C^{\phi'} \\ {e_2}' & {e_1}' \end{bmatrix}\begin{bmatrix} \lambda \\ \theta \end{bmatrix}$$

To solve the optimization problem of types (3.16), we use Sequential Minimal Optimization (SMO) [88]. The classification of test data points to class $+1$ or -1 is done in the same manner as in the linear case of the LS-GBLSTSVM model. The algorithm of the proposed GBLSTSVM and LS-GBLSTSVM model is given in Algorithm 1.

Algorithm 1 Linear GBLSTSVM and LS-GBLSTSVM Model Algorithm

1: Initialize GB as the entire dataset T and set G as an empty collection: $GB = T$, $G = \{\}$.
2: Initialize $Object$ as a collection containing GB: $Object = \{GB\}$.
3: **for** $j = 1$ to $|Object|$ **do**
4:     **if** $pur(GB_j) < \rho$ **then**
5:         Split GB_j into GB_{j1} and GB_{j2} using 2-means clustering.
6:         Update $Object$ with the newly formed granular balls: $Object \leftarrow GB_{j1}, GB_{j2}$.
7:     **else**
8:         Calculate center o_j and radius d_j of GB_j:
9:         $o_j = \frac{1}{n_j}\sum_{i=1}^{n_j} x_i$, where $x_i \in GB_j$ and n_j is the number of samples in GB_j.
10:         $d_j = \frac{1}{n_j}\sum_{i=1}^{n_j} ||x_i - o_j||$.
11:         Assign label y_j to GB_j based on the majority class samples within GB_j.
12:         Add $GB_j = \{((o_j, d_j), y_j)\}$ to G.
13:     **end if**
14: **end for**
15: **if** $Object \neq \{\}$ **then**
16:     Repeat steps 3-14 for further splitting.
17: **end if**
18: Generate granular balls: $G = \{((o_j, d_j), y_j),\ \ j = 1, 2, \ldots, k\}$, where k is the number of granular balls.
19: For GBLSTSVM, compute w_1, b_1, w_2, and b_2 using (3.3) and (3.4) and for LS-GBLSTSVM, solve (3.16) and (3.17) to obtain α, β, λ, and θ, then compute w_1, b_1, w_2, and b_2 using (3.15) and (3.18).
20: Classify testing samples into class $+1$ or -1 using (3.5) or (3.19).

3.3 Computational Complexity

We initiate the computational analysis by treating the training dataset T as the initial granular ball set (GB). This set GB undergoes a binary split using the 2-means clustering algorithm, initially resulting in computational complexity of $\mathcal{O}(2m)$, where m is the total number of training data points. In subsequent iterations, if both resultant granular balls remain impure, they are further divided into four granular balls, maintaining a maximum computational complexity of $\mathcal{O}(2m)$ per iteration. This iterative process continues for a total of ω iterations. Consequently, the overall computational complexity of generating granular balls is approximately $\mathcal{O}(2m\omega)$ or less, depending on the purity of the generated granular balls and the number of iterations required.

Suppose that m_1 is the number of $+1$ labeled data samples and m_2 is the number of -1 labeled data samples with $m = m_1 + m_2$. The LSTSVM model requires the calculation of two matrix inverses of order $(m + 1)$. However, using the Sherman-Morrison-Woodbury (SMW) formula [89], the calculation involves solving three inverses of reduced dimensions. Therefore, in the LSTSVM model, the time complexity includes two inversions of size $\mathcal{O}({m_1}^3)$ and one inversion of size $\mathcal{O}({m_2}^3)$ if $m_1 < m_2$. Conversely, if $m_1 \geq m_2$, the complexity involves two inversions of size $\mathcal{O}({m_2}^3)$ and one inversion of size $\mathcal{O}({m_1}^3)$. GBLSTSVM computes the inverses of two matrices with order $(k+1)$, where k represents the total number of granular balls generated on a training dataset T. Hence, the total time complexity of the GBLSTSVM model is approximately less than or equal to $\mathcal{O}(2m\omega) + \mathcal{O}(k^3)$. Given that ω represents the number of iterations, it follows that ω is considerably smaller than m, and also, k is significantly less than m. Thus, $\mathcal{O}(2m\omega)+\mathcal{O}(k^3) \ll \mathcal{O}({m_1}^3)+\mathcal{O}({m_2}^3)$. Hence, the computational complexity of GBLSTSVM is substantially lower than that of LSTSVM.

The computational complexity of the SMO algorithm is $\mathcal{O}(k)$ to $\mathcal{O}(k^{2.2})$. Therefore, the complexity of each optimization problem in the LS-GBLSTSVM model falls approximately between $\mathcal{O}(2m\omega) + \mathcal{O}(k)$ and $\mathcal{O}(2m\omega) + \mathcal{O}(k^{2.2}) \ll \mathcal{O}({m_1}^3)+\mathcal{O}({m_2}^3)$. Therefore, the computational complexity of LS-GBLSTSVM is considerably lower than that of LSTSVM.

3.4 Experimental Results and Discussions

In this section, we assess the efficacy of the proposed GBLSTSVM and LS-GBLSTSVM models. We evaluate their performance against LSTSVM [10] and various other baseline models over UCI [70] and KEEL [71] benchmark datasets with and without label noise to the ensure comprehensive testing. Furthermore, we conduct experiments on NDC datasets [72]. Moreover, we provide a sensitivity analysis of the hyperparameters and granular ball computing parameters.

3.4.1 Experimental Setup

To evaluate the performance of the GBLSTSVM and LS-GBLSTSVM models, a series of experiments are conducted. These experiments are carried out on a PC with an Intel(R) Xeon(R) Gold 6226R processor running at 2.90GHz and 128 GB of RAM. The PC is operating on Windows 11 and utilizes Python 3.11. To solve the dual of QPP in GBSVM, the "QP solvers" function from the CVXOPT package is employed. The dataset is randomly split, with 70% samples are used for training and 30% are for testing purposes. The hyperparameters are tuned using the grid search method and five-fold cross-validation.The hyperparameters c_i $(i = 1, 2, 3, 4)$ were tuned within the range $\{10^{-5}, 10^{-4}, \cdots, 10^5\}$. For the nonlinear case, a Gaussian kernel is utilized, defined as $K(x_i, x_j) = \exp(\frac{-1}{2\sigma^2}||x_i - x_j||^2)$, where σ varied within the range $\{2^{-5}, 2^{-4}, \cdots, 2^5\}$. In the proposed LS-GBLSTSVM model, the values of c_1 and c_2 are set to be equal, as well as the values of c_3 and c_4, for both linear and nonlinear cases.

3.4.2 Experiments on Real World UCI and KEEL Datasets on Linear Kernel

In this subsection, we conduct extensive statistical analyses to compare the proposed GBLSTSVM and LS-GBLSTSVM models with LSTSVM [10] along with several other baseline models, namely SVM [5], TSVM [9], and GBSVM [63]. To solve the optimization problem associated with GBSVM, we employ the PSO algorithm. Our experimental investigation encompasses diverse scenarios, encompassing both linear and nonlinear cases, and involves meticulous numerical experimentation.

We conduct experiments on 34 UCI [70] and KEEL [71] benchmark

Table 3.1: Average accuracy and average rank of the baseline models and the proposed models over UCI and KEEL datasets with Linear kernel.

	Noise	SVM [5]	TSVM [9]	GBSVM [63]	LSTSVM [10]	GBLSTSVM†	LS-GBLSTSVM†
Average ACC	0%	81.58	71.52	73.71	86.67	**88.26**	86.79
	5%	80.65	81.26	77.25	85.33	**87.50**	85.59
	10%	81.70	82.83	76.63	84.82	**87.47**	85.50
	15%	80.03	80.17	73.89	84.12	**86.04**	84.93
	20%	80.21	80.55	75.76	83.79	**86.09**	83.85
Average Rank	0%	3.88	5.26	5.15	2.59	**1.62**	2.50
	5%	4.46	4.37	4.97	2.94	**1.71**	2.56
	10%	4.40	3.65	4.99	3.10	**1.93**	2.94
	15%	4.24	4.22	5.29	2.74	**1.68**	2.84
	20%	4.29	4.19	5.16	2.72	**1.71**	2.93

† represents the proposed model.

datasets. Table 3.12 shows the detailed experimental results of every model over each dataset. All the experimental results discussed in this subsection are obtained at a 0% noise level for both linear and Gaussian kernels. The average accuracy (ACC) and average rank of the linear case are presented in Table 3.1. The average ACC of the GBLSTSVM model is 88.26%, while the LS-GBLSTSVM model achieves an average ACC of 86.79%. On the other hand, the average ACC of the SVM, TSVM, GBSVM, and LSTSVM models are 81.58%, 71.52%, 73.71%, and 86.67%, respectively. In terms of average ACC, our proposed GBLSTSVM and LS-GBLSTSVM models outperform the baseline SVM, TSVM, GBSVM, and LSTSVM models. To further evaluate the performance of our proposed models, we employ the ranking method. In this method, each model is assigned a rank for each dataset, with the best-performing model receiving the lowest rank and the worst-performing model receiving the highest rank. The average rank of a model is calculated as the average of its ranks across all datasets. If we consider a set of M datasets, where l models are evaluated on each dataset, we can represent the position of the s^{th} model on the t^{th} dataset as r_s^t. In this case, the average rank of the s^{th} model is calculated as $\mathscr{R}_s = \frac{1}{M}\sum_{t=1}^{M} r_s^t$. The average rank of SVM, TSVM, GBSVM, and LSTSVM models are 3.88, 5.26, 5.15, and 2.59, respectively. On the other hand, the average rank of the proposed GBLSTSVM and LS-GBLSTSVM models are 1.62 and 2.50, respectively. Based on the average rank, our proposed models demonstrate a superior performance compared to the baseline models. This indicates that our proposed models exhibit better generalization ability.

To assess the statistical significance of the proposed models, we employ the Friedman test [90]. The purpose of this test is to assess the presence of significant disparities among the compared models by examining the average ranks assigned to each model. By evaluating the rankings, we can determine if there are statistically significant differences among the given models. The null hypothesis in this test assumes that all models have the same average rank, indicating an equivalent level of performance. The Friedman test follows the chi-squared distribution (χ_F^2) with $(l-1)$ degrees of freedom and is given by $\chi_F^2 = \frac{12M}{l(l+1)}\left[\sum_s \mathscr{R}_s{}^2 - \frac{l(l+1)^2}{4}\right]$. The Friedman statistic F_F is given by $F_F = \frac{(M-1)\chi_F^2}{M(l-1)-\chi_F^2}$, where, F-distribution has $(l-1)$ and $(l-1)\times(M-1)$ degrees of freedom. For $l=6$ and $M=34$, we get $\chi_F^2 = 110.03$ and $F_F = 60.54$ at 5% level significance. From the statistical F-distribution table, we find that $F_F(5, 165) = 2.2689$. Since $60.54 > 2.2689$, we reject the null hypothesis, indicating a significant statistical difference among the compared models. To

Table 3.2: Wilcoxon-signed rank test of the baseline models w.r.t. the proposed GBLSTSVM over UCI and KEEL data with the Linear kernel.

Model	$\mathscr{R}+$	$\mathscr{R}-$	p-value	Null Hypothesis
SVM [5]	465	0	0.000001819	Rejected
TSVM [9]	496	0	0.00000123	Rejected
GBSVM [63]	561	0	0.0000005639	Rejected
LSTSVM [10]	294.5	30.5	0.0004013	Rejected

Table 3.3: Wilcoxon-signed rank test of the baseline models w.r.t. the proposed LS-GBLSTSVM over UCI and KEEL datasets with Linear kernel.

Model	$\mathscr{R}+$	$\mathscr{R}-$	p-value	Null Hypothesis
SVM [5]	452	13	0.000006632	Rejected
TSVM [9]	489	7	0.000002435	Rejected
GBSVM [63]	527	1	0.0000009169	Rejected
LSTSVM [10]	221.5	156.5	0.4419	Not Rejected

further establish the statistical significance of our proposed GBLSTSVM and LS-GBLSTSVM models with the baseline models, we conduct the Wilcoxon signed rank test [90]. This test calculates the differences in accuracy between pairs of models on each dataset. These differences are then ranked in ascending order based on their absolute values, with tied ranks being averaged. Subsequently, the sum of positive ranks ($\mathscr{R}+$) and the sum of negative ranks ($\mathscr{R}-$) are computed. The null hypothesis in this test typically assumes that there is no significant difference between the performances of the models, meaning that the median difference in accuracy is zero. However, if the difference be-

tween $\mathscr{R}+$ and $\mathscr{R}-$ is sufficiently large, indicating a consistent preference for one model over the other across the datasets. If the resulting p-value from the test is less than 0.05, then the null hypothesis is rejected. The rejection of the null hypothesis signifies that there exists a statistically significant difference in performance between the compared models. Table 3.2 presents the results, demonstrating that our proposed GBLSTSVM model outperforms the baseline SVM, TSVM, GBSVM, and LSTSVM models. Furthermore, Table 3.3 illustrates that the proposed LS-GBLSTSVM model exhibits superior performance compared to the SVM, TSVM, and GBSVM models. The Wilcoxon test strongly suggests that the proposed GBLSTSVM and LS-GBLSTSM models possess a comprehensive statistical advantage over the baseline models.
Moreover, we employ a pairwise win-tie-loss sign test. This test is conducted under the assumption that both models are equal and each model wins on $\frac{M}{2}$ datasets, where M denotes the total number of datasets. To establish statistical significance, the model must win on approximately $\frac{M}{2} + 1.96\frac{\sqrt{M}}{2}$ datasets over the other model. In cases where there is an even number of ties between the compared models, these ties are evenly distributed between the models. However, if the number of ties is odd, one tie is disregarded, and the remaining ties are divided among the specified models. In our case, with $M = 34$, if one of the models achieves a minimum of 22.71 wins, it indicates a significant distinction between the models. The results presented in Table 3.4 clearly show that our proposed models have outperformed the baseline models in the majority of the UCI and KEEL datasets.

Table 3.4: Pairwise win-tie-loss test of proposed GBLSTSVM and baseline models on UCI and KEEL datasets with linear kernel

	SVM [5]	TSVM [9]	GBSVM [63]	LSTSVM [10]	GBLSTSVM
TSVM [9]	[3, 1, 30]				
GBSVM [63]	[5, 3, 26]	[19, 0, 15]			
LSTSVM [10]	[25, 4, 5]	[29, 5, 0]	[31, 1, 2]		
GBLSTSVM	[30, 4, 0]	[31, 3, 0]	[33, 1, 0]	[21, 9, 4]	
LS-GBLSTSVM	[27, 4, 3]	[30, 3, 1]	[31, 1, 2]	[15, 7, 12]	[3, 11, 20]

wherein $\begin{bmatrix} x & y & z \end{bmatrix}$, x signifies no. of wins, y no. of draws, and z no. of losses.

3.4.3 Experiments on Real World UCI and KEEL Datasets on Gaussian Kernel

Table ?? demonstrates that our proposed GBLSTSVM and LS-GBLSTSVM models outperform the baseline models in Gaussian kernel space in most of the datasets. Table 3.5 presents the average accuracy (ACC) and average rank of the proposed GBLSTSVM and LS-GBLSTSVM models, as well as the baseline models using the Gaussian kernel. Our proposed GBSLSTVM and LS-GBLSTSVM models achieve an average accuracy of 83.64% and 84.55%, respectively, which is superior to the baseline models. Additionally, the average rank of our proposed GBLSTSVM and LS-GBLSTSVM models is 2.90 and 2.84, respectively, indicating a lower rank compared to the baseline models. This suggests that our proposed models exhibit better generalization ability than the baseline models.

Table 3.5: Average accuracy and average rank of the baseline models and the proposed models over UCI and KEEL datasets with Gaussian kernel.

	Noise	SVM [5]	TSVM [9]	GBSVM [63]	LSTSVM [10]	GBLSTSVM†	LS-GBLSTSVM†
Average ACC	0%	75.47	82.40	78.20	76.89	83.64	**84.55**
	5%	76.02	81.07	78.42	75.03	**82.67**	82.62
	10%	75.49	80.00	79.21	74.97	**83.06**	82.68
	15%	77.52	82.05	80.00	75.79	**84.05**	83.15
	20%	76.63	81.93	77.86	77.17	**83.11**	82.65
Average Rank	0%	4.32	3.24	3.99	3.72	2.90	**2.84**
	5%	4.28	3.44	4.04	3.68	2.81	**2.75**
	10%	4.16	3.66	3.93	3.68	**2.69**	2.88
	15%	3.91	3.44	4.28	3.78	**2.49**	3.10
	20%	4.04	3.04	4.49	3.60	**2.85**	2.97

† represents the proposed model.

Table 3.6: Wilcoxon-signed rank test of the baseline models w.r.t. the proposed GBLSTSVM over UCI and KEEL datasets with Gaussian kernel.

Model	$\mathscr{R}+$	$\mathscr{R}-$	p-value	Null Hypothesis
SVM [5]	421	75	0.0007234	Rejected
TSVM [9]	290	238	0.6335	Not Rejected
GBSVM [63]	377	184	0.08628	Not Rejected
LSTSVM [10]	226	50	0.007777	Rejected

Furthermore, we conduct the Friedman test [90] and Wilcoxon signed rank test [90] for the Gaussian kernel. For $l = 6$ and $M = 34$, we obtained $\chi_F^2 = 16.6279$ and $F_F = 3.5777$ at a significance level of 5%. Referring to the statistical F-distribution table, we find that $F_F(5, 165) = 2.2689$. Since

Table 3.7: Wilcoxon-signed rank test of the baseline models w.r.t. the proposed LS-GBLSTSVM over UCI and KEEL datasets with Gaussian kernel.

Model	$\mathscr{R}+$	$\mathscr{R}-$	p-value	Null Hypothesis
SVM [5]	435	93	0.00143	Rejected
TSVM [9]	332	164	0.1018	Not Rejected
GBSVM [63]	409	152	0.02218	Rejected
LSTSVM [10]	334	101	0.01213	Rejected

Table 3.8: Pairwise win-tie-loss test of proposed and baseline models on UCI and KEEL datasets with Gaussian kernel

	SVM [5]	TSVM [9]	GBSVM [63]	LSTSVM [10]	GBLSTSVM
TSVM [9]	[22, 4, 8]				
GBSVM [63]	[17, 1, 16]	[9, 4, 21]			
LSTSVM [10]	[17, 11, 6]	[15, 1, 18]	[18, 0, 16]		
GBLSTSVM	[22, 5, 7]	[18, 2, 14]	[22, 1, 11]	[14, 15, 5]	
LS-GBLSTSVM	[22, 5, 7]	[19, 3, 12]	[21, 1, 12]	[17, 8, 9]	[14, 8, 12]

wherein $[x \quad y \quad z]$, x signifies no. of wins, y no. of draws, and z no. of losses.

$3.5777 > 2.2689$, we reject the null hypothesis. Consequently, there exists a significant statistical difference among the compared models. Moreover, the Wilcoxon signed test presented in Table 3.6 for GBLSTSVM and Table 3.7 for LS-GBLSTSVM demonstrate that our proposed models possess a significant statistical advantage over the baseline models. The pairwise win-tie-loss results presented in Table 3.8 further emphasize the superiority of our proposed models over the baseline models.

3.4.4 Experiments on Real World UCI and KEEL Datasets with Added Label Noise on Linear and Gaussian Kernel

The proposed GBTSVM and LS-GBTSVM models are experimentally evaluated using UCI and KEEL benchmark datasets. To assess their performance, label noise is introduced at varying levels of 5%, 10%, 15%, and 20%. The results, presented in Table 3.12 and Table 3.13 demonstrate the effectiveness of these models compared to baseline models in both linear and nonlinear cases. Throughout the evaluations, the GBLSTSVM and LS-GBLSTSVM models consistently outperformed the baseline models. The proposed GBLSTSVM demonstrates a superior average ACC compared to the baseline models, with an improvement of up to 3% when increasing the label noise from 5% to 20% for the linear kernel. Similarly, our proposed LS-GBLSTSVM has a better average ACC than the baseline models with linear kernel. The average ACC of LS-GBLSTSVM at noise levels of 5%, 10%, 15%, and 20% are 85.59, 85.50, 84.93, and 83.85, respectively, which is higher than all baseline models. Additionally, the proposed models outperform the baseline models in terms of average rank, even when considering various levels of label noise. For the Gaussian kernel, our proposed GBLSTSVM has up to 3% better average ACC, and LS-GBLSTSVM has up to 2% better average ACC compared to baseline models on increasing the levels of label noise from 5% to 20%. Also, our proposed models have a lower average rank than the baseline models in increasingly noisy conditions as well. This can be attributed to the incorporation of granular balls within these models, which exhibit a coarser granularity and possess the ability to mitigate the effects of label noise. The key feature of these granular balls is their strong influence of the majority label within them, effectively reducing the impact of noise points from minority labels on the classification results. This approach significantly enhances the models' resistance to label noise contamination. The consistent superiority of the GBLSTSVM and LS-GBLSTSVM models over the baseline models highlights their potential effectiveness in real-world scenarios where noise is commonly encountered in datasets.

3.4.5 Experiments on NDC Datasets

The previous comprehensive analyses have consistently shown the superior performance of the proposed GBLSTSVM and LS-GBLSTSVM models compared to the baseline models across the majority of UCI and KEEL benchmark datasets. Furthermore, we conduct an experiment using the NDC datasets [72] to highlight the enhanced training speed and scalability of our proposed models. For this, all hyperparameters are set to 10^{-5}, which is the lowest value among the specified ranges. These NDC datasets' sample sizes vary from 10k to 5m with 32 features. The results presented in Table 3.9 show the efficiency and scalability of the proposed GBLSTSVM and LS-GBLSTSVM models. Across the NDC datasets, our models consistently outperform the baseline models in terms of both accuracy and training times, thus confirming their robustness and efficiency, particularly when dealing with large-scale datasets. In the context of ACC, our GBLSTSVM model demonstrates superior accuracy, with an increase of up to 3% when the NDC dataset scale is expanded from 10k to

Table 3.9: Accuracy and time of the proposed GBLSTSVM and LS-GBLSTSVM with baseline models on NDC datasets with Linear kernel.

NDC datasets	SVM [5] ACC(%) Time (s)	TSVM [9] ACC(%) Time (s)	GBSVM [63] ACC(%) Time (s)	LSTSVM [10] ACC(%) Time (s)	GBLSTSVM† ACC(%) Time (s)	LS-GBLSTSVM† ACC(%) Time (s)
NDC 10K	81.64 310.66	80.78 209.79	64.97 1,510.66	83.34 12.02	**85.84** 10.66	83.29 15.55
NDC 50K	80.35 941.11	79.44 816.81	60.57 2,809.49	82.80 54.10	84.84 30.11	**84.85** 39.09
NDC 100K	^{c}	^{c}	^{d}	82.91 70.16	**85.12** 53.47	84.93 109.86
NDC 300K	^{c}	^{c}	^{d}	83.21 124.44	**84.86** 112.41	83.93 159.27
NDC 500K	^{c}	^{c}	^{d}	83.14 199.72	**85.90** 165.98	83.79 184.03
NDC 1m	^{c}	^{c}	^{d}	83.07 301.42	**84.75** 221.76	83.94 265.91
NDC 3m	^{c}	^{c}	^{d}	83.02 357.24	**84.56** 267.65	83.61 291.51
NDC 5m	^{c}	^{c}	^{d}	83.10 406.63	84.30 316.89	**84.99** 499.65

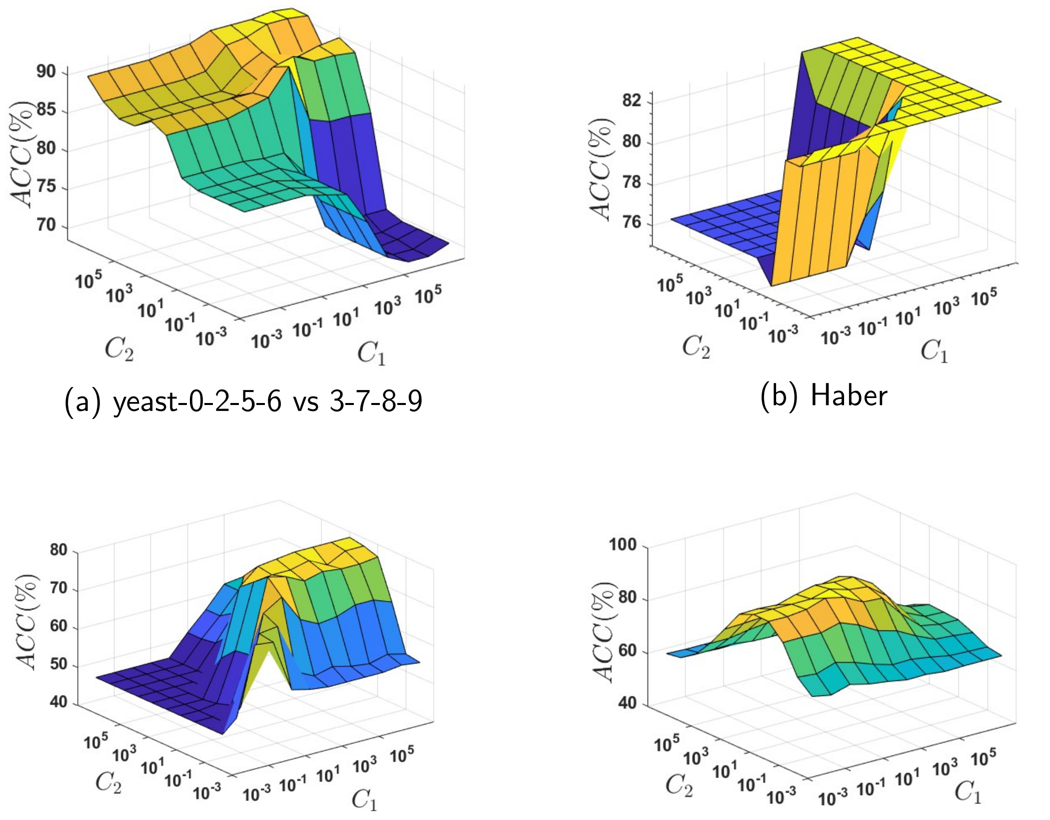
^{c} Terminated because of out of memory.
^{d} Experiment is terminated because of the out of bound issue shown by the PSO algorithm.
† represents the proposed model.

5m. Additionally, GBLSTSVM demonstrates reduced training time across all ranges of NDC datasets compared to LSTSVM. Specifically, for the NDC 5m dataset, the LS-GBLSTSVM model achieves an impressive accuracy of 84.99%, which stands as the highest accuracy. The experimental results demonstrate a significant reduction of 100 to 1000 times in the training duration of the GBLSTSVM and LS-GBLSTSVM models compared to the baseline models. This exceptional decrease in training time can be attributed to the significantly lower count of generated granular balls on a dataset in comparison to the total number of samples.

3.4.6 Sensitivity Analysis of Hyperparameters

To thoroughly understand the subtle effects of the hyperparameters on the model's generalization ability, we systematically explore the hyperparameter space by varying the values of c_1 and c_2. This exploration allows us to identify the configuration that maximizes predictive accuracy and enhances the model's resilience to previously unseen data.

The graphical representations in Figure 3.2 provide visual insights into the impact of parameter tuning on the accuracy (ACC) of our GBLSTSVM model for the linear case. These visuals demonstrate an apparent variation in the model's accuracy across a range of c_1 and c_2 values, highlighting the



(a) yeast-0-2-5-6 vs 3-7-8-9 (b) Haber

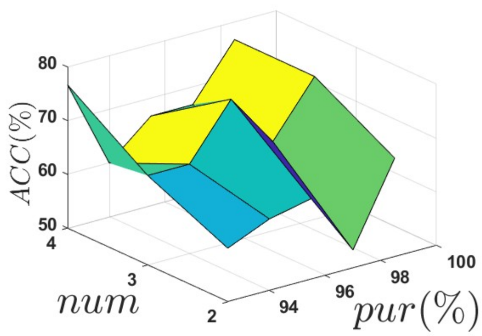
(c) Monks3 (d) Spambase

Figure 3.2: The effect of hyperparameter (c_1, c_2) tuning on the accuracy (ACC) of some UCI and KEEL datasets on the performance of linear GBLSTSVM.

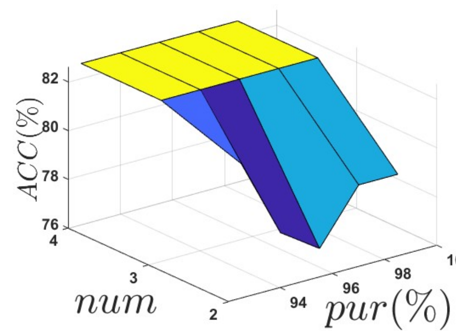
sensitivity of our model's performance to these hyperparameters. In Figure 3.2 (a), it is evident that lower values of c_1 combined with higher values of c_2 result in improved accuracy. Similarly, Figure 3.2 (d) shows that optimal accuracy is achieved when both c_1 and c_2 are set to mid-range values. It has been observed that lower values of c_1 and higher values of c_2 give the best generalization performance.

3.4.7 Sensitivity Analysis of Granular Parameters

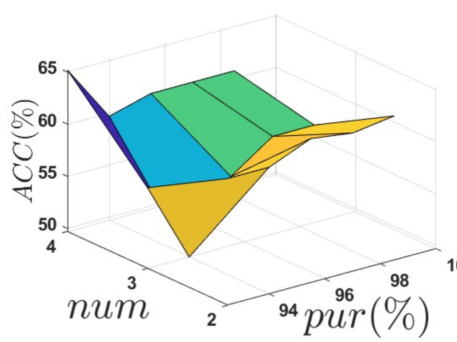
In the context of granular computing, we can ascertain the minimum number of granular balls to be generated on the training dataset T, denoted as num. For our binary classification problem, we establish the minimum value for num at 2. Hence, our goal is to generate at least two granular balls for each dataset. The purity (pur) of a granular ball is a crucial characteristic. By adjusting the purity level of the granular balls, we can simplify the distribution of data points in space and effectively capture the data points distribution using these



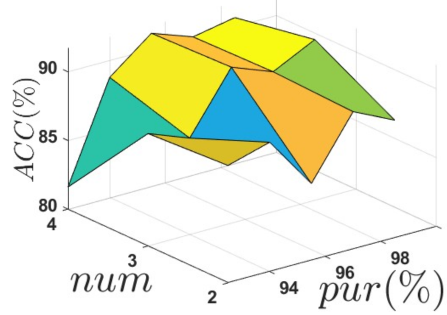
(a) breast cancer wisc prog



(b) haberman



(c) Monks2



(d) Spambase

Figure 3.3: The effect of granular parameter (num, pur) tuning on the accuracy (ACC) of some UCI and KEEL datasets on the performance of linear GBLSTSVM.

Table 3.10: Performance of the proposed Linear GBLSTSVM across varying purities, showcasing the relationship between the number of Granular Balls and the resulting accuracies.

pur	1	0.97	0.94	0.91	0.88	0.85	0.82	0.79
Dataset	ACC $n(GB)$	ACC $n(GB)$	ACC $n(GB)$	ACC $n(GB)$	ACC $n(GB)$	ACC $n(GB)$	ACC $n(GB)$	ACC $n(GB)$
spambase	89.21 400	90.88 316	86.31 277	83.13 201	79.36 180	81.03 134	78.13 107	78.86 99
musk_1	83.22 59	78.32 63	81.12 60	79.72 57	79.72 51	80.42 48	79.72 37	78.52 40
tic_tac_toe	99.65 108	99.65 113	75.35 105	97.22 100	99.65 93	99.65 73	99.65 76	99.65 57
monks_3	80.24 69	83.23 67	81.44 65	79.04 67	80.84 74	80.84 60	82.04 56	79.04 54
vehicle1	83.46 96	84.25 94	84.25 93	83.07 75	76.77 69	82.28 61	84.25 49	79.13 31

Table 3.11: Performance of the proposed Linear LS-GBLSTSVM across varying purities, showcasing the relationship between the number of Granular Balls and the resulting accuracies.

pur	1	0.97	0.94	0.91	0.88	0.85	0.82	0.79
Dataset	ACC $n(GB)$	ACC $n(GB)$	ACC $n(GB)$	ACC $n(GB)$	ACC $n(GB)$	ACC $n(GB)$	ACC $n(GB)$	ACC $n(GB)$
musk_1	79.02 59	75.52 63	79.02 60	74.13 57	77.62 51	72.73 48	78.32 37	76.22 40
tic_tac_toe	97.92 108	100.00 113	90.63 105	95.14 100	97.22 93	98.26 73	99.65 76	91.32 57
monks_3	67.07 69	68.26 67	60.48 65	55.09 67	76.05 74	70.66 60	64.07 56	61.68 54
monks_2	53.04 85	50.28 92	48.07 88	64.14 89	58.01 77	44.75 72	51.38 81	64.09 62
breast_cancer	60.47 41	74.41 40	66.28 39	37.21 40	43.02 36	55.81 35	62.79 31	60.47 24

granular balls. To analyze the impact of num and pur on the generalization performance of GBLSTSVM, we have tuned num within the range of $\{2, 3, 4\}$, and pur within the range of $\{92.5, 94.0, 95.5, 97.0, 98.5\}$. The visual depictions in Figure 3.3 offer valuable visualizations of how this tuning affects the accuracy (ACC) of our GBLSTSVM model. Careful examination of these visuals depicts that there exists an optimal value of num and pur using which our proposed GBLSTSVM mode gives the optimal generalization performance. In the case of Figure 3.3 (a), when both pur and num are increased simultaneously, a significant rise in ACC can be observed. This suggests that the accuracy of our proposed model improves as the purity increases and the minimum number of granular balls generated increases. Similar patterns can be observed in other figures in Figure 3.3. This aligns with the principle of granular computing. As the value of num rises, the minimum count of granular balls required to cover our sample space also increases. With an increase in pur, these granular balls divide even more, resulting in a greater generation of granular balls. This process effectively captures the data patterns, ultimately leading to the best possible generalization performance. Tables 3.10 and 3.11 illustrate the variation in ACC for linear GBLSTSVM and LS-GBLSTSVM model and the number of granular balls generated ($n(GB)$) while tuning purity levels across various UCI and KEEL datasets when num is fixed at 2.

Table 3.12: Performance comparison of the proposed GBLSTSVM and LS-GBLSTSVM along with the baseline models based on classification accuracy using linear kernel for UCI and KEEL datasets.

Model Dataset	Noise	SVM ACC (%) (c_1)	TSVM ACC (%) (c_1, c_2)	GBSVM ACC (%) (c_1)	LSTSVM ACC (%) (c_1, c_2)	GBLSTSVM† ACC (%) (c_1, c_2)	LS-GBLSTSVM† ACC (%) (c_1, c_2)
acute_nephritis	0%	100	100	90.33	100	100	100.00
		(0.1)	(0.01, 0.01)	(100000)	(0.001, 0.001)	(0.00001, 1000)	(0.00001, 0.00001)
	5%	100	$100.00	90.33	100	100	100.00
		(0.1)	(0.1, 1)	(100000)	(0.1, 1)	(0.00001, 0.01)	(0.00001, 0.00001)
	10%	100	100	94.44	100	100	100.00
		(0.1)	(0.1, 1)	(0.00001)	(0.01, 0.01)	(0.00001, 0.00001)	(0.0001, 0.00001)
	15%	100	86.11	83.33	88.89	100	100.00
		(0.1)	(0.1, 0.1)	(10000)	(0.01, 0.01)	(100000, 0.00001)	(0.00001, 0.00001)
	20%	100	97.22	83.33	97.22	100	88.89
		(0.1)	(0.1, 0.1)	(10000)	(0.1, 0.1)	(0.00001, 100000)	(0.00001, 0.0001)
breast_cancer_wisc_prog	0%	75.67	75	73	75	76.67	76.67
		(0.001)	(10, 1)	(10)	(1,1)	(0.00001, 0.00001)	(0.00001, 0.01)
	5%	68.33	67.33	66.67	65	68.33	76.67
		(0.001)	(0.01, 0.1)	(100000)	(10,100)	(100000,100000)	(0.00001, 0.00001)
	10%	72.33	68.33	72.67	76.67	76.67	76.67
		(0.001)	(0.01, 0.1)	(1000)	(0.1, 0.001)	(0.00001,0.00001)	(0.00001, 0.00001)
	15%	73.67	75	69.67	71.66666667	76.67	73.33
		(0.00001)	(1, 0.1)	(1000)	(1, 0.1)	(0.00001,0.00001)	(0.00001, 0.00001)
	20%	75.67	65.00	71.33	76.67	76.67	76.67
		(0.00001)	(0.00001, 0.00001)	(1000)	(0.0001, 0.00001)	(0.00001,10000)	(0.00001, 0.00001)
breast_cancer	0%	72.09	60	62.79	70.93	74.42	73.26
		(0.001)	(0.001, 0.01)	(0.00001)	(0.1, 0.1)	(1000, 0.00001)	(0.1, 0.1)
	5%	70.93	65.12	80	69.77	72.09	74.42
		(0.001)	(10, 100)	(100000)	(0.1, 0.1)	(0.00001, 0.00001)	(0.1, 0.1)
	10%	68.6	65.12	74.42	72.09	74.42	74.42
		(0.001)	(0.000001, 0.000001)	(100)	(1, 1)	(0.00001, 0.00001)	(0.1, 0.1)
	15%	68.6	75.58	54.65	75.58	76.74	72.09
		(0.001)	(0.000001, 0.000001)	(0.000001)	(0.01, 0.01)	(100000, 0.00001)	(10, 0.1)
	20%	65.12	74.42	59.3	75.58	74.42	76.74
		(0.001)	(10, 1)	(1000)	(100, 10)	(100000, 0.00001)	(0.00001, 0.00001)
conn_bench_sonar_mines_rocks	0%	74.60	73.02	41.2	74.60	79.37	74.60
		(0.01)	(10, 1)	(1000)	(1, 1)	(0.00001, 10000)	(1000, 100)
	5%	74.60	77.78	69.68	76.19	80.95	71.43
		(0.01)	(10, 1)	(1000)	(0.1, 0.1)	(100000, 100000)	(1000, 10)
	10%	74.60	77.78	70.68	71.43	79.37	73.02
		(0.001)	(10, 1)	(100000)	(100, 100)	(10000, 100000)	(0.1, 0.1)
	15%	71.43	71.43	68.09	68.25	73.02	68.25
		(0.001)	(0.1, 0.1)	(100000)	(100, 10)	(0.00001, 100000)	(0.00001, 0.00001)
	20%	71.43	66.67	69.84	69.84	76.67	74.60
		(0.001)	(0.1, 0.1)	(100000)	(100, 10)	(0.00001, 0.00001)	(10, 10)
crossplane130	0%	97.24	71.35	100	100.00	100.00	100.00
		(0.001)	(0.00001, 0.00001)	(1)	(0.00001, 0.00001)	(0.00001, 0.00001)	(1, 0.1)
	5%	97.44	100	97.44	100.00	100.00	100.00
		(0.1)	(0.00001, 0.00001)	(0.00001)	(0.001, 0.01)	(0.00001, 0.00001)	(1000, 10)
	10%	100	100	97.44	100.00	100.00	100.00
		(0.001)	(0.00001, 0.00001)	(0.00001)	(0.001, 0.001)	(0.00001, 0.00001)	(0.1, 0.01)
	15%	100	100	100	100.00	100.00	97.44
		(0.001)	(0.00001, 0.00001)	(10)	(0.00001, 0.00001)	(0.00001, 0.00001)	(100, 10)
	20%	97.44	100	97.44	97.44	100.00	100.00
		(0.001)	(10, 1000)	(0.00001)	(0.00001, 0.00001)	(0.00001, 0.00001)	(1, 0.1)
crossplane150	0%	55.56	100.00	72.22	100.00	100.00	97.78
		(0.01)	(0.00001, 0.00001)	(100000)	(0.00001, 0.00001)	(0.00001, 0.00001)	(0.0001, 0.00001)
	5%	55.56	95.56	86.67	95.56	100.00	80.00
		(0.01)	(0.00001, 0.00001)	(0.00001)	(0.00001, 0.00001)	(0.00001, 0.00001)	(0.00001, 0.1)
	10%	53.33	95.56	86.67	75.56	100.00	97.78
		(0.01)	(0.00001, 0.00001)	(100000)	(0.00001, 0.00001)	(0.00001, 0.00001)	(0.0001, 0.00001)
	15%	55.56	62.22	57.78	62.22	64.44	80.00
		(0.1)	(0.00001, 0.00001)	(10)	(0.00001, 0.00001)	(0.00001, 1000)	(0.00001, 0.1)
	20%	53.33	64.44	51.89	53.33	55.56	80.00
		(0.1)	(1, 1)	(100000)	(0.00001, 0.00001)	(0.00001, 100000)	(0.00001, 10)
ecoli-0-1_vs_5	0%	88.61	66.68	77.5	95.83	95.83	97.22
		(0.01)	(0.1, 1)	(1000)	(0.1, 1)	(10,10000)	(1, 0.1)
	5%	88.61	85.83	86.11	95.83	94.44	91.67
		(0.01)	(0.1, 1)	(10)	(0.1, 1)	(0.00001,100000)	(1, 0.1)
	10%	87.22	84.44	96.43	94.44	95.83	95.83
		(0.01)	(0.1, 1)	(10)	(0.1, 1)	(1, 1)	(0.1, 0.001)
	15%	87.22	86.11	79.17	86.11	93.06	88.89
		(0.01)	(0.001, 0.001)	(10)	(0.00001, 0.00001)	(1, 0.1)	(1, 0.1)
	20%	85.83	88.89	88.89	86.11	94.44	91.67
		(0.01)	(0.0001, 0.00001)	(10000)	(0.01, 0.01)	(1,1000)	(1, 10000)
ecoli-0-1-4-6_vs_5	0%	95.81	67.88	94.05	97.62	97.62	94.05
		(0.01)	(1, 1)	(10)	(0.1, 1)	(0.1, 100000)	(0.0001, 0.1)
	5%	97.62	88.81	94.05	96.43	98.81	96.43
		(0.01)	(10, 1)	(10)	(1, 1)	(10, 10)	(0.0001, 0.00001)
	10%	87.62	85.24	69.35	96.43	98.81	98.81
		(0.01)	(0.00001, 0.00001)	(10)	(1, 1)	(0.1, 1000)	(0.0001, 0.00001)
	15%	90.62	85.24	95.24	94.05	94.05	94.05
		(0.01)	(0.1, 0.1)	(10)	(1, 1)	(0.0001, 0.00001)	(0.00001, 0.1)
	20%	94.05	89.05	94.05	97.62	96.43	94.05
		(0.00001)	(1, 1)	(10000)	(1, 1)	(0.1, 0.1)	(0.00001, 0.1)
ecoli-0-1-4-7_vs_2-3-5-6	0%	85.05	68.51	52.48	93.07	91.09	87.13
		(0.01)	(0.1, 1)	(100000)	(1, 10)	(1, 0.1)	(0.0001, 0.00001)
	5%	84.06	87.13	73.27	93.07	91.09	91.09
		(0.01)	(0.1, 0.01)	(10)	(0.1, 1)	(1, 100000)	(10000, 10)
	10%	94.06	91.09	64.36	91.09	90.10	91.09
		(0.01)	(0.00001, 0.00001)	(10)	(0.1, 0.1)	(1, 100000)	(0.0001, 0.00001)
	15%	85.05	74.55	84.16	90.10	93.07	92.08
		(0.01)	(100000, 100)	(10)	(0.1, 0.1)	(10, 100)	(1, 0.1)
	20%	87.13	81.09	74.26	90.10	96.04	87.13
		(0.00001)	(100, 100000)	(10)	(1, 1)	(1, 1)	(1, 10000)

† represents the proposed model.

Table 3.12 (Continued)

Model Dataset	Noise	SVM ACC (%) (c_1)	TSVM ACC (%) (c_1, c_2)	GBSVM ACC (%) (c_1)	LSTSVM ACC (%) (c_1, c_2)	GBLSTSVM† ACC (%) (c_1, c_2)	LS-GBLSTSVM† ACC (%) (c_1, c_2)
ecoli2	0%	89.11	87.13	84.85	87.13	89.11	89.11
		(0.00001)	(1, 1)	(10)	(0.1, 1)	(10, 1000)	(0.1, 0.01)
	5%	86.11	86.14	76.44	86.14	89.11	89.11
		(0.00001)	(0.00001, 0.0001)	(0.00001)	(0.01, 1)	(0.0001, 0.00001)	(10000, 10)
	10%	83.11	86.14	83.66	88.12	90.10	88.12
		(0.00001)	(0.00001, 0.0001)	(10000)	(0.01, 0.01)	(0.00001, 1)	(0.1, 0.0001)
	15%	89.11	85.11	75.32	89.11	90.10	89.11
		(0.00001)	(0.001, 0.00001)	(10)	(0.001, 0.00001)	(0.0001, 1)	(10, 0.1)
	20%	85.11	85.11	85.15	89.11	89.11	90.10
		(0.00001)	(0.001, 0.00001)	(100)	(0.01, 0.00001)	(0.0001, 0.00001)	(100, 0.1)
ecoli3	0%	89.11	87.13	84.85	87.13	89.11	89.11
		(0.00001)	(1, 1)	(10)	(0.1, 1)	(10, 1000)	(0.1, 0.01)
	5%	86.11	86.14	86.44	86.14	89.11	89.11
		(0.00001)	(0.00001, 0.0001)	(0.00001)	(0.01, 0.1)	(0.0001, 0.00001)	(10000, 10)
	10%	89.11	86.14	80.66	88.12	90.10	88.12
		(0.00001)	(0.00001, 0.0001)	(10000)	(0.01, 0.01)	(0.00001, 0.1)	(0.1, 0.0001)
	15%	89.11	85.11	70.32	89.11	90.10	89.11
		(0.00001)	(0.001,0.00001)	(10)	(0.001, 0.00001)	(0.00001, 1)	(10, 0.1)
	20%	85.11	85.11	85.15	89.11	89.11	90.10
		(0.00001)	(0.001, 0.00001)	(100)	(0.01, 0.00001)	(0.00001, 0.00001)	(100, 0.1)
fertility	0%	88	89	85	90.00	90.00	90.00
		(0.00001)	(0.00001, 0.00001)	(100)	(0.00001, 0.00001)	(0.00001, 0.00001)	(0.00001, 0.0001)
	5%	89	90	86.67	90.00	90.00	86.67
		(0.00001)	(0.00001, 0.0001)	(100)	(0.01, 0.1)	(0.00001, 0.00001)	(0.00001, 0.0001)
	10%	90	80	76.67	76.67	90.00	86.67
		(0.00001)	(0.00001, 0.0001)	(10)	(0.001, 0.01)	(0.0001, 0.00001)	(0.00001, 0.0001)
	15%	70	86.67	80	86.67	90.00	90.00
		(0.00001)	(0.01, 0.01)	(1000)	(0.01, 0.01)	(0.00001, 0.00001)	(1, 1000)
	20%	80	86.67	70.67	90.00	90.00	86.67
		(0.00001)	(0.1, 0.1)	(1000)	(1, 0.1)	(0.01, 0.001)	(0.001, 0.01)
glass2	0%	89.77	90.77	88.46	92.31	90.77	90.77
		(0.00001)	(0.00001, 0.00001)	(100000)	(0.01, 0.1)	(0.00001, 0.00001)	(0.00001, 0.1)
	5%	88.77	87.77	90.17	87.69	90.77	90.77
		(0.00001)	(0.001, 0.001)	(10)	(0.01, 0.01)	(0.00001, 0.00001)	(0.00001, 0.1)
	10%	89.77	90.77	90.17	90.77	90.77	90.77
		(0.00001)	(0.001, 0.001)	(10)	(0.001, 0.0001)	(0.00001, 0.00001)	(0.00001, 0.1)
	15%	90.77	90.77	75.38	90.77	90.77	90.77
		(0.00001)	(0.0001, 0.00001)	(100)	(0.01, 0.001)	(0.00001, 0.00001)	(0.00001, 0.1)
	20%	80.76923077	80.76923077	89.23076923	90.77	90.77	90.77
		(0.00001)	(0.0001, 0.00001)	(100)	(0.01, 0.01)	(0.00001, 0.00001)	(0.00001, 0.1)
glass5	0%	92.92	89.00	80.46	95.38	96.92	96.92
		(0.00001)	(0.1, 1)	(100000)	(0.00001, 0.1)	(10000, 1000)	(0.00001, 0.01)
	5%	92.92	92.46	86.15	96.92	96.92	96.92
		(0.00001)	(1, 1)	(10)	(0.1, 0.1)	(0.00001, 0.00001)	(0.00001, 0.01)
	10%	92.92	92.46	96.92	96.92	96.92	96.92
		(0.00001)	(1, 1)	(10)	(0.001, 0.00001)	(0.00001, 0.00001)	(0.00001, 0.01)
	15%	96.92	90.85	94.92	96.92	96.92	96.92
		(0.00001)	(0.0001,0.00001)	(10)	(1, 1)	(0.00001, 0.00001)	(0.00001, 0.01)
	20%	92.92	94.92	91.92	95.38	96.92	96.92
		(0.00001)	(10, 1)	(10)	(100, 10)	(0.00001, 0.00001)	(0.00001, 0.01)
haber	0%	77.17	57.96	77.17	78.26	82.61	78.26
		(0.01)	(0.00001, 0.00001)	(10)	(0.1, 0.1)	(0.00001, 0.00001)	(0.00001, 0.0001)
	5%	76.09	75	77.17	77.17	82.61	81.52
		(0.01)	(1000, 10)	(10)	(0.1, 0.1)	(0.00001, 0.00001)	(10, 0,1)
	10%	77.17	78.26	58.04	77.17	82.61	78.26
		(0.01)	(1000, 10)	(100)	(0.00001, 0.00001)	(0.00001, 0.00001)	(1, 0,1)
	15%	78.26	78.26	59.78	78.26	78.26	77.17
		(0.01)	(0.00001, 0.00001)	(0.00001)	(0.00001, 0.00001)	(0.00001, 0.1)	(0.1, 0.1)
	20%	76.09	73.91	68.48	78.26	82.61	77.17
		(0.01)	(0.00001, 0.00001)	(0.00001)	(0.00001, 0.00001)	(0.00001, 0.00001)	(0.1, 0.1)
haberman_survival	0%	77.17	57.96	78.26	78.26	82.61	81.52
		(0.01)	(0.00001,0.00001)	(10)	(0.1,0.1)	(0.00001,0.00001)	(0.1, 0.1)
	5%	76.09	75	78.26	77.17391304	82.61	78.26
		(0.01)	(10, 1000)	(0.00001)	(0.1,0.1)	(0.00001,0.00001)	(0.0001, 0.00001)
	10%	77.17	78.26	75	77.17	82.61	78.26
		(0.01)	(1000, 10)	(10)	(0.00001,0.00001)	(0.00001,0.00001)	(1, 0,1)
	15%	78.26	78.26	59.78	78.26	82.61	78.26
		(0.01)	(0.00001,0.00001)	(0.00001)	(0.00001,0.00001)	(0.00001,0.00001)	(1, 0,1)
	20%	76.09	73.91	59.78	78.26	81.52173913	77.17
		(0.01)	(1000, 10)	(0.00001)	(0.00001,0.00001)	(0.00001,0.00001)	(0.1, 0.1)
haberman	0%	77.17	57.96	77.17	78.26	82.61	81.52
		(0.01)	(0.00001, 0.00001)	(10)	(0.1,0.1)	(0.00001,0.00001)	(0.00001, 0.0001)
	5%	76.09	75	77.17	77.17391304	82.61	78.26
		(0.01)	(10, 1000)	(10)	(0.1,0.1)	(0.00001,0.00001)	(10, 0,1)
	10%	77.17	78.26	58.04	77.17	82.61	78.26
		(0.01)	(10, 1000)	(100)	(0.00001,0.00001)	(0.00001,0.00001)	(1, 0,1)
	15%	78.26	78.26	59.78	78.26	78.26086957	77.17
		(0.01)	(0.00001,0.00001)	(0.00001)	(0.00001,0.00001)	(0.1, 1)	(0.1, 0.1)
	20%	76.09	73.91	68.48	78.26	82.61	78.26
		(0.01)	(10, 1000)	(0.00001)	(0.00001,0.00001)	(0.00001,0.00001)	(0.1, 0.1)
heart_hungarian	0%	77.53	75.28	71.53	76.40	79.78	79.78
		(0.1)	(1, 1)	(10)	(1, 1)	(0.00001, 0.00001)	(0.01, 0.00001)
	5%	77.53	73.03	77.53	74.16	80.90	78.65
		(0.01)	(1, 0.1)	(100)	(1, 1)	(0.1,0.1)	(100, 10)
	10%	78.65	73.03	75.28	73.03	79.78	70.79
		(0.01)	(1, 0.1)	(0.00001)	(0.00001,0.00001)	(0.00001, 0.00001)	(0.0001, 0.00001)
	15%	73.65	66.29	75.28	76.40	78.65	74.16
		(0.01)	(100, 1000)	(0.00001)	(1, 0.1)	(10, 1)	(10000, 1000)
	20%	77.53	72.40	71.16	77.53	74.16	71.91
		(0.001)	(0.1, 0.1)	(0.1)	(1, 1)	(1, 0.00001)	(0.1, 0.1)

† represents the proposed model.

Table 3.12 (Continued)

Model / Dataset	Noise	SVM ACC (%) (c_1)	TSVM ACC (%) (c_1, c_2)	GBSVM ACC (%) (c_1)	LSTSVM ACC (%) (c_1, c_2)	GBLSTSVM† ACC (%) (c_1, c_2)	LS-GBLSTSVM† ACC (%) (c_1, c_2)
led7digit-0-2-4-5-6-7-8-9_vs_1	0%	92.23 (0.00001)	66.77 (1, 1)	78.35 (100000)	93.98 (100, 100)	94.74 (100000, 100000)	93.98 (10, 0.1)
	5%	83.23 (0.00001)	83.98 (1, 1)	78.35 (100000)	95.49 (1, 1)	94.74 (0.1, 100000)	90.23 (0.0001, 0.00001)
	10%	83.23 (0.00001)	84.74 (0.00001,0.00001)	90.98 (10)	94.74 (0.01, 0.01)	96.24 (0.00001, 1)	93.23 (0.0001, 0.0001)
	15%	83.23 (0.00001)	84.74 (1000, 100)	68.42 (100)	94.74 (0.1, 0.1)	94.74 (0.00001, 0.00001)	93.23 (10000, 1)
	20%	92.23 (0.00001)	83.74 (1000, 100)	80.41 (100000)	94.74 (0.00001,0.00001)	94.74 (0.00001, 0.00001)	93.23 (0.00001, 0.1)
mammographic	0%	79.58 (0.001)	60.41 (0.00001,0.00001)	80.28 (0.00001)	82.01 (0.00001, 0.00001)	84.08 (0.1, 0.1)	83.74 (0.01, 0.0001)
	5%	79.7 (0.001)	64.01 (10000, 10)	80.28 (0.00001)	81.66 (100, 100)	83.74 (1, 1)	84.08 (0.1, 0.1)
	10%	80.28 (0.001)	55.02 (10, 100)	79.93 (0.00001)	83.04 (1000, 1000)	84.43 (0.01, 0.1)	84.43 (0.00001, 0.00001)
	15%	79.93 (0.001)	81.31 (1, 1)	80.62 (0.00001)	82.01 (1000, 1000)	84.78 (0.001, 0.1)	84.43 (0.1, 0.1)
	20%	79.58 (0.001)	82.01 (1, 1)	79.58 (100000)	81.66 (1000, 1000)	82.01 (0.001, 0.001)	85.47 (1, 0.1)
monk2	0%	62.98 (0.00001)	62.43 (0.001, 0.00001)	62.54 (100)	62.98 (1, 1)	69.61 (0.00001, 0.00001)	62.43 (100, 10)
	5%	62.98 (0.00001)	61.98 (0.001, 0.00001)	61.59 (0.00001)	62.98 (1, 1)	63.54 (0.00001, 0.00001)	65.75 (0.00001, 0.1)
	10%	60.98 (0.00001)	62.98 (0.001, 0.00001)	60.28 (1000)	62.98 (0.01, 0.00001)	62.98 (0.0001, 0.00001)	62.43 (0.00001, 0.001)
	15%	60.98 (0.00001)	60.98 (0.001, 0.00001)	60.80 (0.1)	62.98 (0.01, 0.00001)	62.98 (0.0001, 0.00001)	62.98 (100, 0.1)
	20%	60.98 (0.00001)	60.98 (0.001, 0.00001)	59.72 (0.00001)	62.98 (0.0001, 0.00001)	62.98 (0.0001, 0.00001)	67.96 (0.00001, 0.1)
monks_2	0%	62.98 (0.00001)	62.43 (0.0001, 0.00001)	62.54 (1000)	62.98 (0.0001, 0.00001)	62.98 (100, 10000)	64.64 (10, 10000)
	5%	61.78 (0.00001)	61.98 (0.0001, 0.00001)	59.72 (10)	62.98 (0.0001, 0.00001)	62.98 (10, 100)	62.98 (1, 10000)
	10%	60.98 (0.00001)	62.98 (0.0001, 0.00001)	60.49 (0.00001)	62.98 (0.0001, 0.00001)	62.98 (0.001, 0.00001)	64.09 (0.00001, 0.001)
	15%	62.98 (0.00001)	60.98 (0.0001, 0.00001)	60.96 (10)	62.98 (0.0001, 0.00001)	62.98 (0.00001, 0.00001)	64.64 (1, 10000)
	20%	62.98 (0.00001)	60.98 (0.0001, 0.00001)	58.62 (0.00001)	62.98 (0.0001, 0.00001)	64.09 (1000, 10)	61.33 (10, 100000)
monks_3	0%	75.45 (0.1)	59.7 (1, 1)	59.88 (100000)	78.44 (0.1,0.1)	81.44 (10, 10)	77.84 (0.1, 0.1)
	5%	73.65 (0.1)	77.25 (10, 10000)	59.88 (1)	79.04 (0.00001, 0.00001)	80.84 (1, 1)	77.25 (0.00001, 0.00001)
	10%	73.05 (0.1)	76.65 (0.00001,0.00001)	70.66 (0.00001)	76.65 (0.00001, 0.00001)	80.84 (1, 1)	73.65 (0.001, 0.0001)
	15%	73.05 (100000)	70.44 (0.00001,0.00001)	70.06 (100000)	78.44 (0.00001, 0.00001)	82.04 (1, 1)	75.45 (0.1, 0.1)
	20%	71.86 (0.01)	71.26 (10, 10000)	80 (100000)	76.65 (0.00001, 0.00001)	78.44 (1, 0.1)	65.87 (0.1, 0.1)
musk_1	0%	68.53 (0.001)	59.15 (1, 1)	52.66 (10)	79.72 (0.1,0.1)	81.12 (10000, 100000)	78.32 (0.0001, 0.00001)
	5%	67.13 (0.001)	77.62 (1, 0.1)	56.15 (100000)	70.63 (100, 10000)	80.42 (1000, 100000)	79.02 (10000, 10)
	10%	76.92 (0.001)	79.02 (1, 1)	80 (100000)	69.23 (10, 10000)	69.93 (0.00001, 10000)	79.02 (0.1, 0.01)
	15%	69.02 (0.001)	65.03 (10, 0.1)	71.32 (100000)	69.23 (1, 0.001)	69.93 (100000, 0.00001)	74.13 (10, 10)
	20%	67.83 (0.01)	67.13 (10, 0.1)	70 (100000)	56.64 (1, 1000)	76.92 (0.00001, 0.00001)	69.23 (10, 0.1)
ozone	0%	94.58 (0.00001)	85.58 (1, 1)	94.58 (10)	96.58 (0.00001, 0.00001)	96.58 (100000, 100000)	96.58 (0.00001, 0.1)
	5%	86.58 (0.00001)	85.29 (0.00001, 0.00001)	96.58 (0.001)	96.58 (0.0001, 0.0001)	96.58 (0.00001, 0.00001)	96.58 (0.00001, 0.1)
	10%	96.58 (0.00001)	96.58 (0.00001, 0.00001)	75 (0.00001)	96.58 (0.0001, 0.00001)	96.58 (0.00001, 0.00001)	96.58 (0.00001, 0.1)
	15%	80.58 (0.00001)	85.79 (1, 1)	83.09 (100)	96.58 (0.0001, 0.00001)	96.58 (0.00001, 0.00001)	96.58 (0.00001, 0.1)
	20%	86.58 (0.00001)	96.58 (10, 10)	66.89 (10)	96.58 (0.001, 0.00001)	96.58 (0.00001, 0.00001)	96.58 (0.00001, 0.1)
spambase	0%	88.78 (0.001)	74.18 (0.001, 0.001)	79.79 (100)	91.53 (0.1, 0.1)	90.88 (0.1, 0.01)	88.70 (1, 0.1)
	5%	88.49 (0.001)	81.17 (0.1, 0.1)	51.99 (10000)	90.30 (0.01, 0.01)	89.86 (0.01, 0.001)	88.70 (0.01, 0.001)
	10%	88.41 (0.001)	91.09 (0.1, 0.1)	81.25 (10000)	90.80 (0.01, 0.01)	90.51 (0.0001, 0.0001)	79.29 (100, 0.01)
	15%	88.27 (0.001)	88.85 (0.001, 0.001)	74.41 (100)	88.92 (0.01, 0.01)	90.88 (0.1, 0.01)	84.79 (100, 1)
	20%	88.63 (10)	89.43 (0.1, 0.1)	88.92 (10000)	89.28 (100, 100)	90.08 (0.1, 0.01)	80.01 (1, 0.1)
spectf	0%	76.54 (0.01)	62.39 (0.1, 0.01)	70.4 (100)	82.72 (0.1, 0.01)	83.95 (1000, 10000)	77.78 (100, 100)
	5%	76.54 (0.001)	77.78 (0.1, 0.1)	70.99 (100000)	75.31 (0.1, 0.01)	77.78 (0.00001, 0.00001)	75.31 (0.001, 0.00001)
	10%	76.54 (0.001)	80.25 (0.01, 0.01)	62.67 (100000)	76.54 (0.001, 0.00001)	77.78 (0.00001, 0.00001)	79.01 (0.0001, 0.00001)
	15%	76.54 (0.001)	82.72 (0.01, 0.01)	70.44 (10)	80.25 (0.001, 0.001)	80.25 (1000, 1000)	81.48 (1000, 0.100)
	20%	76.54 (0.001)	77.78 (0.00001, 0.00001)	73.33 (1)	80.25 (0.00001, 0.00001)	80.25 (0.00001, 0.00001)	80.25 (0.00001, 0.01)

† represents the proposed model.

Table 3.12 (Continued)

Model Dataset	Noise	SVM ACC (%) (c_1)	TSVM ACC (%) (c_1, c_2)	GBSVM ACC (%) (c_1)	LSTSVM ACC (%) (c_1, c_2)	GBLSTSVM† ACC (%) (c_1, c_2)	LS-GBLSTSVM† ACC (%) (c_1, c_2)
tic_tac_toe	0%	75.69	68.66	76.88	99.65	99.65	99.65
		(0.001)	(0.00001, 0.00001)	(100000)	(0.00001, 0.00001)	(0.00001, 0.00001)	(0.1, 0.1)
	5%	75.35	95.65	70.47	99.65	99.65	99.65
		(0.001)	(0.00001, 0.00001)	(1000)	(0.00001, 0.00001)	(0.00001, 0.00001)	(0.01, 0.00001)
	10%	76.04	89.65	70.97	99.65	99.65	99.65
		(0.001)	(0.00001, 0.00001)	(100)	(0.00001, 0.00001)	(0.00001, 0.00001)	(0.00001, 0.00001)
	15%	74.65	89.65	60.76	99.65	99.65	99.65
		(0.001)	(0.00001, 0.00001)	(10)	(0.00001, 0.00001)	(0.00001, 0.00001)	(0.1, 0.1)
	20%	73.96	89.65	63.33	99.65	99.65	93.40
		(0.001)	(10, 100)	(100000)	(0.00001, 0.00001)	(0.00001, 0.00001)	(0.1, 0.1)
vehicle1	0%	76.38	59.44	73.62	77.95	83.46	77.17
		(0.00001)	(10, 1)	(10)	(100, 10)	(1, 1)	(0.01, 0.00001)
	5%	76.38	78.74	73.62	77.95	81.50	77.56
		(0.00001)	(10, 1)	(10)	(100, 10)	(0.01, 0.1)	(0.01, 0.00001)
	10%	76.38	81.5	71.26	83.86	83.46	77.56
		(0.00001)	(0.1, 0.1)	(10)	(10, 10)	(1, 1)	(0.01, 0.0001)
	15%	76.38	80.71	70.47	83.07	83.07	80.71
		(0.00001)	(0.01, 0.01)	(10)	(0.1, 0.1)	(0.1, 1000)	(10000, 10)
	20%	76.38	78.35	76.38	77.17	80.71	75.20
		(0.00001)	(10, 1)	(10)	(0.1, 0.01)	(0.1, 0.1)	(0.00001, 1)
yeast-0-2-5-6_vs_3-7-8-9	0%	83.71	66.23	64.9	91.72	94.37	93.38
		(0.001)	(10, 10)	(10)	(100, 100)	(1, 1)	(100, 10)
	5%	88.38	81.06	82.65	91.39	94.37	92.05
		(0.001)	(0.00001, 0.00001)	(100)	(1000, 1000)	(1, 1)	(1, 0.1)
	10%	91.72	82.05	54.97	92.05	92.38	92.05
		(0.001)	(0.1, 0.1)	(100)	(0.01, 0.01)	(1, 1)	(10, 0.01)
	15%	88.01	81.06	75.5	92.05	93.38	93.38
		(0.001)	(0.00001, 0.00001)	(10)	(0.1, 0.1)	(10, 100)	(100, 0.1)
	20%	87.35	87.55	84.7	91.72	92.72	90.40
		(0.001)	(100, 100)	(100)	(1, 1)	(10, 10)	(1, 0.1)
yeast-0-2-5-7-9_vs_3-6-8	0%	86.79	66.67	68.55	97.02	97.35	95.70
		(0.001)	(1, 1)	(10)	(0,1, 1)	(0.01, 0.1)	(1000, 100)
	5%	88.14	80.79	68.42	95.70	97.35	97.68
		(0.001)	(1, 1)	(10)	(1, 1)	(0.01, 0.1)	(10, 0.1)
	10%	87.02	94.37	87.15	97.35	94.70	93.71
		(0.001)	(1, 1)	(10000)	(1, 1)	(0.1, 1)	(0.1, 0.00001)
	15%	87.5	80.79	74.34	97.35	98.01	93.05
		(0.001)	(0.1, 0.1)	(10)	(1, 1)	(1, 1)	(1, 0.1)
	20%	87.5	80.79	77.63	96.69	97.68	98.01
		(0.001)	(10000, 10000)	(1000)	(0.1, 0.1)	(1, 1)	(10, 1)
yeast-0-5-6-7-9_vs_4	0%	81.19	68.29	56.6	90.57	94.34	91.19
		(0.00001)	(0.1, 0.1)	(100000)	(0.01, 0.01)	(0.1, 0.1)	(0.00001, 0.0001)
	5%	81.19	81.19	68.55	92.45	91.19	91.82
		(0.00001)	(0.1, 0.01)	(100000)	(10, 10)	(0.00001, 0.00001)	(1, 0.1)
	10%	81.19	91.82	83.4	92.45	94.19	91.19
		(0.00001)	(0.1, 0.1)	(10000)	(10, 10)	(0.00001, 1)	(0.00001, 0.00001)
	15%	81.19	81.82	89.94	91.82	94.34	91.19
		(0.00001)	(0.001, 0.0001)	(10)	(0.0001, 0.00001)	(0.00001, 0.00001)	(0.00001, 0.1)
	20%	81.19	91.19	83.65	93.08	94.34	91.19
		(0.00001)	(0.001, 0.00001)	(10)	(1000, 100)	(0.00001, 0.00001)	(0.00001, 0.1)
yeast-2_vs_4	0%	85.81	67.7	54.19	95.48	93.55	94.84
		(0.00001)	(0.1, 1)	(10)	(0.1, 1)	(100, 10)	(0.0001, 0.00001)
	5%	85.81	84.84	85.81	93.55	96.13	93.55
		(0.00001)	(0.1, 1)	(100)	(0.1, 1)	(0.00001, 1)	(1, 0.1)
	10%	85.81	87.1	72.67	90.97	92.26	88.68
		(0.00001)	(1, 1)	(100)	(1000, 100)	(0.1, 0.00001)	(0.01, 0.001)
	15%	72.87	86.45	75.76	88.39	90.97	93.55
		(0.00001)	(1, 1)	(10000)	(1, 1)	(0.00001, 1)	(10, 0.1)
	20%	85.81	87.1	76.13	87.10	94.84	85.81
		(0.00001)	(10, 1)	(10)	(1000, 100)	(0.1, 0.1)	(0.00001, 0.1)
yeast3	0%	79.91	67.15	79.03	93.27	94.17	93.72
		(0.001)	(0.1, 1)	(10000)	(0.1, 1)	(1, 1)	(0.01, 0.01)
	5%	80.81	81.48	81.26	91.26	93.95	90.13
		(0.001)	(1, 1)	(1)	(0.001, 0.001)	(1, 1)	(10, 0.1)
	10%	89.69	89.69	82.74	91.26	94.39	88.79
		(0.001)	(0.1, 0.1)	(10)	(1, 1)	(0.1, 1)	(0.001, 0.0001)
	15%	89.46	88.79	82.87	91.03	93.95	89.46
		(0.001)	(0.00001, 0.00001)	(10)	(1, 1)	(0.1, 1)	(1, 0.1)
	20%	88.12	80.81	76.23	91.03	93.95	88.12
		(0.00001)	(0.00001, 0.00001)	(10)	(0.1, 0.1)	(0.1, 1)	(10, 0.1)
Average Accuracy	0%	81.58	71.52	73.71	86.67	**88.26**	86.79
	5%	80.65	81.26	77.25	85.33	**87.50**	85.59
	10%	81.70	82.83	76.63	84.82	**87.47**	85.50
	15%	80.03	80.17	73.89	84.12	**86.04**	84.93
	20%	80.21	80.55	75.76	83.79	**86.09**	83.85
Average Rank	0%	3.88	5.26	5.15	2.59	**1.62**	2.50
	5%	4.46	4.37	4.97	2.94	**1.71**	2.56
	10%	4.40	3.65	4.99	3.10	**1.93**	2.94
	15%	4.24	4.22	5.29	2.74	**1.68**	2.84
	20%	4.29	4.19	5.16	2.72	**1.71**	2.93

† represents the proposed model.

Table 3.13: Performance comparison of the proposed GBLSTSVM and LS-GBLSTSVM along with the baseline models based on classification accuracy using Gaussian kernel for UCI and KEEL datasets.

DATASET	NOISE	SVM ACC(%) (c_1, μ)	TSVM ACC(%) (c_1, c_2, μ)	GBSVM ACC(%) (c_1, μ)	LSTSVM ACC(%) (c_1, c_2, μ)	GBLSTSVM† ACC(%) (c_1, c_2, μ)	LS-GBLSTSVM† ACC(%) (c_1, c_2, μ)
acute_nephritis	0%	41.67	90.00	65.58	41.66	100.00	94.44
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 2)	(10, 1000, 2)
	5%	41.67	90.00	65.59	41.67	100.00	83.33
		(0.00001, 0.03125)	(0.001, 0.001, 0.03125)	(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 0.5)	(10, 1000, 2)
	10%	41.67	90.00	84.44	41.67	100.00	91.67
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 2)	(0.1, 0.1, 32)
	15%	41.67	90.44	80.33	41.67	100.00	100.00
		(0.00001, 0.03125)	(0.001, 0.001, 0.03125)	(100000, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.1, 0.1, 32)
	20%	41.67	86.11	83.33	41.67	100.00	100.00
		(0.00001, 0.03125)	(0.01, 0.001, 0.03125)	(100000, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.1, 0.1, 32)
breast_cancer_wisc_prog	0%	75.66	75.66	83.33	76.66	76.67	75.00
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(10, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.1, 0.01, 32)
	5%	75.67	75.67	76.67	76.67	63.33	70.00
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100, 1, 32)
	10%	73.67	75.67	70.67	76.67	63.33	78.33
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(1000, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.00001, 0.1, 32)
	15%	74.67	71.67	69.67	76.67	63.33	58.33
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(1000, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.00001, 0.1, 32)
	20%	71.67	72.67	71.33	76.67	63.33	76.67
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(1000, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.00001, 0.1, 32)
breast_cancer	0%	74.42	67.44	100.00	74.41	70.93	72.09
		(0.00001, 0.03125)	(0.1, 0.001, 0.0625)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100000, 1000, 32)
	5%	74.42	67.44	74.42	74.42	73.26	74.42
		(0.00001, 0.03125)	(0.1, 0.001, 0.125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(10, 100000, 32)
	10%	74.42	69.77	90.78	74.42	73.26	75.58
		(0.00001, 0.03125)	(0.1, 0.001, 0.5)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(10, 1, 32)
	15%	74.42	68.60	58.84	74.42	75.58	74.42
		(0.00001, 0.03125)	(0.1, 0.001, 0.0625)	(100000, 2)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1000, 100, 32)
	20%	74.42	74.42	77.91	74.42	76.74	74.42
		(0.00001, 0.03125)	(0.01, 0.00001, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1000, 100, 32)
conn_bench_sonar_mines_rocks	0%	52.38	55.55	71.26	53.96	80.95	77.78
		(10, 1)	(0.1, 0.1, 0.03125)	(1000, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1000, 10, 32)
	5%	52.38	55.56	69.68	53.97	77.78	69.84
		(1, 1)	(0.1, 0.1, 0.03125)	(1000, 0.0625)	(0.00001, 0.00001, 0.25)	(100000, 100000, 32)	(10000, 10, 32)
	10%	47.62	53.97	69.68	53.97	68.25	74.60
		(0.1, 4)	(0.0001, 0.00001, 0.03125)	(100000, 0.03125)	(0.00001, 0.00001, 0.25)	(100000, 100000, 32)	(10, 1, 32)
	15%	50.79	55.56	61.10	53.97	76.19	73.02
		(1, 1)	(0.1, 0.1, 0.03125)	(100000, 0.03125)	(0.00001, 0.00001, 0.25)	(100000, 100000, 32)	(1, 1, 32)
	20%	46.03	57.14	69.84	53.97	66.67	76.19
		(0.001, 16)	(0.1, 0.1, 0.03125)	(100000, 0.03125)	(0.00001, 0.00001, 0.25)	(100000, 100000, 32)	(10, 10, 32)
crossplane130	0%	51.28	100.00	100.00	48.71	100.00	100.00
		(0.001, 16)	(0.00001, 0.00001, 0.03125)	(100000, 32)	(0.00001, 0.001, 0.03125)	(100000, 100000, 32)	(10, 10, 32)
	5%	81.28	100.00	90.44	51.28	97.44	100.00
		(0.001, 4)	(0.01, 0.001, 4)	(100, 2)	(0.00001, 0.00001, 32)	(100000, 100000, 32)	(10, 10, 32)
	10%	51.28	97.87	89.74	48.72	97.44	94.87
		(0.001, 2)	(1, 1, 0.03125)	(10, 1)	(0.01, 0.001, 0.0625)	(100000, 100000, 32)	(0.1, 0.1, 32)
	15%	81.28	100.00	94.78	48.72	97.44	94.87
		(0.001, 2)	(1, 1, 0.03125)	(100000, 32)	(0.01, 0.001, 0.0625)	(100000, 100000, 32)	(0.1, 0.1, 32)
	20%	81.28	90.44	89.19	74.36	100.00	92.31
		(0.01, 0.25)	(0.1, 0.1, 16)	(100000, 32)	(0.01, 0.001, 0.0625)	(100000, 100000, 32)	(100, 1, 32)
crossplane150	0%	57.77	67.77	62.22	68.88	55.56	80.00
		(0.00001, 0.03125)	(0.0001, 0.001, 0.03125)	(100000, 0.03125)	(100, 0.001, 0.03125)	(100000, 100000, 16)	(0.1, 0.1, 32)
	5%	57.78	68.00	82.67	37.78	55.56	62.22
		(0.00001, 0.03125)	(0.0001, 0.00001, 0.03125)	(0.00001, 0.03125)	(100, 0.001, 0.125)	(100000, 100000, 16)	(0.1, 1000, 32)
	10%	57.78	61.11	66.67	37.78	80.00	77.78
		(0.00001, 0.03125)	(1, 1, 0.03125)	(100000, 0.03125)	(100, 0.001, 0.125)	(100000, 100000, 32)	(0.001, 1000, 32)
	15%	62.22	65.56	57.78	37.78	82.22	73.33
		(1, 0.25)	(0.1, 0.1, 0.03125)	(10, 0.03125)	(100, 0.001, 16)	(100000, 100000, 32)	(100, 100, 32)
	20%	57.78	64.44	51.89	37.78	84.44	75.56
		(0.00001, 0.03125)	(1, 1, 0.03125)	(100000, 0.03125)	(100, 0.001, 16)	(100000, 100000, 32)	(100, 100, 32)
ecoli-0-1_vs_5	0%	94.44	97.22	97.22	88.89	88.89	94.44
		(1, 1)	(0.00001, 0.0001, 0.125)	(10, 8)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.1, 1, 32)
	5%	88.89	90.22	90.22	88.89	97.22	94.44
		(0.00001, 0.0625)	(0.01, 0.01, 0.03125)	(1000, 4)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 8)	(1, 0.1, 32)
	10%	88.89	95.83	100.00	88.89	94.44	93.06
		(0.00001, 0.0625)	(0.01, 0.01, 0.03125)	(100000, 1)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 16)	(0.1, 1000, 32)
	15%	88.89	94.44	100.00	88.89	94.44	91.67
		(0.00001, 0.03125)	(0.01, 0.01, 0.03125)	(100000, 2)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 16)	(100, 10, 32)
	20%	88.89	95.83	88.89	88.89	88.89	88.89
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 2)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(10, 100, 32)
ecoli-0-1-4-6_vs_5.	0%	98.80	100.00	100.00	94.05	94.05	92.86
		(1, 1)	(0.01, 0.1, 0.0625)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1, 0.1, 8)
	5%	94.05	92.81	88.92	94.05	94.05	98.81
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(10, 1, 32)
	10%	94.05	94.05	94.05	94.05	98.81	97.62
		(0.00001, 0.03125)	(0.01, 0.01, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1, 0.1, 32)
	15%	94.05	94.05	90.87	94.05	94.05	94.05
		(0.00001, 0.03125)	(0.01, 0.01, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1, 0.1, 32)
	20%	94.05	90.62	73.10	94.05	95.24	97.62
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 16)	(100, 10, 32)
ecoli-0-1-4-7_vs_2-3-5-6	0%	87.12	96.04	82.69	87.12	87.13	93.07
		(0.00001, 0.03125)	(0.1, 1, 0.5)	(100000, 8)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 8)	(10, 1, 32)
	5%	87.13	90.04	80.54	87.13	93.07	72.28
		(0.00001, 0.03125)	(0.1, 0.1, 0.25)	(100000, 2)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.00001, 0.01, 32)
	10%	87.13	83.07	85.69	87.13	92.08	84.16
		(0.00001, 0.03125)	(0.01, 0.01, 0.03125)	(100000, 4)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.1, 0.1, 32)
	15%	87.13	93.07	92.54	87.13	90.10	87.13
		(0.00001, 0.03125)	(1, 0.01, 0.03125)	(100000, 4)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1, 10, 32)
	20%	87.13	93.07	73.47	87.13	78.22	92.08
		(0.00001, 0.03125)	(0.1, 0.01, 0.125)	(100000, 0.25)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100, 100, 32)

† represents the proposed model.

Table 3.13 (Continued)

DATASET	NOISE	SVM ACC(%) (c_1, μ)	TSVM ACC(%) (c_1, c_2, μ)	GBSVM ACC(%) (c_1, μ)	LSTSVM ACC(%) (c_1, c_2, μ)	GBLSTSVM[†] ACC(%) (c_1, c_2, μ)	LS-GBLSTSVM[†] ACC(%) (c_1, c_2, μ)
ecoli2	0%	86.10 (0.00001, 0.03125)	87.09 (0.1, 1, 0.03125)	82.85 (10, 0.0625)	89.11 (0.00001, 0.00001, 0.03125)	89.11 (100000, 100000, 32)	88.12 (1, 0.1, 32)
	5%	85.11 (0.00001, 0.03125)	87.11 (0.001, 0.01, 0.03125)	72.44 (0.00001, 0.03125)	89.11 (0.00001, 0.00001, 0.03125)	89.11 (100000, 100000, 0.25)	88.12 (1, 0.1, 32)
	10%	84.11 (0.00001, 0.03125)	86.14 (0.1, 0.1, 0.03125)	80.66 (10000, 0.03125)	89.11 (0.00001, 0.00001, 0.03125)	88.12 (100000, 100000, 32)	86.14 (1, 10, 32)
	15%	83.11 (0.00001, 0.03125)	86.11 (0.0001, 0.00001, 0.03125)	77.32 (10, 0.03125)	89.11 (0.00001, 0.00001, 0.03125)	90.10 (100000, 100000, 32)	89.11 (1, 10, 32)
	20%	81.11 (0.00001, 0.03125)	85.11 (0.0001, 0.00001, 0.03125)	81.15 (100, 0.03125)	89.11 (0.00001, 0.00001, 0.03125)	90.10 (100000, 100000, 32)	94.06 (0.00001, 0.1, 32)
ecoli3	0%	86.10 (0.00001, 0.03125)	87.09 (0.1, 1, 0.03125)	82.85 (10, 0.0625)	89.10 (0.00001, 0.00001, 0.03125)	89.11 (100000, 100000, 32)	88.12 (1, 0.1, 32)
	5%	85.11 (0.00001, 0.03125)	87.11 (0.001, 0.01, 0.03125)	86.44 (0.00001, 0.03125)	89.11 (0.00001, 0.00001, 0.03125)	89.11 (100000, 100000, 0.25)	88.12 (1, 0.1, 32)
	10%	84.11 (0.00001, 0.03125)	86.14 (0.1, 0.1, 0.03125)	81.66 (10000, 0.03125)	89.11 (0.00001, 0.00001, 0.03125)	88.12 (100000, 100000, 32)	86.14 (1, 10, 32)
	15%	83.11 (0.00001, 0.03125)	86.11 (0.0001, 0.00001, 0.03125)	72.32 (10, 0.03125)	89.11 (0.00001, 0.00001, 0.03125)	90.10 (100000, 100000, 32)	89.11 (0.1, 10, 32)
	20%	81.11 (0.00001, 0.03125)	85.11 (0.0001, 0.00001, 0.03125)	81.15 (100, 0.03125)	89.11 (0.00001, 0.00001, 0.03125)	90.10 (100000, 100000, 32)	90.10 (0.00001, 0.1, 32)
fertility	0%	89.00 (0.00001, 0.03125)	89.00 (0.00001, 0.00001, 0.03125)	87.00 (100, 0.0625)	90.00 (0.00001, 0.00001, 0.03125)	90.00 (100000, 100000, 32)	90.00 (0.00001, 0.1, 32)
	5%	89.00 (0.00001, 0.03125)	86.67 (0.00001, 0.00001, 0.03125)	80.67 (100, 0.0625)	90.00 (0.00001, 0.00001, 0.03125)	90.00 (100000, 100000, 32)	90.00 (0.00001, 0.1, 32)
	10%	89.00 (0.00001, 0.03125)	88.00 (0.00001, 0.00001, 0.03125)	72.67 (10, 0.03125)	90.00 (0.00001, 0.00001, 0.03125)	90.00 (100000, 100000, 32)	90.00 (0.00001, 0.1, 32)
	15%	86.00 (0.00001, 0.03125)	89.00 (0.0001, 0.00001, 0.03125)	80.00 (1000, 0.03125)	90.00 (0.00001, 0.00001, 0.03125)	90.00 (100000, 100000, 32)	90.00 (0.00001, 0.1, 32)
	20%	56.67 (10, 1)	89.00 (0.1, 0.01, 0.03125)	70.67 (10000, 0.03125)	90.00 (0.00001, 0.00001, 0.03125)	90.00 (100000, 100000, 32)	90.00 (10, 100, 32)
glass2	0%	89.76 (0.00001, 0.03125)	89.76 (0.00001, 0.00001, 0.03125)	75.76 (0.00001, 0.03125)	90.76 (0.00001, 0.00001, 0.03125)	90.77 (100000, 100000, 32)	90.77 (0.00001, 0.1, 32)
	5%	89.77 (0.00001, 0.03125)	88.77 (1, 1, 0.03125)	80.17 (10, 0.03125)	90.77 (0.00001, 0.00001, 0.03125)	90.77 (100000, 100000, 32)	90.77 (0.00001, 0.1, 32)
	10%	89.77 (0.00001, 0.03125)	89.77 (0.001, 0.00001, 0.03125)	80.17 (10, 0.03125)	90.77 (0.00001, 0.00001, 0.03125)	90.77 (100000, 100000, 32)	90.77 (0.00001, 0.1, 32)
	15%	86.77 (0.00001, 0.03125)	89.77 (0.001, 0.00001, 0.03125)	75.38 (100, 0.03125)	90.77 (0.00001, 0.00001, 0.03125)	90.77 (100000, 100000, 32)	90.77 (0.00001, 0.1, 32)
	20%	82.77 (0.00001, 0.03125)	89.77 (0.001, 0.00001, 0.03125)	89.23 (100, 0.03125)	90.77 (0.00001, 0.00001, 0.03125)	90.77 (100000, 100000, 32)	90.77 (0.00001, 0.1, 32)
glass5	0%	86.92 (1, 1)	94.92 (0.001, 0.01, 0.03125)	80.75 (1, 1)	96.92 (0.00001, 0.00001, 0.03125)	96.92 (100000, 100000, 32)	96.92 (0.00001, 0.1, 32)
	5%	86.92 (0.00001, 0.03125)	86.92 (0.1, 0.01, 0.03125)	86.15 (10, 0.03125)	96.92 (0.00001, 0.00001, 0.03125)	96.92 (100000, 100000, 32)	92.31 (0.00001, 0.1, 32)
	10%	86.92 (0.00001, 0.03125)	92.92 (0.1, 0.01, 0.03125)	76.92 (10, 0.03125)	96.92 (0.00001, 0.00001, 0.03125)	96.92 (100000, 100000, 32)	95.38 (0.00001, 0.1, 32)
	15%	82.92 (0.00001, 0.03125)	90.92 (0.01, 0.00001, 0.03125)	94.92 (10, 0.03125)	96.92 (0.00001, 0.00001, 0.03125)	96.92 (100000, 100000, 32)	96.92 (0.00001, 0.1, 32)
	20%	81.92 (0.00001, 0.03125)	90.92 (0.0001, 0.00001, 0.03125)	81.92 (10, 0.03125)	96.92 (0.00001, 0.00001, 0.03125)	96.92 (100000, 100000, 32)	96.92 (0.00001, 0.1, 32)
haber	0%	82.61 (0.00001, 0.03125)	75.35 (1, 0.1, 0.125)	57.61 (100000, 0.25)	82.61 (0.00001, 0.00001, 0.03125)	82.61 (100000, 100000, 32)	82.61 (100000, 100000, 32)
	5%	82.61 (0.00001, 0.03125)	70.26 (1, 0.1, 0.125)	72.10 (100000, 0.25)	82.61 (0.00001, 0.00001, 0.03125)	80.43 (100000, 100000, 32)	82.61 (100000, 100000, 32)
	10%	82.61 (0.00001, 0.03125)	75.35 (1, 0.1, 0.125)	74.00 (100000, 0.25)	82.61 (0.00001, 0.00001, 0.03125)	80.43 (100000, 100000, 32)	70.65 (100, 10000, 32)
	15%	82.61 (0.00001, 0.03125)	79.35 (1, 0.1, 0.125)	80.67 (100000, 2)	82.61 (0.00001, 0.00001, 0.03125)	82.61 (100000, 100000, 32)	82.61 (100000, 100000, 32)
	20%	82.61 (0.00001, 0.03125)	78.26 (1, 0.1, 0.125)	78.70 (100000, 0.25)	82.61 (0.00001, 0.00001, 0.03125)	77.17 (100000, 100000, 32)	80.43 (0.1, 0.1, 32)
haberman_survival	0%	82.60 (0.00001, 0.03125)	79.35 (1, 0.1, 0.03125)	57.61 (100000, 32)	82.60 (0.00001, 0.00001, 0.03125)	81.52 (100000, 100000, 32)	82.61 (100000, 100000, 32)
	5%	82.61 (0.00001, 0.03125)	78.26 (1, 0.1, 0.03125)	70.97 (100000, 32)	82.61 (0.00001, 0.00001, 0.03125)	82.61 (100000, 100000, 32)	80.43 (100000, 100000, 32)
	10%	82.61 (0.00001, 0.03125)	79.35 (1, 0.1, 0.03125)	78.59 (100000, 32)	82.61 (0.00001, 0.00001, 0.03125)	81.52 (100000, 100000, 32)	75.00 (1000, 10000, 32)
	15%	82.61 (0.00001, 0.03125)	79.35 (1, 0.1, 0.03125)	79.42 (100000, 32)	82.61 (0.00001, 0.00001, 0.03125)	83.70 (100000, 100000, 32)	81.52 (100000, 100000, 32)
	20%	82.61 (0.00001, 0.03125)	78.26 (1, 0.1, 0.03125)	75.65 (100000, 32)	82.61 (0.00001, 10, 0.5)	84.78 (100000, 100000, 32)	82.61 (100000, 100000, 32)
haberman	0%	81.52 (0.1, 1)	75.35 (0.1, 1, 0.25)	57.61 (100000, 0.25)	80.43 (0.0001, 10, 0.125)	82.61 (100000, 100000, 32)	82.61 (100000, 100000, 32)
	5%	81.52 (0.1, 1)	70.26 (0.1, 1, 0.125)	72.10 (100000, 0.25)	80.43 (0.0001, 10, 0.125)	80.43 (100000, 100000, 32)	82.61 (100000, 100000, 32)
	10%	82.61 (0.00001, 0.03125)	75.35 (0.1, 1, 0.125)	79.00 (100000, 0.25)	81.52 (0.001, 0.00001, 0.03125)	80.43 (100000, 100000, 32)	70.65 (100, 10000, 32)
	15%	82.61 (0.00001, 0.03125)	79.35 (0.1, 1, 0.125)	80.67 (100000, 2)	81.52 (0.001, 0.00001, 0.03125)	82.61 (100000, 100000, 32)	82.61 (100000, 100000, 32)
	20%	82.61 (0.00001, 0.03125)	78.26 (0.1, 1, 0.125)	75.70 (100000, 0.25)	82.61 (0.00001, 0.00001, 0.03125)	77.17 (100000, 100000, 32)	80.43 (0.1, 0.1, 32)
heart_hungarian	0%	65.29 (0.1, 1)	71.91 (0.1, 0.1, 0.03125)	71.52 (10, 0.0625)	65.16 (0.00001, 0.00001, 0.03125)	79.77 (100000, 100000, 32)	82.02 (1000, 10, 32)
	5%	62.42 (0.1, 1)	70.03 (1, 0.1, 0.03125)	72.53 (100, 0.03125)	65.17 (0.00001, 0.00001, 0.03125)	76.40 (100000, 100000, 32)	79.78 (100, 10, 32)
	10%	62.17 (0.00001, 0.03125)	58.43 (0.01, 0.1, 0.03125)	70.28 (0.00001, 0.03125)	65.17 (0.00001, 0.00001, 0.03125)	75.28 (100000, 100000, 32)	73.03 (100, 10, 32)
	15%	65.17 (0.00001, 0.03125)	65.17 (0.00001, 0.00001, 0.03125)	75.28 (0.00001, 0.03125)	65.17 (0.00001, 0.00001, 0.03125)	75.28 (100000, 100000, 32)	75.28 (100, 10, 32)
	20%	62.17 (0.00001, 0.03125)	62.92 (0.00001, 0.00001, 0.03125)	71.16 (0.1, 0.03125)	65.17 (0.00001, 0.00001, 0.03125)	75.28 (100000, 100000, 32)	71.91 (100, 10, 32)

[†] represents the proposed model.

Table 3.13 (Continued)

DATASET	NOISE	SVM ACC(%) (c_1, μ)	TSVM ACC(%) (c_1, c_2, μ)	GBSVM ACC(%) (c_1, μ)	LSTSVM ACC(%) (c_1, c_2, μ)	GBLSTSVM† ACC(%) (c_1, c_2, μ)	LS-GBLSTSVM† ACC(%) (c_1, c_2, μ)
led7digit-0-2-4-5-6-7-8-9_vs_1	0%	81.95	93.98	100.00	93.23	93.23	93.23
		(1, 1)	(0.00001, 0.0001, 0.125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(10, 1000, 32)
	5%	62.42	70.03	72.53	65.17	76.40	79.78
		(0.1, 1)	(0.00001, 0.00001, 0.03125)	(1000, 8)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100, 1000, 32)
	10%	62.17	58.43	70.28	65.17	75.28	93.23
		(0.1, 1)	(0.01, 0.01, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100000, 100000, 32)
	15%	87.22	81.73	80.76	93.23	93.23	93.23
		(0.1, 1)	(0.00001, 0.00001, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100000, 100000, 32)
	20%	93.23	90.23	92.89	93.23	93.23	93.98
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100000, 100000, 32)
mammographic	0%	52.94	79.93	75.76	52.94	79.23	79.93
		(0.00001, 0.03125)	(0.1, 0.1, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1, 0.1, 32)
	5%	52.94	81.66	80.00	52.94	60.90	80.62
		(0.0001, 2)	(0.1, 0.1, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100, 100, 32)
	10%	72.94	81.66	80.76	52.94	64.71	70.24
		(0.00001, 0.03125)	(0.1, 0.1, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1000, 1000, 32)
	15%	52.94	81.66	100.00	52.94	76.12	75.09
		(0.00001, 0.03125)	(0.1, 0.1, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(10, 0.1, 32)
	20%	68.13	80.28	74.05	52.94	64.71	79.24
		(0.1, 0.5)	(0.1, 0.1, 0.03125)	(10000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100, 10, 32)
monk2	0%	60.98	63.37	60.54	62.98	64.64	59.67
		(0.00001, 0.03125)	(1, 1, 0.03125)	(100, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100, 1, 32)
	5%	60.98	61.71	71.59	62.98	60.77	62.43
		(0.00001, 0.03125)	(1, 1, 0.03125)	(0.00001, 0.25)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100, 10, 32)
	10%	60.98	63.43	60.28	62.98	59.12	64.09
		(0.00001, 0.03125)	(1, 1, 0.03125)	(1000, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.0001,1, 32)
	15%	63.48	60.11	60.80	62.98	66.85	64.09
		(1, 0.5)	(1, 10, 0.03125)	(0.1, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.0001,1, 32)
	20%	60.98	75.69	59.72	62.98	71.82	65.19
		(0.00001, 0.03125)	(1, 10, 0.03125)	(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100, 10, 32)
monks_2.	0%	60.98	64.11	60.48	62.98	60.77	58.56
		(0.00001, 0.03125)	(1, 10, 0.03125)	(1000, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100, 1, 32)
	5%	60.98	69.11	59.72	62.98	62.43	63.54
		(0.00001, 0.03125)	(1, 0.1, 0.03125)	(10, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100, 10, 32)
	10%	60.98	72.93	60.49	62.98	59.12	64.09
		(0.00001, 0.03125)	(10, 0.1, 0.03125)	(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.0001,1, 32)
	15%	62.98	66.72	60.96	62.98	62.98	62.98
		(0.00001, 0.03125)	(10, 1, 0.03125)	(10, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.0001,1, 32)
	20%	60.98	64.64	58.62	62.98	59.12	62.98
		(0.00001, 0.03125)	(0.1, 0.1, 0.03125)	(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(100, 10, 32)
monks_3	0%	46.11	75.21	69.52	54.49	64.67	78.44
		(0.00001, 0.03125)	(1, 0.1, 0.25)	(100000, 32)	(0.0001, 0.00001, 0.25)	(100000, 100000, 32)	(10, 0.1, 32)
	5%	69.89	74.61	70.67	52.10	78.44	82.04
		(0.00001, 0.03125)	(0.1, 0.1, 0.125)	(100000, 32)	(0.001, 0.0001, 4)	(100000, 100000, 32)	(1, 0.1, 32)
	10%	66.11	81.62	71.26	51.50	77.25	82.04
		(0.00001, 0.03125)	(0.1, 0.1, 0.125)	(100000, 32)	(0.0001, 0.00001, 4)	(100000, 100000, 32)	(1, 0.1, 32)
	15%	76.11	85.03	80.00	51.50	76.05	79.04
		(0.00001, 0.03125)	(0.1, 0.1, 0.03125)	(100000, 32)	(0.0001, 0.00001, 4)	(100000, 100000, 32)	(1, 1, 32)
	20%	76.11	79.64	70.06	75.45	77.84	68.26
		(0.00001, 0.03125)	(0.1, 0.1, 0.03125)	(100000, 32)	(0.001, 0.0001, 0.03125)	(100000, 100000, 32)	(0.1, 0.01, 32)
musk_1	0%	53.15	83.15	46.85	53.14	82.51	81.82
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(0.1, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1, 0.1, 32)
	5%	53.15	53.15	41.96	53.15	82.52	82.52
		(0.00001, 0.03125)	(0.0001, 0.00001, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1, 0.1, 32)
	10%	53.15	53.15	61.54	53.15	81.12	74.83
		(0.00001, 0.03125)	(0.0001, 0.00001, 0.03125)	(10000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(10, 0.1, 32)
	15%	69.23	53.15	52.27	53.15	81.82	65.03
		(0.00001, 0.03125)	(0.0001, 0.00001, 0.03125)	(10, 8)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1, 0.1, 32)
	20%	58.04	53.15	51.76	53.15	78.32	67.83
		(0.00001, 0.0625)	(0.0001, 0.00001, 0.03125)	(1, 1)	(1, 0.00001, 0.03125)		(1, 0.1, 32)
ozone	0%	96.58	96.58	80.00	96.58	96.58	96.58
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1, 0.1, 32)
	5%	96.58	94.58	80.67	96.58	96.58	96.58
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100, 4)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1, 0.1, 32)
	10%	96.58	96.58	96.58	96.58	96.58	90.54
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100, 0.25)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(10000, 1, 32)
	15%	96.58	96.58	95.65	96.58	96.58	86.47
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(10, 0.0625)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.001, 100, 32)
	20%	96.58	96.58	95.78	96.58	96.58	79.11
		(0.00001, 0.03125)	(0.00001, 0.00001, 0.03125)	(100, 0.03125)	(1, 0.00001, 0.03125)	(100000, 100000, 32)	(1, 10000, 32)
spambase	0%	62.20	84.79	70.65	62.27	71.32	86.67
		(0.1, 1)	(0.01, 1, 0.125)	(0.00001, 8)	(1, 1, 0.03125)	(100000, 100000, 32)	(100, 1, 32)
	5%	62.27	81.82	62.00	62.27	68.36	83.78
		(1, 0.5)	(1, 0.1, 0.0625)	(0.01, 8)	(1, 1, 0.03125)	(100000, 100000, 32)	(100, 10, 32)
	10%	66.33	80.88	79.72	62.27	66.98	73.14
		(10, 0.5)	(0.1, 0.1, 0.03125)	(100000, 8)	(1, 1, 0.03125)	(100000, 100000, 32)	(1, 1, 32)
	15%	67.05	79.44	70.87	62.27	57.64	86.60
		(1, 0.5)	(0.1, 0.1, 0.03125)	(0.1, 32)	(1, 1, 0.03125)	(100000, 100000, 32)	(10, 1, 32)
	20%	64.52	77.55	72.89	62.27	59.67	67.20
		(1, 0.5)	(0.1, 0.1, 0.03125)	(1000, 0.5)	(1, 1, 0.03125)	(100000, 100000, 32)	(1, 1, 32)
spectf	0%	80.25	79.42	80.25	82.71	82.71	75.31
		(0.00001, 0.03125)	(0.1, 0.1, 0.03125)	(0.00001, 32)	(0.001, 0.0001, 0.03125)	(100000, 100000, 32)	(1, 0.1, 32)
	5%	80.25	86.42	75.31	80.25	83.95	81.48
		(0.00001, 0.03125)	(0.1, 0.1, 0.03125)	(100, 32)	(0.001, 0.0001, 0.03125)	(100000, 100000, 32)	(0.00001, 0.1, 32)
	10%	80.25	82.42	80.25	80.25	83.95	85.19
		(0.00001, 0.03125)	(0.1, 0.1, 0.03125)	(100000, 32)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(1000, 100, 32)
	15%	80.25	86.42	80.25	80.25	80.25	80.25
		(0.00001, 0.03125)	(1, 0.1, 0.03125)	(0.00001, 8)	(0.00001, 0.00001, 0.03125)	(100000, 100000, 32)	(0.00001, 0.1, 32)
	20%	80.25	85.19	81.48	76.54	82.72	81.48
		(0.00001, 0.03125)	(1, 0.1, 0.03125)	(1, 8)	(0.00001, 100, 0.125)	(100000, 100000, 32)	(0.00001, 1, 32)

† represents the proposed model.

Table 3.13 (Continued)

DATASET	NOISE	SVM ACC(%) (c_1, μ)	TSVM ACC(%) (c_1, c_2, μ)	GBSVM ACC(%) (c_1, μ)	LSTSVM ACC(%) (c_1, c_2, μ)	GBLSTSVM† ACC(%) (c_1, c_2, μ)	LS-GBLSTSVM† ACC(%) (c_1, c_2, μ)
tic_tac_toe	0%	66.32 (0.00001, 0.03125)	95.00 (0.1, 0.01, 0.25)	95.00 (100000, 32)	68.40 (0.001, 0.01, 0.03125)	78.82 (100000, 100000, 32)	55.21 (0.1, 1000, 32)
	5%	66.32 (0.00001, 0.03125)	97.57 (0.1, 0.01, 0.03125)	92.89 (100000, 32)	66.31 (0.001, 0.01, 0.03125)	81.25 (100000, 100000, 32)	76.04 (1, 0.1, 32)
	10%	66.32 (0.00001, 0.03125)	97.92 (0.00001, 0.00001, 0.03125)	66.67 (100000, 32)	66.32 (0.001, 0.001, 0.125)	84.38 (100000, 100000, 32)	86.81 (10, 1, 32)
	15%	66.32 (0.00001, 0.03125)	93.40 (0.00001, 0.00001, 0.03125)	90.76 (100000, 32)	66.32 (0.001, 0.001, 0.125)	80.21 (100000, 100000, 32)	71.18 (1, 0.1, 32)
	20%	86.32 (0.00001, 0.03125)	92.71 (0.01, 0.01, 0.25)	87.57 (100000, 32)	66.32 (0.001, 0.001, 0.125)	87.85 (100000, 100000, 32)	66.32 (0.00001, 0.1, 32)
vehicle1	0%	75.98 (0.00001, 0.03125)	80.31 (0.01, 0.01, 0.03125)	73.62 (10000, 32)	76.37 (10, 0.00001, 0.03125)	75.59 (100000, 100000, 32)	76.77 (10, 10, 32)
	5%	76.38 (0.00001, 0.03125)	76.77 (1, 1, 0.03125)	80.89 (100000, 32)	76.38 (10000, 0.00001, 0.03125)	74.80 (100000, 100000, 32)	72.05 (10, 100, 32)
	10%	76.38 (0.00001, 0.03125)	77.17 (0.00001, 0.00001, 0.03125)	72.87 (100000, 32)	76.38 (10000, 0.00001, 0.03125)	75.98 (100000, 100000, 32)	70.47 (1000, 10, 32)
	15%	76.38 (0.00001, 0.03125)	79.13 (1, 0.01, 0.0625)	70.00 (100000, 32)	76.38 (10000, 0.00001, 0.03125)	72.83 (100000, 100000, 32)	75.20 (10, 1, 32)
	20%	76.38 (0.00001, 0.03125)	78.35 (1, 0.01, 0.0625)	76.38 (100000, 32)	76.38 (10000, 0.00001, 0.03125)	76.77 (100000, 100000, 32)	74.41 (10, 10000, 32)
yeast-0-2-5-6_vs_3-7-8-9	0%	91.39 (0.00001, 0.03125)	84.04 (0.1, 0.1, 2)	100.00 (100000, 32)	91.39 (10, 0.00001, 0.03125)	91.39 (100000, 100000, 32)	94.04 (10000, 10, 32)
	5%	91.39 (0.00001, 0.03125)	93.38 (1, 0.1, 0.125)	90.89 (100000, 32)	91.39 (10000, 0.00001, 0.03125)	92.05 (100000, 100000, 32)	88.08 (0.001, 100, 32)
	10%	91.39 (0.00001, 0.03125)	92.38 (1, 0.1, 0.125)	90.47 (100000, 32)	91.39 (10000, 0.00001, 0.03125)	92.72 (100000, 100000, 16)	92.05 (10, 1, 32)
	15%	91.39 (0.00001, 0.03125)	91.39 (1, 0.1, 0.125)	90.78 (100000, 32)	91.39 (10000, 0.00001, 0.03125)	91.39 (100000, 100000, 32)	90.40 (0.001, 1000, 32)
	20%	91.39 (0.00001, 0.03125)	91.39 (10, 0.01, 0.125)	90.87 (100000, 32)	91.39 (10000, 0.00001, 0.03125)	90.40 (100000, 100000, 32)	91.39 (0.0001, 10000, 32)
yeast-0-2-5-7-9_vs_3-6-8	0%	87.50 (0.00001, 0.03125)	53.15 (0.1, 0.00001, 0.03125)	69.48 (100000, 32)	90.72 (10, 0.00001, 0.03125)	90.72 (100000, 100000, 32)	96.03 (1, 0.1, 32)
	5%	87.50 (0.00001, 0.03125)	87.50 (0.00001, 0.00001, 0.03125)	100.00 (100000, 32)	90.73 (10000, 0.00001, 0.03125)	95.36 (100000, 100000, 16)	94.04 (10000, 10, 32)
	10%	87.50 (0.00001, 0.03125)	86.18 (0.001, 0.001, 0.03125)	95.03 (100000, 32)	90.73 (10000, 0.00001, 0.03125)	93.38 (100000, 100000, 32)	98.01 (10, 1, 32)
	15%	90.73 (0.00001, 0.03125)	87.50 (0.01, 0.001, 0.03125)	85.67 (100000, 32)	90.73 (10000, 0.00001, 0.03125)	92.38 (100000, 100000, 32)	97.02 (1000, 10, 32)
	20%	90.73 (0.00001, 0.03125)	87.50 (0.01, 0.00001, 0.03125)	89.76 (100000, 8)	90.73 (10000, 0.00001, 0.03125)	89.07 (100000, 100000, 32)	90.73 (0.0001, 10000, 32)
yeast-0-5-6-7-9_vs_4	0%	91.19 (0.00001, 0.03125)	82.45 (0.001, 0.01, 0.0625)	100.00 (1000, 2)	91.20 (10, 0.00001, 0.03125)	93.08 (100000, 100000, 32)	91.20 (0.0001, 10000, 32)
	5%	91.19 (0.00001, 0.03125)	88.68 (0.1, 0.1, 0.03125)	89.57 (1000, 2)	91.19 (10000, 0.00001, 0.03125)	88.68 (100000, 100000, 16)	90.57 (0.0001, 10000, 32)
	10%	91.19 (0.00001, 0.03125)	79.31 (0.1, 0.01, 0.125)	80.54 (1000, 2)	91.19 (10000, 0.00001, 0.03125)	91.19 (100000, 100000, 32)	77.36 (10, 1, 32)
	15%	91.19 (0.00001, 0.03125)	90.57 (0.0001, 0.00001, 0.03125)	79.34 (1000, 4)	91.19 (10000, 0.00001, 0.03125)	89.94 (100000, 100000, 16)	89.31 (10000, 10, 32)
	20%	91.19 (0.00001, 0.03125)	90.57 (0.0001, 0.00001, 0.03125)	89.43 (1000, 4)	91.19 (10000, 0.00001, 0.03125)	93.71 (100000, 100000, 32)	90.57 (0.0001, 10000, 32)
yeast-2_vs_4	0%	85.81 (0.00001, 0.03125)	94.19 (0.1, 0.1, 0.03125)	100.00 (100000, 32)	85.80 (10, 0.00001, 0.03125)	89.03 (100000, 100000, 32)	95.48 (10000, 10, 32)
	5%	85.81 (0.00001, 0.125)	94.19 (0.001, 0.01, 0.03125)	95.48 (100000, 32)	85.81 (10000, 0.00001, 0.03125)	85.81 (100000, 100000, 16)	85.81 (10, 1, 32)
	10%	85.81 (0.00001, 0.125)	82.26 (0.1, 0.01, 0.03125)	90.67 (10000, 32)	85.81 (10000, 0.00001, 0.03125)	94.84 (100000, 100000, 32)	92.90 (0.0001, 10000, 32)
	15%	85.81 (0.00001, 0.125)	90.32 (1, 0.01, 0.125)	100.00 (10000, 32)	85.81 (10000, 0.00001, 0.03125)	92.26 (100000, 100000, 32)	94.84 (1000, 10, 32)
	20%	85.81 (0.00001, 0.125)	89.68 (0.1, 0.01, 0.03125)	82.78 (100000, 32)	85.81 (10000, 0.00001, 0.03125)	89.68 (100000, 100000, 32)	92.26 (1000, 10, 32)
yeast3	0%	89.24 (0.00001, 0.03125)	92.38 (0.01, 0.1, 0.125)	60.84 (100000, 32)	88.11 (10, 0.00001, 0.03125)	88.11 (100000, 100000, 32)	92.15 (10000, 10, 32)
	5%	88.12 (0.00001, 0.03125)	90.13 (1, 0.1, 0.03125)	89.42 (100000, 2)	88.12 (10000, 0.00001, 0.03125)	90.36 (100000, 100000, 16)	89.46 (10, 1, 32)
	10%	88.12 (0.00001, 0.03125)	80.81 (0.1, 0.1, 0.03125)	90.00 (100000, 2)	88.12 (10000, 0.00001, 0.03125)	88.34 (100000, 100000, 32)	90.81 (0.0001, 10000, 32)
	15%	88.12 (0.00001, 0.03125)	87.89 (0.1, 0.1, 0.03125)	100.00 (1000, 2)	88.12 (10000, 0.00001, 0.03125)	91.70 (100000, 100000, 16)	91.48 (0.0001, 10000, 32)
	20%	88.12 (0.00001, 0.03125)	90.13 (0.1, 0.1, 0.03125)	88.87 (100000, 2)	88.12 (10000, 0.00001, 0.03125)	88.34 (100000, 100000, 32)	88.12 (10000, 10, 32)
Average Accuracy	0%	75.47	82.40	78.20	76.89	83.64	**84.55**
	5%	76.02	81.07	78.42	75.03	**82.67**	82.62
	10%	75.49	80.00	79.21	74.97	**83.06**	82.68
	15%	77.52	82.05	80.00	75.79	**84.05**	83.15
	20%	76.63	81.93	77.86	77.17	**83.11**	82.65
Average Rank	0%	4.32	3.24	3.99	3.72	2.90	**2.84**
	5%	4.28	3.44	4.04	3.68	2.81	**2.75**
	10%	4.16	3.66	3.93	3.68	**2.69**	2.88
	15%	3.91	3.44	4.28	3.78	**2.49**	3.10
	20%	4.04	3.04	4.49	3.60	**2.85**	2.97

† represents the proposed model.

Chapter 4

GRVFL-2V: Graph Random Vector Functional Link Based on Two-View Learning

In this chapter, we propose a novel model called graph random vector functional link based on two-view learning (GRVFL-2V). Moreover, we present the computational complexity and algorithm of the proposed model. We also discuss the results from experimental and statistical analyses

4.1 The proposed Graph Random Vector Functional Link Based on Two-View Learning model

In this section, we introduce a novel model called GRVFL-2V, which integrates the RVFL with MVL. Our model is structured to leverage the strengths of RVFL while incorporating both intrinsic and penalty graphical representations of multiview data through the GE framework. To derive the network output weights, the optimization process integrates subspace learning (SL) criteria, incorporating both intrinsic and penalty-based SL methodologies within the GE framework. To preserve the geometric structure effectively, we incorporate local Fisher discriminant analysis (LFDA) [74] into the GE framework. Additionally, we introduce a regularization parameter for GE to enhance the learning process further. To trade off the error between multiple views, we introduce a coupling term in the primal optimization problem of our proposed model. This term minimizes the combined error from both views, resulting in an improved generalization performance for our model. The proposed optimization problem of the GRVFL-2V model is presented below:

$$\begin{aligned}\min_{\beta_1,\beta_2} \; & \frac{c_1}{2}||\xi_1||_2^2 + \frac{c_2}{2}||\xi_2||_2^2 + \frac{c_3}{2}||\beta_1||_2^2 + \frac{1}{2}||\beta_2||_2^2 + \\ & \frac{\theta_1}{2}||G_1^{1/2}\beta_1||_2^2 + \frac{\theta_2}{2}||G_2^{1/2}\beta_2||_2^2 + \rho\xi_1^t\xi_2 \\ \text{s.t. } \; & Z_1\beta_1 - Y_{true} = \xi_1 \text{ and } Z_2\beta_2 - Y_{true} = \xi_2,\end{aligned} \tag{4.1}$$

$$\text{where } Z_1 = \begin{bmatrix} X_A & H_A \end{bmatrix} \text{ and } Z_2 = \begin{bmatrix} X_B & H_B \end{bmatrix}. \tag{4.2}$$

$H_A = \phi(X_AW_A + B_A)$ and $H_B = \phi(X_BW_B + B_B)$, where $\phi(\cdot)$ is a non-linear activation function. W_A and B_A are the randomly initialized weight and bias matrices for $view-A$ and W_B and B_B for $view-B$, respectively. The primal formulation (4.1) has the following components:

1. Connections between the concatenated matrix Z_1 and the output layer are established by the weight matrix β_1, while the weight matrix β_2 establishes connections between the Z_2 and the output layer. Minimizing $||\beta_1||_2^2$ and $||\beta_2||_2^2$ incorporates the structural risk minimization (SRM) principle.

2. G_1 and G_2 represent the graph embedding matrices, and minimizing $||G_1^{1/2}\beta_1||_2^2$ and $||G_2^{1/2}\beta_2||_2^2$ facilitates the preservation of structural relationships between data points.

3. ξ_1 and ξ_2 represent the empirical errors in $view-A$ and $view-B$, respectively. Minimizing $||\xi_1||_2^2$ and $||\xi_2||_2^2$ leads to the reduction of empirical errors in both views.

4. The term $\xi_1^t\xi_2$ acts as a coupling term that integrates information from both views and promotes the simultaneous minimization of errors in both views.

5. The variables θ_1 and θ_2 are graph regularization parameters, while c_1, c_2, and c_3 represent the regularization parameters and ρ is the coupling parameter.

The lagrangian of (4.1) is:

$$
\begin{aligned}
L = \frac{c_1}{2}||\xi_1||^2 + \frac{c_2}{2}||\xi_2||^2 + \frac{c_3}{2}||\beta_1||^2 + \frac{1}{2}||\beta_2||^2 + \\
\frac{\theta_1}{2}||G_1^{1/2}\beta_1||^2 + \frac{\theta_2}{2}||G_2^{1/2}\beta_2||^2 + \rho\xi_1^t\xi_2 \\
-{\alpha_1}^t(Z_1\beta_1 - Y_{true} - \xi_1) - {\alpha_2}^t(Z_2\beta_2 - Y_{true} - \xi_2).
\end{aligned} \tag{4.3}
$$

Partially differentiating w.r.t. $\xi_1, \xi_2, \beta_1, \beta_2, \alpha_1$, and α_2 we get

$$\frac{\partial L}{\partial \xi_1} = c_1\xi_1 + \rho\xi_2 + \alpha_1 = 0, \tag{4.4}$$

$$\frac{\partial L}{\partial \xi_2} = c_2\xi_2 + \rho\xi_1 + \alpha_2 = 0, \tag{4.5}$$

$$\frac{\partial L}{\partial \beta_1} = c_3\beta_1 + \theta_1 G_1\beta_1 - {Z_1}^t\alpha_1 = 0, \tag{4.6}$$

$$\frac{\partial L}{\partial \beta_2} = \beta_2 + \theta_2 G_2\beta_2 - {Z_2}^t\alpha_2 = 0, \tag{4.7}$$

$$\frac{\partial L}{\partial \alpha_1} = Z_1\beta_1 - Y_{true} - \xi_1 = 0, \tag{4.8}$$

$$\frac{\partial L}{\partial \alpha_2} = Z_2\beta_2 - Y_{true} - \xi_2 = 0. \tag{4.9}$$

Substituting (4.4) and (4.5) in (4.6) and (4.7) respectively, we get

$$c_3\beta_1 + \theta_1 G_1\beta_1 + {Z_1}^t(c_1\xi_1 + \rho\xi_2) = 0, \tag{4.10}$$

$$\beta_2 + \theta_2 G_2\beta_2 + {Z_2}^t(c_2\xi_2 + \rho\xi_1) = 0. \tag{4.11}$$

Substituting the value of (4.8) and (4.9) in (4.10) and (4.11), we get

$$
\begin{aligned}
c_3\beta_1 + \theta_1 G_1\beta_1 + {Z_1}^t\left(c_1(Z_1\beta_1 - Y_{true}) + \rho(Z_2\beta_2 - Y_{true})\right) = 0, \\
\beta_2 + \theta_2 G_2\beta_2 + {Z_2}^t\left(c_2(Z_1\beta_1 - Y_{true}) + \rho(Z_2\beta_2 - Y_{true})\right) = 0.
\end{aligned}
$$

On simplifying, we get

$$
\begin{aligned}
c_3\beta_1 + \theta_1 G_1\beta_1 + c_1{Z_1}^t Z_1\beta_1 + \rho{Z_1}^t Z_2\beta_2 = {Z_1}^t(c_1 + \rho)Y_{true}, \\
\beta_2 + \theta_2 G_2\beta_2 + c_2{Z_2}^t Z_2\beta_2 + \rho{Z_2}^t Z_1\beta_1 = {Z_2}^t(c_2 + \rho)Y_{true}.
\end{aligned}
$$

Finally, we get

$$
\begin{aligned}
(c_3 I_1 + \theta_1 G_1 + c_1{Z_1}^t Z_1)\beta_1 + (\rho{Z_1}^t Z_2)\beta_2 = {Z_1}^t(c_1 + \rho)Y_{true}, \\
(\rho{Z_2}^t Z_1)\beta_1 + (I_2 + \theta_2 G_2 + c_2{Z_2}^t Z_2)\beta_2 = {Z_2}^t(c_2 + \rho)Y_{true},
\end{aligned}
$$

where I_1 and I_2 are the identity matrices of the appropriate dimension. Converting into matrix form, we get

$$
\begin{bmatrix} c_3 I_1 + \theta_1 G_1 + c_1{Z_1}^t Z_1 & \rho{Z_1}^t Z_2 \\ \rho{Z_2}^t Z_1 & I_2 + \theta_2 G_2 + c_2{Z_2}^t Z_2 \end{bmatrix}
\begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix}
= \begin{bmatrix} {Z_1}^t(c_1 + \rho) \\ {Z_2}^t(c_2 + \rho) \end{bmatrix} Y_{true},
$$

$$\begin{bmatrix}\beta_1\\ \beta_2\end{bmatrix} = \begin{bmatrix} c_3I_1+\theta_1G_1+c_1{Z_1}^tZ_1 & \rho {Z_1}^tZ_2 \\ \rho {Z_2}^tZ_1 & I_2+\theta_2G_2+c_2{Z_2}^tZ_2\end{bmatrix}^{-1}\begin{bmatrix}{Z_1}^t(c_1+\rho)\\ {Z_2}^t(c_2+\rho)\end{bmatrix}Y_{true}. \tag{4.12}$$

For a new data point x having representation x_A and x_B w.r.t. $view-A$ and $view-B$, respectively, we give the classification function as follows:

$$class(x) = \arg\max_{i\in\{1,2\}}\{y_{c_i}\}, \tag{4.13}$$

where

$$y_c = \frac{1}{2}\left([x_A \;\; \phi(x_AW_A+b_A)]\beta_1 + [x_B \;\; \phi(x_BW_B+b_B)]\beta_2\right)$$

and $y_c = (y_{c_1}, y_{c_2})$.

where W_A and W_B are the randomly generated weights matrices, and b_A and b_B represents the bias column vectors of the bias matrices B_A and B_B, respectively. Since, all the columns of the bias matrix B_A are identical, hence b_A can be chosen to be any column of B_A. A similar argument follows for b_B. The algorithm for the proposed GRVFL-2V is presented in Algorithm 2.

4.1.1 LFDA Under the GE Framework

The intrinsic as well as the penalty graphs are constructed using concatenated matrices Z_1 and Z_2. Specifically, for Z_1, we have $\mathcal{G}_1^{int} = \{Z_1, {}_1\Delta^{int}\}$ and $\mathcal{G}_1^{pen} = \{Z_1, {}_1\Delta^{pen}\}$. Similarly for Z_2, we have $\mathcal{G}_2^{int} = \{Z_2, {}_2\Delta^{int}\}$ and $\mathcal{G}_1^{pen} = \{Z_2, {}_2\Delta^{pen}\}$. Hence, the intrinsic graph is denoted as $G_{int}^1 = {Z_1}^t\mathbb{L}_1Z_1$ and the penalty graph as $G_{pen}^1 = {Z_1}^t\mathbb{U}_1Z_1$. Similarly, for Z_2, the intrinsic graph is $G_{int}^2 = {Z_2}^t\mathbb{L}_2Z_2$ and the penalty graph is $G_{pen}^2 = {Z_2}^t\mathbb{U}_2Z_2$. Within the graph embedding (GE) framework, we employ the weighting scheme of Local Fisher Discriminant Analysis (LFDA) [74]. Consequently, the weights for the LFDA model's intrinsic and penalty graphs are determined as follows:

$${}_i\Delta_{kl}^{int} = \begin{cases} \frac{\lambda_{kl}}{N_{c_k}}, & c_k = c_l \\ 0, & otherwise. \end{cases} \tag{4.14}$$

$${}_i\Delta_{kl}^{pen} = \begin{cases} \lambda_{kl}(\frac{1}{N} - \frac{1}{N_{c_k}}), & c_k = c_l \\ \frac{1}{N}, & otherwise. \end{cases} \tag{4.15}$$

Where $i = 1, 2$. In this context, N_{c_k} denotes the number of samples within the class labeled as c_k, while λ_{kl} quantifies the similarity between h_k and h_l, where $h_k, h_l \in Z_i (i = 1, 2)$. The kernel function is utilized in this paper to compute the similarity measure, i.e., $\lambda_{kl} = exp(\frac{-||h_k - h_l||^2}{2\sigma^2})$ where σ is a scaling parameter.

Algorithm 2 GRVFL-2V Model Algorithm

Input: Training datasets X_A and X_B.

Output: GRVFL-2V model.

1: Given the values for $c_1, c_2, c_3, \theta_1, \theta_2$, and ρ.
2: Compute Z_1 and Z_2 using Equation (4.2).
3: Compute intrinsic and penalty graph weights using Equations (4.14) and (4.15) for Z_1 and Z_2.
4: Calculate Laplacian matrices $\mathbb{L}_i = \mathbb{D}_i - {}_i\Delta^{int}$ and $\mathbb{U}_i = \mathbb{L}_i^p = \mathbb{D}_i^p - {}_i\Delta^{pen}$ for $Z_i (i = 1, 2)$.
5: Compute $G_{int}^1 = {Z_1}^t \mathbb{L}_1 Z_1$ and $G_{pen}^1 = {Z_1}^t \mathbb{U}_1 Z_1$ for Z_1, and $G_{int}^2 = {Z_2}^t \mathbb{L}_2 Z_2$ and $G_{pen}^2 = {Z_2}^t \mathbb{U}_2 Z_2$ for Z_2.
6: Compute $G_1 = (G_{pen}^1)^{-1} G_{int}^1$ and $G_2 = (G_{pen}^2)^{-1} G_{int}^2$.
7: Use Equation (4.12) to calculate β_1 and β_2.
8: Use test condition (4.13) to classify a new data point.

4.2 Computational Complexity

We analyze the computational complexity of our proposed GRVFL-2V model in this section. For X_A, computing the graph embedding (GE) matrix G_1 using the methodologies from [91] which account for intrinsic as well as penalty graph structures results in a time complexity of $\mathcal{O}((m+h_1)^3 + (m+h_1)^2 l)$. Similarly, for X_B, computing G_2 yields a complexity of $\mathcal{O}((n+h_2)^3 + (n+h_2)^2 l)$. Thus, the GE process incurs a total computational complexity of $\mathcal{O}((n+h_1)^3 + (n+h_1)^2 l) + \mathcal{O}((m+h_2)^3 + (m+h_2)^2 l)$. To solve our model, we address (4.12). The complexity of this step is primarily governed by the inversion of a square matrix with an order of $(m+n+h_1+h_2)$, leading to a computational complexity of $\mathcal{O}((m+n+h_1+h_2)^3)$. Consequently, the total computational complexity of our proposed model becomes $\mathcal{O}((n+h_1)^3 + (n+h_1)^2 l) + \mathcal{O}((m+h_2)^3 + (m+h_2)^2 l) + \mathcal{O}((m+n+h_1+h_2)^3) \approx \mathcal{O}((m+n+h_1+h_2)^3)$.

4.3 Experiments, Results, and Discussions

The performance of the proposed model has been evaluated against the baseline models: SVM2K [82], MvTSVM [83], ELM1 (performance of ELM [11] on

'$view - A$'), ELM2 (performance of ELM [11] on '$view - B$'), RVFL1 (performance of RVFL [6] on '$view - A$') and RVFL2 (performance of RVFL [6] on '$view - B$'), and MVLDM [92]. The datasets used for the evaluation are UCI [70], KEEL [71], Animal with Attributes (AwA)[1], and Corel5K[2].

4.3.1 Experimental Setup

The experiments are carried out on a PC equipped with an Intel(R) Xeon(R) Gold 6226R processor clocked at 2.90GHz and 128 GB of RAM. The system runs on Windows 11 and uses Python 3.11. The dual of the QPP in SVM2K [82] and MvTSVM [83] is solved using the "QP solvers" function from the CVXOPT package. The dataset is randomly divided, allocating 70% of the samples for training and 30% for testing. Hyperparameters are optimized and validated using a five-fold cross-validation approach. The regulazation parameters c_i $(i = 1, 2, 3)$, the graph regularization parameters θ_j $(j = 1, 2)$, and the coupling parameter ρ are tuned within the range $\{10^{-5}, 10^{-4}, \cdots, 10^{5}\}$. In our experiments, we have taken $c_1 = c_2 = c_3$ and $\theta_1 = \theta_2$. All the hyperparameters of the baseline models were also taken within the same range. The hidden neurons (h_l) varies as 3:20:203.

4.3.2 Experiments on UCI and KEEL Datasets

Within this subsection, we delve into analyzing the statistical significance of the results obtained from our experiments, specifically concentrating on datasets sourced from UCI and KEEL repositories [70, 71]. Through our assessment, we encompass a total of 27 datasets. Given the absence of inherent multi-view characteristics in the UCI and KEEL datasets, we designated the 95% principal component extracted from the original data as '$view - B$', while the unaltered data itself acts as '$view - A$'. In order to thoroughly compare the performance levels between our proposed graph random vector functional link based on the two-view learning (GRVFL-2V) model and the baseline models with optimized hyperparameters, the outcomes are illustrated in Table 4.1.

[1] http://attributes.kyb.tuebingen.mpg.de

[2] https://wang.ist.psu.edu/docs/related/

Table 4.1: Performance comparison of the proposed GRVFL-2V along with the baseline models based on classification accuracy for UCI and KEEL datasets.

Dataset	SVM2K [82] (C_1)	MvTSVM [83] (C_1, C_2, D)	ELM1 [11] (C, N)	ELM2 [11] (C, N)	RVFL1 [6] (C, N)	RVFL2 [6] (C, N)	MVLDM [92] ($C_1, \nu_1, \nu_2, \sigma$)	GRVFL-2V† (C_1, θ, ρ, N)
aus	87.02	71.15	85.98	86.06	85.98	85.98	71.98	87.50
	(0.001)	(0.00001, 0.00001, 0.00001)	(0.01, 123)	(0.001, 163)	(0.001, 163)	(10, 23)	(0.001, 0.01, 0.1, 0.25)	(1, 0.0001, 0.00001, 63)
bank	80.74	71.86	85.54	85.39	85.83	85.61	73.67	89.98
	(0.001)	(0.00001, 0.00001, 0.00001)	(0.00001, 3)	(10000, 23)	(1000, 163)	(10, 123)	(0.1, 0.001, 0.001, 4)	(100, 0.1, 0.0001, 203)
breast_cancer	62.45	55.58	69.77	65.12	67.44	66.28	70	72.09
	(0.001)	(0.00001, 0.00001, 0.00001)	(1, 83)	(0.01, 43)	(0.0001, 3)	(1, 83)	(0.0001, 0.1, 0.001, 4)	(10, 0.1, 0.0001, 23)
breast_cancer_wisc	90.04	81.43	92.1	92.1	92.14	95.1	75	98.57
	(0.00001)	(0.00001, 0.00001, 0.00001)	(0.0001, 203)	(0.01, 63)	(0.00001, 23)	(0.0001, 203)	(0.0001, 0.0001, 0.00001, 0.25)	(0.00001, 10, 0.00001, 163)
breast_cancer_wisc_diag	95.49	88.6	92.42	97.08	95.83	96.49	93.15	98.25
	(0.001)	(0.00001, 0.00001, 0.00001)	(0.01, 163)	(0.001, 163)	(0.001, 163)	(100, 3)	(0.001, 0.0001, 1, 4)	(1000, 100000, 100, 103)
breast_cancer_wisc_prog	58.33	58.33	68.33	68.33	73.33	68.33	71.17	75.00
	(0.001)	(0.00001, 0.00001, 0.00001)	(0.01, 203)	(0.01, 83)	(100000, 3)	(0.01, 203)	(0.0001, 0.0001, 0.1, 2)	(100000, 10, 10000, 23)
brwisconsin	97.56	61.95	97.07	96.1	97.07	96.59	95.59	97.07
	(0.00001)	(0.00001, 0.00001, 0.00001)	(0.0001, 143)	(0.001, 63)	(0.001, 63)	(100000, 23)	(0.001, 0.00001, 0.1, 4)	(0.001, 1000, 0.00001, 103)
bupa or liver-disorders	54.8	42.31	63.46	65.38	63.46	66.35	55.34	69.23
	(0.00001)	(0.00001, 0.00001, 0.00001)	(1000, 23)	(0.1, 163)	(0.1, 163)	(0.1, 163)	(0.001, 0.001, 0.001, 4)	(100000, 0.01, 100, 23)
checkerboard_Data	87.02	43.75	86.98	86.06	86.98	86.98	84.06	87.50
	(0.00001)	(0.00001, 0.00001, 0.00001)	(0.01, 123)	(0.001, 163)	(0.001, 163)	(10, 23)	(0.001, 0.001, 0.001, 1)	(1, 0.0001, 0.00001, 63)
chess_krvkp	80.45	82.35	95.62	93.33	95.62	94.68	97.7	96.98
	(0.001)	(0.00001, 0.00001, 0.00001)	(100, 203)	(100000, 203)	(10000, 203)	(100, 203)	(0.1, 0.1, 0.1, 4)	(0.1, 0.0001, 0.1, 203)
cleve	80	75.56	80	85.56	81.11	81.11	84.27	84.44
	(0.1)	(0.00001, 0.00001, 0.00001)	(0.001, 103)	(0.001, 123)	(0.01, 3)	(0.01, 23)	(0.01, 0.0.1, 0.1, 1)	(0.01, 10, 0.0001, 23)
cmc	64.25	55.88	69.91	70.14	68.1	71.72	74.38	72.17
	(0.00001)	(0.00001, 0.00001, 0.00001)	(0.01, 143)	(0.01, 183)	(100, 63)	(1000, 43)	(0.0001, 0.01, 0.001, 4)	(1000, 1, 0.001, 23)
conn_bench_sonar_mines_rocks	80.95	46.03	80.54	74.6	88.54	73.02	75.81	77.78
	(1000)	(0.00001, 0.00001, 0.00001)	(10, 3)	(0.01, 203)	(0.01, 203)	(0.1, 183)	(0.0001, 0.00001, 10, 0.5)	(10, 1000, 0.1, 183)
cylinder_bands	68.18	60.39	76.62	74.03	72.08	74.68	71.9	74.68
	(0.00001)	(0.00001, 0.00001, 0.00001)	(0.001, 203)	(100, 143)	(0.001, 183)	(1, 23)	(0.0001, 0.1, 0.001, 4)	(100, 0.01, 0.0001, 43)
fertility	75.25	75	90	90	90	90	86.67	90.00
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.01, 83)	(0.01, 63)	(0.01, 63)	(0.01, 83)	(0.001, 0.001, 100, 2)	(0.01, 0.0001, 0.0001, 23)
hepatitis	80.85	78.72	78.85	80.85	80.85	78.72	78.26	80.85
	(0.00001)	(0.00001, 0.00001, 0.00001)	(1, 23)	(0.01, 183)	(0.1, 123)	(10, 3)	(0.001, 0.01, 0.001, 4)	(1000, 100000, 1000, 163)
hill_valley	60.98	53.3	68.05	51.92	68.78	51.92	56.2	77.75
	(0.00001)	(0.00001, 0.00001, 0.00001)	(10000, 103)	(1, 103)	(10, 103)	(10000, 103)	(0.0001, 0.0001, 0.1, 1)	(0.01, 0.1, 0.1, 143)
mammographic	80.27	77.06	80.28	82.01	80.28	81.66	83.33	83.74
	(0.01)	(0.00001, 0.00001, 0.00001)	(1, 83)	(100000, 63)	(100000, 23)	(100, 23)	(0.0001, 0.001, 1, 4)	(100000, 0.00001, 10000, 23)
monks_3	80.24	76.11	95.21	95.41	95.21	95.81	96.39	97.00
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.1, 143)	(0.1, 163)	(0.1, 163)	(0.1, 123)	(0.01, 0.001, 10, 4)	(0.1, 1, 0.00001, 183)
new-thyroid1	78.46	82.31	98.46	96.92	98.46	96.92	95.31	98.46
	(0.1)	(0.00001, 0.00001, 0.00001)	(10, 103)	(1, 43)	(1, 23)	(10, 103)	(0.0001, 1000, 0.1, 0.25)	(0.1, 100000, 0.001, 3)
oocytes_merluccius_nucleus_4d	74.27	64.82	82.41	81.11	83.71	80.78	75.16	83.39
	(0.00001)	(0.00001, 0.00001, 0.00001)	(0.1, 83)	(1, 143)	(1, 143)	(0.1, 83)	(0.1, 100, 0.1, 4)	(100000, 1, 10, 123)
oocytes_trisopterus_nucleus_2f	78.83	58.39	86.13	78.83	85.04	78.83	82.05	84.67
	(0.0001)	(0.00001, 0.00001, 0.00001)	(0.01, 103)	(0.1, 143)	(100000, 43)	(0.01, 103)	(0.1, 0.001, 1, 4)	(10000, 1000, 100, 83)
parkinsons	71.19	71.19	81.36	77.97	89.83	77.97	93.1	83.05
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.1, 163)	(1, 63)	(0.1, 203)	(0.1, 163)	(0.0001, 0.00001, 1, 2)	(100, 0.1, 1, 143)
pima	76.19	73.33	74.89	74.03	74.46	74.03	69.13	76.19
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.1, 63)	(0.01, 143)	(0.01, 143)	(0.1, 63)	(0.0001, 10, 10, 4)	(0.001, 1, 0.01, 3)
pittsburg_bridges_T_OR_D	70.85	65.85	89.32	93.55	80.65	80.65	90	90.32
	(100)	(0.00001, 0.00001, 0.00001)	(0.01, 83)	(0.01, 43)	(1000, 3)	(0.01, 3)	(0.1, 0.00001, 1, 4)	(0.00001, 1, 0.01, 163)
planning	65.85	63.64	76.36	76.36	76.36	76.36	68.52	90.32
	(10)	(0.00001, 0.00001, 0.00001)	(0.01, 23)	(0.00001, 3)	(0.00001, 23)	(0.01, 23)	(0.00001, 0.001, 1, 1)	(0.00001, 1, 0.01, 63)
ripley	89.07	80.67	86.13	87.53	87.53	87.53	89.07	89.60
	(100000)	(0.00001, 0.00001, 0.00001)	(10000, 103)	(100, 203)	(10000, 123)	(10000, 103)	(0.001, 0.001, 0.001, 4)	(0.1, 0.001, 0.0001, 103)
Average ACC	76.65	67.24	82.66	81.7	83.14	81.26	79.9	**85.23**
Average Rank	5.52	7.69	4.02	4.30	3.54	4.41	4.80	**1.74**

† represents the proposed model.

The baseline models SVM2K, MvTSVM, ELM1, EML2, RVFL1, RVFL2, and MVLDM achieves the average ACC of $75, 24\%$, 65.83%, 82.11%, 80.94%, 82.62%, 80.26%, and 80.58%, respectively. The GRVFL-2V model that we put forward exhibited an outstanding average accuracy rate of 84.35%, surpassing the performance levels of the baseline models. The difference in average ACC of the proposed model and the baseline models lies approximately between $4\% - 20\%$. This showcases the superior generalization capabilities inherent in our proposed model compared to the baseline models.

In order to further assess the effectiveness of the proposed model, a ranking method is utilized. This method involves assigning a rank to each model for every dataset, where the model that performs the best is given the lowest rank, and the model that performs the worst is given the highest rank. The average rank for each model is then calculated by finding the mean of its ranks across all datasets. If there are a total of N datasets, each evaluated with λ models, the rank of the p^{th} model on the t^{th} dataset can be represented as s_p^t. Subsequently, the average rank of the p^{th} model is calculated as $R_p = \frac{1}{N}\sum_{t=1}^{N} s_p^t$. The proposed GRVFL-2V model has been able to achieve an average rank of 1.74. On the other hand, the baseline models such as SVM2K, MvTSVM, ELM1, ELM2, RVFL1, RVFL2, and MVLDM have average ranks of 5.52, 7.69, 4.02, 4.30, 3.54, 4.41, and 4.80, respectively. These results clearly demonstrate the superiority of the proposed model over the baseline models.

In order to assess the statistical significance of the proposed model, the Friedman test is utilized as outlined in [90]. This particular test is designed to pinpoint noteworthy variations among the models being compared by scrutinizing their average ranks. The underlying assumption of the null hypothesis is that all models showcase an identical average rank, indicating an equivalent level of performance. The Friedman test adheres to a chi-squared distribution denoted as χ_F^2 with $(\lambda - 1)$ degrees of freedom, which can be calculated using the formula: $\chi_F^2 = \frac{12N}{\lambda(\lambda+1)}\left[\sum_p {R_p}^2 - \frac{\lambda(\lambda+1)^2}{4}\right]$. Here, R_p signifies the average rank of the pth model, λ represents the total number of models, and N denotes the number of datasets involved in the analysis. The Friedman statistic F_F is determined by the formula: $F_F = \frac{(N-1)\chi_F^2}{N(\lambda-1)-\chi_F^2}$, where the F-distribution is characterized by $(\lambda - 1)$ and $(\lambda - 1) \times (N - 1)$ degrees of freedom. Given that $N = 27$ and $\lambda = 8$, the values of χ_F^2 and F_F are calculated to be 91.37 and 24.33, respectively. By consulting the F-distribution table at a 5% level of significance (α), the critical value of $F_F(7, 182) = 2.39$ is obtained. Since the calculated value of F_F is 24.33, which exceeds the critical value of 2.39, the null hypothesis is rejected. This outcome underscores a substantial statistical distinction among the models under comparison. In addition, the Nemenyi

Table 4.2: The statistical comparison of the proposed GRVFL-2V model with the baseline models on UCI and KEEL datasets using the Nemenyi post hoc test.

	SVM2K [82]	MvTSVM [83]	ELM1 [11]	ELM2 [11]	RVFL1 [6]	RVFL2 [6]	MVLDM [92]
GRVFL-2V (proposed)	✓	✓	✓	✓	✓	✓	✓

✓ indicates that the model listed in the row is statistically superior to the model mentioned in the column.

post hoc test [90] is utilized to delve deeper into the differences between the models. If the average ranks of the models exhibit a variance of at least the critical difference (CD), then they are deemed to be significantly distinct. The CD is determined by the formula: $CD = q_\alpha \left[\frac{\lambda(\lambda+1)}{6N}\right]^{1/2}$. On calculating, we get the $CD = 1.35$. The results presented in Table 4.2 provide clear evidence that the average rank difference between the baseline model and the proposed GRFVL-MVL is larger than the critical difference (CD) value. To be more precise, the average rank difference between MvTSVM and GRVFL-2V is 5.95, which is greater than the CD value of 1.35. Similarly, the average rank difference between MVLDM and GRVFL-2V is 3.06, also surpassing the CD value. Therefore, these results unequivocally demonstrate that the proposed model exhibits significant dissimilarity compared to the baseline models.

Furthermore, a pairwise win-tie-loss sign test is conducted under the assumption of equal performance between the two models under the null hypothesis. It is anticipated that each model will emerge victorious in roughly half of the total datasets, denoted as $\frac{N}{2}$, where N represents the total number of datasets. To establish statistical significance, a model must secure wins in approximately $\frac{N}{2} + 1.96\frac{\sqrt{N}}{2}$ datasets more than its counterpart. In scenarios where there is an even number of ties between the models, the ties are

Table 4.3: Pairwise win-tie-loss-sign test of proposed GRVFL-2V model and baseline models on UCI and KEEL datasets.

	SVM2K [82]	MvTSVM [83]	ELM1 [11]	ELM2 [11]	RVFL1 [6]	RVFL2 [6]	Large_Margin [92]
MvTSVM [83]	[2, 2, 23]						
ELM1 [11]	[18, 1, 8]	[27, 0, 0]					
ELM2[11]	[18, 2, 7]	[26, 0, 1]	[10, 4, 13]				
RVFL1 [6]	[21, 1, 5]	[27, 0, 0]	[11, 10, 6]	[14, 4, 9]			
RVFL2 [6]	[18, 1, 8]	[25, 1, 1]	[9, 5, 13]	[10, 9, 8]	[7, 7, 13]		
Large_Margin [92]	[16, 1, 10]	[24, 0, 3]	[11, 0,16]	[11, 0, 16]	[9, 0, 18]	[13, 0, 14]	
GRVFL-2V (proposed)	[25, 0, 2]	[27, 0, 0]	[22, 1, 4]	[23, 1, 3]	[21, 1, 5]	[24, 1, 2]	[25, 0, 2]

wherein $\begin{bmatrix} x & y & z \end{bmatrix}$, x signifies no. of wins, y no. of draws, and z no. of losses.

evenly distributed. Conversely, in cases of an odd number of ties, one tie is discounted, and the remaining ties are divided between the models. With N set at 27, a minimum of 18.59 wins is required to establish a significant difference between the two models. The results depicted in Table 4.3 showcase a distinct advantage for our proposed model over the baseline model. Clearly, we can see that our proposed model wins on at least 21 (> 18.59) datasets out of 27 datasets. Hence, our proposed model is far superior to the baseline models. In particular, we can observe that our proposed GRVFL-2V surpasses SVM2K in 25 datasets out of the 27 datasets. Furthermore, it outperforms the MvTSVM in all 27 datasets while excelling against MVLDM in 25 datasets out of the 27. The findings unequivocally affirm that our proposed model exhibits a significantly higher performance level compared to the baseline models.

4.3.3 Experiments on Corel5k Datasets

The Corel5k[3] dataset serves as a prominent benchmark in the realms of computer vision and image processing. Comprising 5,000 images across 50 distinct categories, each category includes 100 images. This dataset is widely used for a variety of applications, including content-based image retrieval (CBIR), object recognition, and image classification. We divide the dataset into 50 binary datasets using the one-versus-rest approach. For every binary dataset, 100 photographs from the target category make up the positive class, while another 100 images are chosen at random from the other categories to make up the negative class. This approach facilitates the creation of tailored binary datasets for focused analysis and experimentation. The results presented in Table 4.4 provide clear evidence that our proposed model outperforms the baseline models. With an average accuracy (ACC) of 77.33%, our model achieves the highest accuracy among all the models compared. Additionally, the proposed GRVFL-2V model obtains the lowest average rank of 2.94, indicating its superior performance compared to the baseline models. These findings are from the 50 datasets from Corel5k, which further solidifies the superiority of our proposed model.

Table 4.4: Performance comparison of the proposed GRVFL-2V along with the baseline models based on classification accuracy for Corel5k datasets.

Dataset	SVM2K [82]	MvTSVM [83]	ELM1 [11]	ELM2 [11]	RVFL1 [6]	RVFL2 [6]	MVLDM [92]	GRVFL-2V†
	(C_1)	(C_1, C_2, D)	(C, N)	(C, N)	(C, N)	(C, N)	($C_1, \nu_1, \nu_2, \sigma$)	(C_1, θ, ρ, N)
1000	81.67	50	83.33	78.33	83.33	76.67	73.33	83.33
	(0.0625)	(0.00001, 0.00001, 0.00001)	(0.0001, 183)	(0.00001, 43)	(0.0001, 183)	(0.00001, 3)	(0.01, 0.001, 0.001, 4)	(0.01, 10, 0.001, 23)
10000	71.67	48.33	56.67	63.33	71.67	71.67	55	65.00
	(0.03125)	(0.00001, 0.00001, 0.00001)	(1000, 23)	(1000, 23)	(0.01, 3)	(0.00001, 63)	(0.01, 0.01, 0.001, 4)	(1, 10, 0.01, 183)
100000	71.67	55	63.33	73.33	78.33	75	61.67	78.33
	(0.0625)	(0.00001, 0.00001, 0.00001)	(100000, 183)	(0.00001, 123)	(0.001, 183)	(0.00001, 123)	(100, 1000, 100, 4)	(0.1, 100, 0.01, 3)
101000	66.67	58.33	75	73.33	68.33	76.67	55	80.00
	(0.0625)	(0.00001, 0.00001, 0.00001)	(0.001, 123)	(0.00001, 143)	(0.001, 3)	(0.00001, 143)	(100, 10, 100, 4)	(1000, 10000, 10, 23)
102000	81.67	51.67	81.67	75	81.67	71.67	83.33	81.67
	(0.03125)	(0.00001, 0.00001, 0.00001)	(0.01, 63)	(0.00001, 143)	(0.01, 3)	(100000, 3)	(0.001, 0.001, 0.0001, 4)	(0.1, 100, 0.00001, 3)
103000	73.33	45	80	75	81.67	73.33	65	80.00
	(0.03125)	(0.00001, 0.00001, 0.00001)	(0.001, 163)	(0.0001, 43)	(0.001, 163)	(0.00001, 43)	(0.0001, 0.1, 0.001, 4)	(1000, 100000, 0.0001, 43)
104000	71.67	46.67	55	65	76.67	76.67	46.67	51.67
	(32)	(0.00001, 0.00001, 0.00001)	(10, 123)	(10000, 23)	(0.0001, 203)	(0.00001, 83)	(0.0001, 0.0001, 0.00001, 4)	(100, 0.1, 1000, 83)
108000	80	48.33	75.65	76.67	78.33	76.67	78.33	81.67
	(8)	(0.00001, 0.00001, 0.00001)	(0.001, 63)	(0.00001, 83)	(0.1, 3)	(0.00001, 3)	(100, 1000, 10000, 4)	(100, 100000, 0.01, 23)
109000	65	43.33	68.33	73.33	68.33	75	73.33	88.33
	(1)	(0.00001, 0.00001, 0.00001)	(0.0001, 203)	(100, 63)	(0.0001, 203)	(0.0001, 43)	(0.001, 0.001, 0.0001, 0.25)	(1, 100, 0.001, 23)
113000	66.67	45	68.33	63.33	70	63.33	66.67	66.67
	(0.03125)	(0.00001, 0.00001, 0.00001)	(0.001, 123)	(1000, 203)	(0.001, 123)	(0.0001, 23)	(0.0001, 0.001, 0.001, 0.25)	(100000, 10000, 0.00001, 3)
118000	58.33	48.33	68.33	61.67	68.33	56.67	58.33	60.00
	(8)	(0.00001, 0.00001, 0.00001)	(0.001, 103)	(100000, 43)	(0.001, 103)	(1000, 43)	(0.01, 0.1, 0.01, 4)	(100000, 100, 1, 183)
119000	71.33	53.33	81.67	83.33	78.33	80	70	80.00
	(16)	(0.00001, 0.00001, 0.00001)	(0.001, 143)	(0.00001, 123)	(0.01, 63)	(0.00001, 83)	(0.001, 0.0001, 0.01, 0.25)	(0.0001, 0.01, 0.00001, 23)
12000	60	48.33	63.33	58.33	63.33	51.67	61.67	68.33
	(32)	(0.00001, 0.00001, 0.00001)	(0.0001, 83)	(0.00001, 63)	(0.0001, 83)	(0.00001, 43)	(0.1, 0.1, 0.1, 2)	(0.00001, 10000, 1000, 203)
120000	73.33	48.33	78.33	80	78.33	85	66.67	75.00
	(0.03125)	(0.00001, 0.00001, 0.00001)	(0.01, 203)	(0.00001, 203)	(0.01, 203)	(0.00001, 103)	(0.001, 0.0001, 0.001, 4)	(0.0001, 10, 100, 203)
121000	63.33	55	68.33	75	71.67	71.67	71.67	75.00
	(2)	(0.00001, 0.00001, 0.00001)	(0.0001, 163)	(0.00001, 103)	(0.001, 143)	(0.00001, 103)	(0.001, 0.1, 0.01, 2)	(0.0001, 0.01, 0.00001, 3)
122000	78.33	55	70	76.67	70	71.67	61.67	81.67
	(0.0625)	(0.00001, 0.00001, 0.00001)	(0.01, 163)	(100, 23)	(0.01, 163)	(0.00001, 123)	(0.01, 0.001, 0.01, 0.25)	(0.0001, 0.1, 0.00001, 23)
13000	85	50	83.33	91.67	85	90	78.33	85
	(32)	(0.00001, 0.00001, 0.00001)	(0.001, 183)	(0.00001, 103)	(0.001, 183)	(0.00001, 43)	(0.01, 0.001, 0.0001, 2)	(0.0001, 0.001, 0.00001, 3)
130000	58.96	48.33	58.33	65	58.33	61.67	60	61.67
	(0.03125)	(0.00001, 0.00001, 0.00001)	(0.001, 103)	(100, 23)	(0.001, 103)	(0.00001, 23)	(0.001, 0.001, 0.001, 4)	(0.001, 0.1, 0.00001, 83)
131000	78.33	63.33	73.33	76.67	83.33	76.67	81.67	75.00
	(0.03125)	(0.00001, 0.00001, 0.00001)	(0.001, 163)	(0.00001, 83)	(0.01, 23)	(0.00001, 183)	(0.01, 0.001, 0.01, 4)	(1, 1000, 1000, 163)
140000	71.67	40	58.33	76.67	56.67	78.33	45	80.00
	(0.0625)	(0.00001, 0.00001, 0.00001)	(0.1, 183)	(0.00001, 143)	(0.1, 3)	(100, 3)	(0.01, 0.001, 0.0001, 4)	(10, 1, 1, 3)
142000	86.67	51.67	85	68.33	91.67	88.33	78.33	91.67
	(0.25)	(0.00001, 0.00001, 0.00001)	(0.001, 183)	(10000, 23)	(0.1, 3)	(1, 3)	(0.01, 0.001, 0.0001, 4)	(1, 100, 0.01, 3)
143000	66.67	51.67	63.33	65	63.33	65	71.67	58.33
	(0.03125)	(0.00001, 0.00001, 0.00001)	(0.001, 163)	(0.00001, 203)	(0.001, 163)	(0.00001, 203)	(0.01, 0.1, 0.1, 0.25)	(0.001, 10, 1, 183)
144000	73.33	48.33	68.33	71.67	68.33	70	66.67	68.33
	(16)	(0.00001, 0.00001, 0.00001)	(0.01, 63)	(0.00001, 203)	(0.01, 63)	(0.00001, 203)	(0.1, 0.1, 0.1, 0.25)	(100, 0.1, 0.00001, 203)
147000	58.33	55	63.33	73.33	63.33	73.33	70	75.00
	(0.125)	(0.00001, 0.00001, 0.00001)	(0.01, 203)	(0.00001, 123)	(0.01, 203)	(0.00001, 123)	(0.001, 0.001, 0.1, 4)	(0.1, 1, 0.00001, 203)
148000	86.67	50	90	80	88.33	86.67	83.33	83.33
	(2)	(0.00001, 0.00001, 0.00001)	(0.001, 103)	(0.00001, 83)	(0.001, 23)	(0.00001, 63)	(0.001, 0.0001, 0.1, 4)	(0.1, 100, 0.001, 83)
152000	56.67	48.33	65	66.67	66.67	45	51.67	68.33
	(0.25)	(0.00001, 0.00001, 0.00001)	(0.0001, 143)	(0.1, 23)	(0.00001, 103)	(1, 103)	(0.1, 0.1, 0.001, 0.25)	(100, 100000, 10, 43)
153000	79.33	55	76.67	73.33	76.67	68.33	80.00	
	(0.5)	(0.00001, 0.00001, 0.00001)	(0.001, 203)	(0.00001, 43)	(0.001, 203)	(0.00001, 43)	(0.01, 10, 100, 4)	(1, 1000, 0.1, 203)
161000	86.67	46.67	91.67	91.67	88.33	90	93.33	91.67
	(0.5)	(0.00001, 0.00001, 0.00001)	(0.001, 183)	(0.00001, 143)	(0.1, 23)	(0.00001, 43)	(0.001, 0.0001, 0.00001, 2)	(100, 1000, 0.001, 3)
163000	80	41.67	80	78.33	80	85	71.67	83.33
	(4)	(0.00001, 0.00001, 0.00001)	(0.0001, 203)	(1, 23)	(0.001, 83)	(0.00001, 3)	(0.001, 0.0001, 0.00001, 0.5)	(0.0001, 0.01, 0.00001, 43)
17000	83.33	56.67	86.67	90	86.67	86.67	83.33	93.33
	(2)	(0.00001, 0.00001, 0.00001)	(0.001, 163)	(0.00001, 143)	(0.001, 163)	(0.00001, 203)	(0.001, 0.00001, 0.1, 4)	(0.001, 0.1, 0.0001, 203)
171000	88.33	48.33	76.67	68.33	76.67	75	63.33	75.00
	(0.03125)	(0.00001, 0.00001, 0.00001)	(0.001, 123)	(0.00001, 83)	(0.001, 123)	(0.00001, 3)	(0.1, 0.001, 0.1, 0.25)	(0.00001, 0.01, 0.00001, 183)
173000	85	61.67	86.67	71.67	83.33	78.33	80	86.67
	(0.125)	(0.00001, 0.00001, 0.00001)	(0.001, 203)	(1000, 23)	(0.001, 63)	(1000, 3)	(0.0001, 0.01, 0.1, 0.25)	(10, 10000, 0.00001, 23)
174000	81.67	45	81.67	85	83.33	88.33	88.33	90.00
	(0.03125)	(0.00001, 0.00001, 0.00001)	(0.01, 103)	(0.001, 43)	(0.01, 103)	(0.00001, 23)	(0.0001, 0.01, 0.0001, 4)	(0.00001, 0.0001, 0.00001, 3)
182000	76.67	46.67	78.33	76.67	80	76.67	70	81.67
	(0.03125)	(0.00001, 0.00001, 0.00001)	(0.001, 183)	(0.00001, 123)	(0.001, 183)	(0.00001, 123)	(0.00001, 0.0001, 0.0001, 0.25)	(1, 100, 0.1, 3)
183000	70	48.33	80	68.33	80	78.33	68.33	80.00
	(4)	(0.00001, 0.00001, 0.00001)	(0.0001, 203)	(1000, 43)	(0.0001, 203)	(0.00001, 43)	(0.00001, 0.01, 0.00001, 4)	(1, 10000, 0.0001, 203)
187000	83.33	43.33	81.67	80	83.33	85	81.67	83.33
	(0.03125)	(0.00001, 0.00001, 0.00001)	(0.0001, 103)	(10000, 23)	(0.001, 103)	(0.00001, 3)	(0.01, 0.01, 0.01, 0.25)	(0.0001, 0.1, 0.00001, 163)
189000	76.67	58.33	80	81.67	80	75	73.33	75.00
	(0.25)	(0.00001, 0.00001, 0.00001)	(0.001, 203)	(0.00001, 123)	(0.001, 203)	(0.00001, 163)	(0.01, 0.001, 0.001, 4)	(1000, 0.1, 10, 3)
20000	65	40	63.33	73.33	70	63.33	65	73.33
	(0.03125)	(0.00001, 0.00001, 0.00001)	(0.001, 183)	(100000, 23)	(0.01, 3)	(0.00001, 163)	(0.001, 0.001, 0.0001, 4)	(0.1, 100, 0.00001, 43)
201000	86.67	58.33	90	88.33	91.67	86.67	80	90.00
	(1)	(0.00001, 0.00001, 0.00001)	(1, 23)	(0.00001, 63)	(0.001, 63)	(0.00001, 23)	(0.0001, 0.01, 0.001, 4)	(0.001, 0.01, 0.00001, 23)
21000	90	50	93.33	83.33	88.33	86.67	86.67	88.33
	(1)	(0.00001, 0.00001, 0.00001)	(0.001, 183)	(0.00001, 43)	(0.01, 163)	(0.00001, 103)	(0.1, 0.1, 0.1, 0.25)	(1000, 100000, 0.00001, 203)
22000	70	46.67	70	53.33	73.33	68.33	68.33	73.33
	(0.03125)	(0.00001, 0.00001, 0.00001)	(0.001, 143)	(1, 23)	(0.1, 23)	(0.00001, 43)	(0.001, 0.001, 0.001, 0.25)	(10, 1000, 0.001, 203)
231000	65	55	61.67	60	56.67	60	53.33	53.33
	(0.0625)	(0.00001, 0.00001, 0.00001)	(0.1, 123)	(0.00001, 203)	(0.01, 83)	(0.00001, 203)	(0.1, 0.001, 0.0001, 0.25)	(100, 1000, 0.0001, 123)
276000	78.67	56.67	78.33	76.67	80	76.67	71.67	78.33
	(0.25)	(0.00001, 0.00001, 0.00001)	(0.01, 63)	(0.00001, 183)	(0.01, 63)	(0.00001, 183)	(0.001, 0.0001, 0.0001, 4)	(0.01, 10, 0.00001, 23)
296000	78.33	46.67	85	71.67	85	78.33	68.33	76.67
	(16)	(0.00001, 0.00001, 0.00001)	(0.001, 183)	(0.00001, 103)	(0.001, 183)	(0.00001, 3)	(0.00001, 0.001, 0.01, 4)	(0.01, 10, 0.001, 103)
33000	83.33	43.33	86.67	81.67	88.33	81.67	68.33	88.33
	(0.03125)	(0.00001, 0.00001, 0.00001)	(0.001, 143)	(0.00001, 43)	(0.01, 3)	(0.00001, 43)	(0.01, 0.1, 0.1, 0.25)	(1, 10000, 0.1, 63)
335000	78.33	43.33	75	78.33	73.33	75	70	76.67
	(8)	(0.00001, 0.00001, 0.00001)	(0.001, 203)	(0.00001, 43)	(0.01, 3)	(0.00001, 43)	(0.1, 0.1, 0.1, 4)	(0.00001, 1, 0.00001, 103)
34000	80.33	53.33	76.67	86.67	83.33	86.67	73.33	85.00
	(0.25)	(0.00001, 0.00001, 0.00001)	(0.001, 83)	(0.00001, 163)	(0.001, 203)	(0.00001, 163)	(0.1, 0.001, 0.001, 0.5)	(100000, 10000, 1, 3)
384000	85	55	78.33	86.67	78.33	85	78.33	81.67
	(8)	(0.00001, 0.00001, 0.00001)	(0.001, 83)	(1000, 63)	(0.001, 83)	(0.00001, 183)	(0.01, 0.01, 0.001, 2)	(0.00001, 0.01, 0.00001, 143)
41000	65	43.33	66.67	68.33	56.67	65	56.67	66.67
	(0.125)	(0.00001, 0.00001, 0.00001)	(0.1, 43)	(0.00001, 183)	(0.1, 43)	(1000, 23)	(0.001, 0.0001, 0.01, 4)	(0.00001, 0.01, 0.00001, 143)
46000	70	46.67	78.33	81.67	80	83.33	65	71.67
	(0.125)	(0.00001, 0.00001, 0.00001)	(0.001, 163)	(0.00001, 43)	(0.001, 163)	(0.00001, 43)	(0.0001, 0.001, 0.01, 4)	(1000, 100000, 0.00001, 103)
Average ACC	74.87	49.93	74.98	74.83	76.33	75.43	69.87	**77.33**
Average Rank	4.11	7.91	3.99	3.97	3.42	4.05	5.61	**2.94**

† represents the proposed model.

Table 4.5: Performance comparison of the proposed GRVFL-2V along with the baseline models based on classification accuracy for AwA datasets.

Dataset	SVM2K [82] (C_1)	MvTSVM [83] (C_1, C_2, D)	ELM1 [11] (C, N)	ELM2 [11] (C, N)	RVFL1 [6] (C, N)	RVFL2 [6] (C, N)	MVLDM [92] $(C_1, \nu_1, \nu_2, \sigma)$	GRVFL-2V† (C_1, θ, ρ, N)
Chimpanzee vs Giant panda	84.03	47.22	71.53	72.92	71.53	80.89	72.22	90.97
	(0.00001)	(0.00001, 0.00001, 0.00001)	(0.0001, 123)	(0.00001, 163)	(0.001, 3)	(0.001, 3)	(1000, 0.0001, 0.01, 4)	(0.001, 10, 0.00001, 3)
Chimpanzee vs Leopard	80.11	46.53	63.89	83.33	72.92	80.42	68.75	89.58
	(0.00001)	(0.00001, 0.00001, 0.00001)	(1000, 43)	(0.00001, 203)	(0.01, 23)	(0.0001, 3)	(0.001, 0.00001, 0.00001, 4)	(0.001, 100000, 0.00001, 3)
Chimpanzee vs Persian cat	70.86	50	79.86	69.44	79.17	80.56	86.11	83.33
	(0.0001)	(0.00001, 0.00001, 0.00001)	(0.0001, 183)	(0.00001, 183)	(0.0001, 183)	(0.1, 3)	(100, 0.0001, 0.00001, 4)	(10, 10000, 0.00001, 3)
Chimpanzee vs Pig	50.42	51.39	68.75	81.25	69.44	79.17	66.67	83.33
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.00001, 163)	(0.00001, 163)	(0.00001, 163)	(10, 3)	(100000, 0.001, 0.001, 4)	(0.001, 10, 0.0001, 3)
Chimpanzee vs Hippopotamus	70.94	54.86	71.53	70.14	72.92	78.47	78.47	78.47
	(0.00001)	(0.00001, 0.00001, 0.00001)	(0.00001, 143)	(0.00001, 183)	(0.00001, 143)	(0.001, 3)	(1000, 0.0001, 1, 4)	(0.0001, 10, 0.00001, 3)
Chimpanzee vs Humpback whale	92.36	81.39	86.81	92.36	88.89	91.14	81.25	96.53
	(0.1)	(0.00001, 0.00001, 0.00001)	(0.0001, 83)	(0.00001, 183)	(0.0001, 83)	(0.01, 3)	(10000, 0.001, 0.0001, 4)	(0.001, 100000, 0.001, 23)
Chimpanzee vs Raccoon	80.33	63.47	69.44	63.89	73.61	79.86	72.22	83.33
	(0.001)	(0.00001, 0.00001, 100000)	(0.0001, 203)	(0.00001, 123)	(0.001, 3)	(1000, 23)	(10000, 0.00001, 0.01, 4)	(0.001, 1, 0.0001, 3)
Chimpanzee vs Rat	77.08	52.78	57.64	75	63.89	71.94	68.06	82.64
	(10)	(0.00001, 0.00001, 0.00001)	(100000, 43)	(0.00001, 183)	(0.001, 163)	(0.001, 3)	(100000, 0.0001, 0.00001, 4)	(0.1, 1000, 0.0001, 43)
Chimpanzee vs Seal	70.69	53.47	79.17	76.39	79.17	79.81	75.69	85.42
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.0001, 183)	(0.00001, 143)	(0.001, 3)	(0.0001, 3)	(0.001, 0.001, 0.01, 0.25)	(0.001, 100000, 0.01, 3)
Giant panda vs Leopard	80.19	54.17	61.11	78.47	72.92	80.11	61.81	90.97
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.0001, 103)	(0.00001, 203)	(0.001, 3)	(0.001, 3)	(100000, 0.00001, 0.01, 0.25)	(0.001, 10, 0.001, 3)
Giant panda vs Persian cat	81.81	52.08	77.78	76.39	64.58	80.42	66.67	84.03
	(0.0001)	(0.00001, 0.00001, 0.00001)	(100, 103)	(0.00001, 203)	(0.001, 23)	(100000, 3)	(0.01, 0.00001, 0.01, 4)	(0.1, 1000, 0.01, 23)
Giant panda vs Pig	80.56	51.39	63.89	75	65.97	79.81	65.97	88.19
	(0.0001)	(0.00001, 0.00001, 0.00001)	(1, 43)	(0.00001, 203)	(0.001, 3)	(0.001, 3)	(1000, 0.00001, 0.01, 0.25)	(0.001, 10, 0.00001, 3)
Giant panda vs Hippopotamus	77.78	54.17	74.31	81.25	68.06	71.94	74.31	84.72
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.00001, 183)	(0.00001, 183)	(0.01, 3)	(0.01, 23)	(100000, 0.001, 100, 2)	(0.01, 100, 0.001, 23)
Giant panda vs Humpback whale	93.06	46.53	93.06	91.67	93.06	93.22	93.75	81.25
	(0.00001)	(0.00001, 0.00001, 0.00001)	(0.0001, 203)	(0.00001, 203)	(0.0001, 203)	(0.001, 3)	(0.001, 0.00001, 0.01, 4)	(0.001, 1, 0.00001, 3)
Giant panda vs Raccoon	80.19	52.78	68.06	74.31	68.75	80.19	64.58	89.58
	(10000)	(0.00001, 0.00001, 0.00001)	(0.0001, 203)	(0.00001, 163)	(0.00001, 203)	(0.001, 3)	(100000, 0.00001, 0.00001, 2)	(0.001, 10, 0.0001, 3)
Giant panda vs Rat	83.33	69.31	66.67	76.39	68.06	80.5	70.14	84.72
	(1)	(0.00001, 0.00001, 100000)	(10, 103)	(0.00001, 123)	(0.001, 43)	(0.001, 3)	(10000, 0.0001, 0.01, 0.25)	(0.001, 0.1, 0.00001, 43)
Giant panda vs Seal	85.89	56.94	80.56	77.08	80.56	80.19	86.81	89.58
	(0.1)	(0.00001, 0.00001, 0.00001)	(0.0001, 203)	(0.00001, 183)	(0.001, 3)	(0.0001, 3)	(100000, 0.001, 0.01, 2)	(1, 1000, 0.0001, 3)
Leopard vs Persian cat	82.19	79.31	70.83	84.72	77.78	88.19	80.56	90.97
	(0.00001)	(0.00001, 0.00001, 0.00001)	(100, 63)	(0.00001, 203)	(0.001, 23)	(0.001, 3)	(0.00001, 0.00001, 100, 4)	(0.001, 1, 0.001, 3)
Leopard vs Pig	75	61.39	61.11	75	66.67	72.17	68.75	78.47
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.0001, 183)	(0.00001, 183)	(0.001, 3)	(0.001, 3)	(0.01, 0.001, 100, 4)	(0.001, 1, 0.0001, 3)
Leopard vs Hippopotamus	78.17	50.69	73.61	77.08	74.31	75.94	75	80.56
	(10)	(0.00001, 0.00001, 0.00001)	(0.0001, 143)	(0.00001, 143)	(0.0001, 43)	(0.0001, 3)	(10000, 0.0001, 0.001, 4)	(0.01, 10, 0.001, 23)
Leopard vs Humpback whale	90.75	79.31	89.58	91.67	90.97	90.83	89.58	95.83
	(0.00001)	(0.00001, 0.00001, 0.00001)	(0.00001, 103)	(0.00001, 183)	(0.0001, 143)	(0.001, 23)	(100, 0.00001, 0.01, 4)	(0.001, 10, 0.0001, 3)
Leopard vs Raccoon	80.56	55	59.03	57.64	59.03	69.25	56.94	79.17
	(0.0001)	(0.00001, 0.00001, 0.00001)	(0.0001, 183)	(0.0001, 183)	(0.001, 3)	(0.001, 3)	(0.01, 0.00001, 0.00001, 0.25)	(100, 100000, 0.00001, 23)
Leopard vs Rat	76.42	68.61	72.22	76.39	68.06	79.86	65.28	83.33
	(0.0001)	(0.00001, 0.00001, 0.00001)	(10000, 43)	(0.00001, 183)	(0.001, 3)	(0.001, 3)	(10000, 0.0001, 0.00001, 0.25)	(0.001, 1, 0.0001, 3)
Leopard vs Seal	80.42	63.47	75.69	79.86	75	83.33	81.25	84.72
	(10000)	(0.00001, 0.00001, 0.00001)	(0.0001, 123)	(0.00001, 143)	(0.0001, 123)	(0.001, 43)	(10000, 0.0001, 0.0001, 4)	(10, 100000, 0.01, 3)
Persian cat vs Pig	70	69.31	63.89	67.36	70.14	74.31	69.44	73.61
	(0.001)	(0.00001, 0.00001, 0.00001)	(0.0001, 183)	(0.00001, 163)	(0.001, 3)	(10000, 3)	(100, 0.00001, 0.01, 4)	(0.01, 1, 0.00001, 3)
Persian cat vs Hippopotamus	76.81	76.53	75.69	79.86	77.08	78.94	75.69	80.56
	(0.01)	(0.00001, 0.00001, 0.00001)	(1, 63)	(0.00001, 203)	(0.01, 3)	(1, 3)	(100000, 0.001, 0.00001, 0.25)	(0.01, 10, 0.001, 23)
Persian cat vs Humpback whale	71.67	71.39	81.25	88.19	81.94	81.75	85.42	95.14
	(0.00001)	(0.00001, 0.00001, 0.00001)	(0.00001, 143)	(0.00001, 203)	(0.00001, 143)	(1, 23)	(0.01, 0.00001, 0.1, 4)	(0.001, 1, 0.001, 3)
Persian cat vs Raccoon	82.64	79.31	73.61	81.25	69.44	71.64	65.97	84.72
	(0.00001)	(0.00001, 0.00001, 0.00001)	(100, 23)	(0.00001, 163)	(0.001, 23)	(0.001, 3)	(100000, 0.00001, 0.001, 2)	(0.001, 0.1, 0.00001, 43)
Persian cat vs Rat	60.44	64.17	54.86	56.25	59.72	60.67	56.94	65.97
	(0.001)	(0.00001, 0.00001, 0.00001)	(0.001, 183)	(100000, 103)	(0.001, 3)	(0.001, 3)	(1000, 0.00001, 10, 0.5)	(0.0001, 1, 0.0001, 3)
Persian cat vs Seal	80.42	73.47	72.22	71.53	65.97	72.94	83.33	84.72
	(1)	(0.00001, 0.00001, 0.00001)	(0.0001, 183)	(0.00001, 83)	(0.01, 63)	(0.001, 3)	(10000, 0.00001, 0.01, 4)	(0.0001, 10, 0.00001, 3)
Pig vs Hippopotamus	71.53	65.83	70.14	65.97	64.58	67.36	72.22	83.33
	(0.0001)	(0.00001, 0.00001, 0.00001)	(0.0001, 83)	(0.00001, 203)	(0.0001, 123)	(0.01, 63)	(1000, 0.00001, 0.01, 0.25)	(10, 10000, 0.00001, 3)
Pig vs Humpback whale	80.19	77.92	82.64	89.58	82.64	80.19	88.89	82.64
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.00001, 203)	(0.00001, 203)	(0.00001, 203)	(0.001, 3)	(100000, 0.001, 0.0001, 0.25)	(0.1, 1000, 0.0001, 43)
Pig vs Raccoon	71.69	69.31	61.11	72.92	64.58	72.22	62.5	81.25
	(0.00001)	(0.00001, 0.00001, 0.00001)	(0.0001, 43)	(0.00001, 183)	(0.0001, 43)	(10000, 3)	(0.0001, 0.00001, 0.0001, 4)	(0.001, 1, 0.00001, 3)
Pig vs Rat	71.53	68.61	62.5	59.72	57.64	68.06	64.58	63.89
	(0.01)	(0.00001, 0.00001, 0.00001)	(1000, 63)	(0.0001, 203)	(0.001, 43)	(0.001, 3)	(1000, 0.00001, 0.01, 0.25)	(100, 10000, 0.001, 43)
Pig vs Seal	72.69	65.56	70.14	68.06	72.92	74.31	72.92	78.47
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.00001, 163)	(0.00001, 183)	(0.001, 3)	(0.0001, 3)	(0.001, 0.001, 0.00001, 4)	(0.001, 1, 0.00001, 3)
Hippopotamus vs Humpback whale	82.03	80.31	77.78	79.86	81.25	82.81	79.86	88.19
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.0001, 143)	(0.00001, 143)	(0.01, 3)	(0.001, 23)	(0.001, 0.0001, 10000, 0.25)	(0.001, 1, 0.00001, 3)
Hippopotamus vs Raccoon	78.47	75.14	72.22	70.14	78.47	80.94	75.69	77.78
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.0001, 203)	(0.00001, 183)	(0.001, 3)	(0.001, 23)	(0.00001, 0.001, 1, 4)	(0.01, 1000, 0.001, 43)
Hippopotamus vs Rat	75.33	75.83	65.28	71.53	70.14	70.25	64.58	81.25
	(100)	(0.00001, 0.00001, 0.00001)	(1000, 43)	(0.00001, 183)	(0.01, 23)	(0.001, 3)	(100000, 0.0001, 0.1, 4)	(0.001, 0.1, 0.001, 3)
Hippopotamus vs Seal	69.44	49.31	58.33	65.28	61.81	70.83	60.42	61.81
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.1, 63)	(0.00001, 183)	(0.001, 43)	(0.001, 3)	(0.001, 0.01, 0.00001, 4)	(0.001, 1, 0.00001.63)
Humpback whale vs Raccoon	85.67	80.69	84.03	90.97	81.94	82.36	83.33	93.06
	(0.00001)	(0.00001, 0.00001, 10000)	(0.00001, 83)	(0.00001, 123)	(0.0001, 83)	(0.001, 3)	(10000, 0.0001, 0.01, 0.25)	(10, 100000, 0.00001, 3)
Humpback whale vs Rat	82.28	80.31	81.94	86.11	81.94	80.58	77.78	90.97
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.0001, 183)	(0.00001, 203)	(0.0001, 183)	(100, 3)	(100000, 0.001, 10, 0.25)	(0.001, 0.01, 0.0001, 3)
Humpback whale vs Seal	76.39	72.08	77.78	73.61	78.47	79.17	78.47	80.56
	(0.01)	(0.00001, 0.00001, 0.00001)	(0.0001, 143)	(0.00001, 203)	(0.0001, 143)	(0.0001, 3)	(100000, 0.0001, 0.001, 0.25)	(0.01, 100, 0.0001, 3)
Raccoon vs Rat	62.22	61.89	59.72	70.83	61.81	60.22	65.28	70.83
	(100)	(0.00001, 0.00001, 0.00001)	(100, 43)	(0.00001, 163)	(0.001, 163)	(0.001, 3)	(100000, 0.00001, 10, 4)	(0.001, 0.1, 0.001, 3)
Raccoon vs Seal	90.28	75.39	78.47	84.03	82.64	80.28	75.69	88.19
	(100)	(0.00001, 0.00001, 100000)	(0.00001, 183)	(0.00001, 183)	(0.001, 63)	(0.001, 3)	(100000, 0.00001, 0.01, 4)	(100, 100000, 10, 43)
Rat vs Seal	70.86	65.17	68.75	68.75	68.75	67.83	69.87	81.25
	(0.001)	(0.00001, 0.00001, 0.00001)	(0.00001, 183)	(0.00001, 143)	(0.00001, 183)	(0.01, 3)	(100000, 0.001, 0.01, 0.25)	(0.0001, 0.1, 0.00001, 3)
Average ACC	77.46	64.31	71.74	75.99	72.87	77.46	73.33	**83.29**
Average Rank	3.46	6.84	5.99	4.32	5.22	3.59	5.02	**1.56**

† represents the proposed model.

4.3.4 Experiments on Animal with Attributes (AwA) Datasets

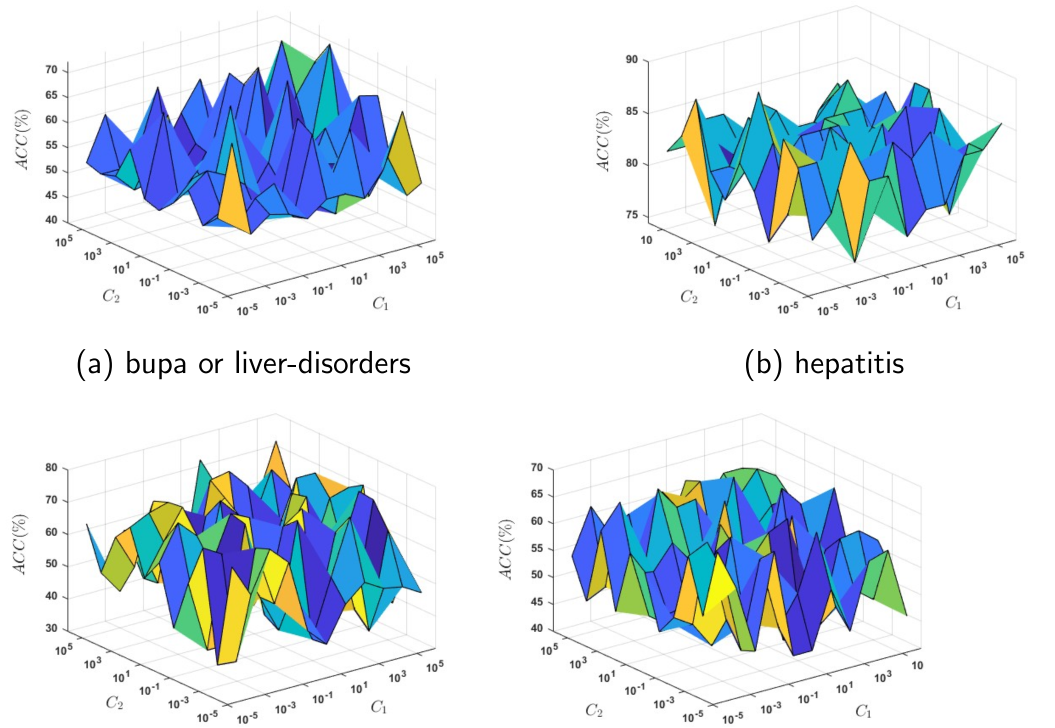
AwA[4] is a large dataset with 30,475 images covering 50 different animal types. Six representations of pre-extracted features are used to describe each image. For our analysis, we focus on a subset of ten specific test classes drawn from this dataset. These classes include animals like chimpanzees, giant pandas, leopards, Persian cats, pigs, hippopotamuses, humpback whales, raccoons, rats, and seals, totaling 6180 images. For our analysis, we employ two distinct feature representations: a 2000-dimensional L_1 normalized speeded-up robust features (SURF) descriptors ($view - B$) and a 252-dimensional histogram of oriented gradient features ($view - A$). We build and train 45 binary classifiers using the one-against-one method for every possible combination of class pairs in the dataset. The average rank and ACC of our proposed model are displayed in Table 4.5 in relation to the baseline models. Across the 45 datasets from AwA, our model achieved the highest average accuracy and the lowest average rank. The average ACC of proposed GRVFL-2V is 83.29, which is upto 4% to 20% higher than the baseline models. Also, GRVFL-2V has the lowest average rank of 1.56. The results clearly demonstrate the superior performance of the proposed GRVFL-2V model over the baseline models.

4.3.5 Sensitivity Analysis of Hyperparameters c_1 and c_2

To gain a comprehensive understanding of how hyperparameters affect the generalization ability of the proposed GRVFL-2V model, we conducted a systematic exploration of the hyperparameter space by tuning the values of c_1 and c_2. This helps us to identify the optimal configuration that maximizes predictive accuracy and improves the model's resilience to previously unseen data. Figure 4.1 visually represents how the accuracy of the model behaves when the hyperparameters are tuned. The visual clearly shows that the proposed model is highly sensitive to the values of hyperparameters c_1 and c_2. In Figure 4.1 (a), we see that optimal accuracy is achieved when $c_1 = 10^5$ and $c_2 = 10^4$ whereas, in Figure 4.1 (c), optimal configuration is found at two distinct coordinates, $(10^5, 10^5)$ and $(10, 10^5)$. Similar observations can be made for Figure 4.1 (b) and Figure 4.1 (d). Overall, these findings underscore the need for careful selection of hyperparameter values to achieve optimal model

[3] https://wang.ist.psu.edu/docs/related/

[4] http://attributes.kyb.tuebingen.mpg.de



(a) bupa or liver-disorders

(b) hepatitis

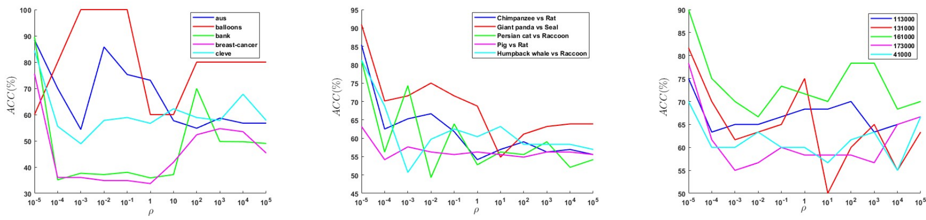
(c) ilpd_indian_liver

(d) monks_2

Figure 4.1: The effect of hyperparameter (c_1, c_2) tuning on the accuracy (ACC) of some UCI and KEEL datasets on the performance of proposed GRVFL-2V model.

performance.

4.3.6 Sensitivity Analysis of Coupling Parameter ρ



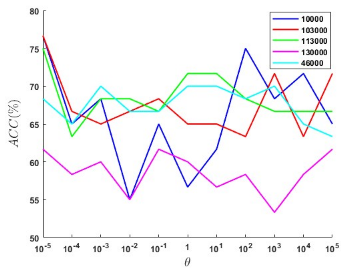
(a) UCI and KEEL (b) AwA (c) Corel5k

Figure 4.2: The effect of coupling parameter ρ tuning on the accuracy (ACC) of UCI and KEEl, AwA, and Corel5k datasets on the performance of proposed GRVFL-2V model.

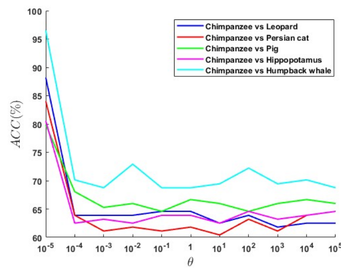
The primal optimization problem (4.1) combines two distinct classification objectives, each corresponding to a different view. These objectives are linked through a coupling term $\xi_1^t \xi_2$, where ρ serves as the regularization constant known as the coupling parameter. We analyzed the impact of ρ on our model's performance by fixing other parameters at their optimal values and tuning ρ within the specified range outlined in the experimental setup. The performance of the proposed model across different datasets is depicted in Figure 4.2. It can be observed from Figure 4.2 (a) that the proposed GRVFL-2V model achieves the highest accuracy when ρ is set to 10^{-5}. This trend is also evident in Figure 4.2 (b) and Figure 4.2 (c). These results indicate that the optimal effect of coupling terms and improved generalization performance are achieved when ρ is tuned to 10^{-5}.

4.3.7 Sensitivity Analysis of Graph Regularization Parameter θ

The primal optimization problem (4.1) involves two graph regularization parameters θ_1 and θ_2 for each view, with the goal of preserving the geometrical aspects of multiview data through the graph embedding GE) framework in the model. In our experiment, we set $\theta_1 = \theta_2 = \theta$ to study the effect of geometrical properties of the multiview data through the LFDA technique under the GE framework. The impact of tuning θ on the efficacy of the GRVFL-2V model is illustrated in Figure 4.3. In Figure 4.3 (a), when utilizing datasets from Corel5k, it is evident that the model's performance is most favorable initially at the minimum ρ value, specifically at $\rho = 10^{-5}$, and subsequently, performance increases for $\rho > 1$. Conversely, Figure 4.3 (b), the optimal performance is achieved at $\rho = 10^{-5}$ but then experiences a significant decline. These results emphasize the critical importance of selecting the appropriate ρ value for maximizing the proposed model's performance.



(a) Corel5k



(b) AwA

Figure 4.3: The effect of graph embedding parameter θ tuning on the accuracy (ACC) of Corel5k and AwA datasets on the performance of proposed GRVFL-2V model.

Chapter 5

Conclusions and Future Directions

The study in this thesis has concentrated on examining two state-of-the-art shallow machine learning classification models, namely the least square twin support vector machine (LSTSVM) and the random vector functional link (RVFL) neural network. These shallow learning models have been widely applied in different scenarios and have demonstrated commendable performance. Nevertheless, it is important to acknowledge that both models possess certain limitations and intricacies that have been addressed and mitigated in the research presented in this thesis. In this thesis, efforts have been made to enhance the performance of these algorithms by addressing inherent issues in the models and proposing solutions to improve their overall effectiveness and applicability in real-world scenarios.

5.1 Conclusions

[1] Literature review: An extensive review of the literature was provided on hyperplane-based classifiers, including the least square twin support vector machine and the granular ball support vector machine. Additionally, an overview was given on granular computing and multiview learning, shedding light on their significance in the field. Furthermore, the theory of random vector functional link neural network and graph embedding was also presented, offering insights into their applications and implications in various domains. This comprehensive presentation aimed to provide a thorough understanding of these advanced concepts and methodologies for

the readers.

[2] Development of least square twin support vector machine: We have proposed two novel models, the granular ball least square twin support vector machine (GBLSTSVM) and the large-scale granular ball least square twin support vector machine (LS-GBLSTSVM), by incorporating the concept of granular computing in LSTSVM. This incorporation enables our proposed models to achieve the following: (i) Robustness: Both GBLSTSVM and LS-GBLSTSVM demonstrate robustness due to the coarse nature of granular balls, making them less susceptible to noise and outliers. (ii) Efficiency: The efficiency of GBLSTSVM and LS-GBLSTSVM stems from the significantly lower number of coarse granular balls compared to finer data points, enhancing computational efficiency. (iii) Scalability: Our proposed models are well suited for large-scale problems, primarily due to the significantly reduced number of generated granular balls compared to the total training data points. The proposed LS-GBLSTSVM model demonstrates exceptional scalability since it does not necessitate matrix inversion to determine optimal parameters. This is evidenced by experiments conducted on the NDC dataset, showcasing their ability to handle large-scale datasets effectively. An extensive series of experiments and statistical analyses have supported the above claims, including ranking schemes, the Friedman test, the Wilcoxon signed rank test, and the win-tie-loss test. The key findings from our experiments include: (i) Both linear and nonlinear versions of GBLSTSVM and LS-GBLSTSVM demonstrate superior efficiency and generalization performance compared to baseline models, with an average accuracy improvement of up to 15%. (ii) When exposed to labeled noise in the UCI and KEEL datasets, our models exhibit exceptional robustness, achieving up to a 10% increase in average accuracy compared to baseline models under noisy conditions. (iii) Evaluating our proposed models on NDC datasets ranging from 10k to 5m samples underscores their scalability, surpassing various baseline models in training speed by up to 1000 times, particularly beyond the NDC-50k threshold, where memory limitations often hinder baseline models. These findings collectively highlight the effectiveness, robustness, and scalability of our proposed GBLSTSVM and LS-GBLSTSVM models, particularly in handling large-scale and noisy datasets.

[3] Development of random vector functional link neural network: We pre-

sented a generic framework that integrates RVFL architecture with MVL, enhancing generalization by incorporating intrinsic and penalty graphical representations of multiview data via the GE framework, resulting in the development of the GRVFL-2V model. The proposed model aims to enhance the classification performance by integrating information from multiple views. To evaluate the performance of the proposed model, it is compared to various baseline models using UCI and KEEL datasets, Corel5K datasets, and AwA datasets. The experimental results reveal several key findings. Firstly, the proposed GRVFL-2V model demonstrates exceptional performance for UCI and KEEL datasets, achieving an average accuracy improvement ranging from 4% to 20% compared to the baseline models. Moreover, it also achieves the lowest rank among all the models considered. Secondly, in the case of the Corel5K dataset, our proposed GRVFL-2V model outperforms the baseline models by achieving the highest average accuracy and the lowest average rank. This indicates its superior performance in handling the Corel5K dataset. Lastly, for the AwA dataset, the proposed model exhibits outstanding performance by achieving an accuracy improvement of up to 20% compared to the baseline models. Additionally, it also attains the least average rank among all the models considered. Furthermore, a comprehensive statistical analysis is conducted to validate the effectiveness and superiority of the proposed model. The analysis includes a ranking scheme, Friedman test, Nemenyi post hoc test, and win-tie-loss test. The results of these tests unequivocally support the superiority of the proposed GRVFL-2V model over the existing baseline models.

5.2 Future Directions

In this section, we explore potential future directions arising from the findings of this thesis.

[1] In this thesis, we have focused on binary classification problems. A crucial area for future research would involve adapting the proposed models to be suitable for multi-class and regression problems.

[2] In this thesis, we presented an RVFL model based on two views. An essential research direction involves extending the proposed model to accommodate datasets with more than two views while simultaneously reducing

computational complexity.

[3] The classification models introduced in this thesis have the potential to be expanded to include advanced kernel-based models such as restricted kernel machines (RKM), Boltzmann machines, and others.

[4] The proposed models do not address scenarios with imbalanced datasets, which may result in reduced generalization performance. Thus, developing novel models based on multi-view and granular computing to handle imbalance will have a significant impact, particularly in the healthcare domain where imbalanced datasets are common.

[5] Shallow learning has limited capacity to identify intricate patterns and features in datasets. Thus, extending the developed models to incorporate deep learning techniques while preserving the overall principles and architecture can be a potential future direction to enhance the models' ability to capture complex data structures and improve performance.

Bibliography

[1] Rajdeep Kaur, Rakesh Kumar, and Meenu Gupta. Deep neural network for food image classification and nutrient identification: A systematic review. *Reviews in Endocrine and Metabolic Disorders*, 24(4):633–653, 2023.

[2] Phani Teja Nallamothu and Mohd Shais Khan. Machine learning for spam detection. *Asian Journal of Advances in Research*, 6(1):167–179, 2023.

[3] Ahmad Yousaf Gill, Ayesha Saeed, Saad Rasool, Ali Husnain, and Hafiz Khawar Hussain. Revolutionizing healthcare: How machine learning is transforming patient diagnoses-a comprehensive review of ai's impact on medical diagnosis. *Journal of World Science*, 2(10):1638–1652, 2023.

[4] Alaa Alslaity and Rita Orji. Machine learning techniques for emotion detection and sentiment analysis: current state, challenges, and future directions. *Behaviour & Information Technology*, 43(1):139–164, 2024.

[5] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[6] Yoh-Han Pao, Gwang-Hoon Park, and Dejan J Sobajic. Learning and generalization characteristics of the random vector functional-link net. *Neurocomputing*, 6(2):163–180, 1994.

[7] Ashwani Kumar Malik, Ruobin Gao, MA Ganaie, M. Tanveer, and PN Suganthan. Random vector functional link network: recent developments, applications, and future directions. *Applied Soft Computing*, page 110377, 2023.

[8] M Sajid, Rahul Sharma, Iman Beheshti, and M Tanveer. Decoding cognitive health using machine learning: A comprehensive evaluation for diagnosis of significant memory concern. *arXiv preprint arXiv:2405.07070*, 2024.

[9] Jayadeva, R. Khemchandani, and Suresh Chandra. Twin support vector machines for pattern classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(5):905–910, 2007.

[10] M Arun Kumar and Madan Gopal. Least squares twin support vector machines for pattern classification. *Expert Systems with Applications*, 36(4):7535–7543, 2009.

[11] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1-3):489–501, 2006.

[12] CL Philip Chen and Zhulin Liu. Broad learning system: An effective and efficient incremental learning system without the need for deep architecture. *IEEE transactions on neural networks and learning systems*, 29(1):10–24, 2017.

[13] Laure Zago, Pierre-Yves Hervé, Robin Genuer, Alexandre Laurent, Bernard Mazoyer, Nathalie Tzourio-Mazoyer, and Marc Joliot. Predicting hemispheric dominance for language production in healthy individuals using support vector machine. *Human Brain Mapping*, 38(12):5871–5889, 2017.

[14] Xuedan Miao, Ying Liu, Haiquan Zhao, and Chunguang Li. Distributed online one-class support vector machine for anomaly detection over networks. *IEEE Transactions on Cybernetics*, 49(4):1475–1488, 2018.

[15] Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. A web search engine-based approach to measure semantic similarity between words. *IEEE Transactions on Knowledge and Data Engineering*, 23(7):977–990, 2010.

[16] Bharat Richhariya and M. Tanveer. EEG signal classification using universum support vector machine. *Expert Systems with Applications*, 106:169–182, 2018.

[17] Bharat Richhariya, M. Tanveer, Ashraf Haroon Rashid, and Alzheimer's Disease Neuroimaging Initiative. Diagnosis of alzheimer's disease using universum support vector machine based recursive feature elimination. *Biomedical Signal Processing and Control*, 59:101903, 2020.

[18] Chun-Fu Lin and Sheng-De Wang. Fuzzy support vector machines. *IEEE Transactions on Neural Networks*, 13(2):464–471, 2002.

[19] Xiaolin Huang, Lei Shi, and Johan AK Suykens. Support vector machine classifier with pinball loss. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(5):984–997, 2013.

[20] M. Tanveer, A Sharma, and PN Suganthan. General twin support vector machine with pinball loss function. *Information Sciences*, 494:311–327, 2019.

[21] Olvi L Mangasarian and Edward W Wild. Multisurface proximal support vector machine classification via generalized eigenvalues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1):69–74, 2005.

[22] Chao Yuan and Liming Yang. Capped l_2, p-norm metric based robust least squares twin support vector machine for pattern classification. *Neural Networks*, 142:457–478, 2021.

[23] Xijiong Xie, Feixiang Sun, Jiangbo Qian, Lijun Guo, Rong Zhang, Xulun Ye, and Zhijin Wang. Laplacian lp norm least squares twin support vector machine. *Pattern Recognition*, 136:109192, 2023.

[24] Hossein Moosaei and Milan Hladk. Sparse solution of least-squares twin multi-class support vector machine using 0 and p-norm for classification and feature selection. *Neural Networks*, 166:471–486, 2023.

[25] Hossein Moosaei, M A. Ganaie, Milan Hladk, and M. Tanveer. Inverse free reduced universum twin support vector machine for imbalanced data classification. *Neural Networks*, 157:125–135, 2023.

[26] Qi Si, Zhixia Yang, and Junyou Ye. Symmetric linex loss twin support vector machine for robust classification and its fast iterative algorithm. *Neural Networks*, 168:143–160, 2023.

[27] M. Tanveer, M. A. Ganaie, A. Bhattacharjee, and C. T. Lin. Intuitionistic fuzzy weighted least squares twin SVMs. *IEEE Transactions on Cybernetics*, 53(7):4400–4409, 2023.

[28] Mushir Akhtar, M. Tanveer, and Mohd. Arshad. Roboss: A robust, bounded, sparse, and smooth loss function for supervised learning, 2023, https://doi.org/10.48550/arXiv.2309.02250.

[29] Mushir Akhtar, M. Tanveer, and Mohd. Arshad. Advancing supervised learning with the wave loss function: A robust and smooth approach, 2024, https://doi.org/10.48550/arXiv.2404.18101.

[30] M. Tanveer, T Rajani, Reshma Rastogi, Yuan-Hai Shao, and M A. Ganaie. Comprehensive review on twin support vector machines. *Annals of Operations Research*, pages 1–46, 2022.

[31] M. Tanveer, C. Gautam, and P.N. Suganthan. Comprehensive evaluation of twin svm based classifiers on uci datasets. *Applied Soft Computing*, 83:105617, 2019.

[32] Le Zhang and PN Suganthan. A survey of randomized algorithms for training neural networks. *Information Sciences*, 364:146–155, 2016.

[33] Zhaoquan Shafiq, M.and Gu. Deep residual learning for image recognition: A survey. *Applied Sciences*, 12(18):8972, 2022.

[34] M. Sajid, A. K. Malik, and M. Tanveer. Intuitionistic fuzzy broad learning system: Enhancing robustness against noise and outliers. *IEEE Transactions on Fuzzy Systems*, pages 1–10, 2024, 10.1109/TFUZZ.2024.3400898.

[35] Burak Gülmez. Stock price prediction with optimized deep lstm network with artificial rabbits optimization algorithm. *Expert Systems with Applications*, 227:120346, 2023.

[36] M. Tanveer, M.A. Ganaie, Iman Beheshti, Tripti Goel, Nehal Ahmad, Kuan-Ting Lai, Kaizhu Huang, Yu-Dong Zhang, Javier Del Ser, and Chin-Teng Lin. Deep learning for brain age estimation: A systematic review. *Information Fusion*, 96:130–143, 2023.

[37] Weipeng Cao, Xizhao Wang, Zhong Ming, and Jinzhu Gao. A review on neural networks with random weights. *Neurocomputing*, 275:278–287, 2018.

[38] Robert A Jacobs. Increased rates of convergence through learning rate adaptation. *Neural Networks*, 1(4):295–307, 1988.

[39] Marco Gori and Alberto Tesi. On the problem of local minima in backpropagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(1):76–86, 1992.

[40] W.F. Schmidt, M.A. Kraaijveld, and R.P.W. Duin. Feedforward neural networks with random weights. In *Proceedings., 11th IAPR International Conference on Pattern Recognition. Vol.II. Conference B: Pattern Recognition Methodology and Systems*, pages 1–4, 1992.

[41] PN Suganthan and Rakesh Katuwal. On the origins of randomization-based feedforward neural networks. *Applied Soft Computing*, 105:107239, 2021.

[42] A.K. Malik, Ruobin Gao, M.A. Ganaie, M. Tanveer, and PN Suganthan. Random vector functional link network: Recent developments, applications, and future directions. *Applied Soft Computing*, 143:110377, 2023.

[43] Michael J Kearns and Umesh Vazirani. *An introduction to computational learning theory*. MIT press, 1994.

[44] Boris Igelnik and Yoh-Han Pao. Stochastic choice of basis functions in adaptive function approximation and the functional-link net. *IEEE Transactions on Neural Networks and Learning Systems*, 6(6):1320–1329, 1995.

[45] M. Sajid, A. K. Malik, M. Tanveer, and PN Suganthan. Neuro-fuzzy random vector functional link neural network for classification and regression problems. *IEEE Transactions on Fuzzy Systems*, 32(5):2738–2749, 2024.

[46] Yongshan Zhang, Jia Wu, Zhihua Cai, Bo Du, and S Yu Philip. An unsupervised parameter learning model for RVFL neural network. *Neural Networks*, 112:85–97, 2019.

[47] Xin Li, Yu Yang, Niaoqing Hu, Zhe Cheng, and Junsheng Cheng. Discriminative manifold random vector functional link neural network for rolling bearing fault diagnosis. *Knowledge-Based Systems*, 211:106507, 2021.

[48] Wei Cui, Le Zhang, Bing Li, Jing Guo, Wei Meng, Haixia Wang, and Lihua Xie. Received signal strength based indoor positioning using a random vector functional link network. *IEEE Transactions on Industrial Informatics*, 14(5):1846–1855, 2017.

[49] Tatiana Chakravorti and Penke Satyanarayana. Non linear system identification using kernel based exponentially extended random vector functional link network. *Applied Soft Computing*, 89:106117, 2020.

[50] Peng-Bo Zhang and Zhi-Xin Yang. A new learning paradigm for random vector functional-link network: RVFL+. *Neural Networks*, 122:94–105, 2020.

[51] Wei Dai, Yanshuang Ao, Linna Zhou, Ping Zhou, and Xuesong Wang. Incremental learning paradigm with privileged information for random vector functional-link networks: IRVFL+. *Neural Computing and Applications*, 34(9):6847–6859, 2022.

[52] Weipeng Cao, Pengfei Yang, Zhong Ming, Shubin Cai, and Jiyong Zhang. An improved fuzziness based random vector functional link network for liver disease detection. In *2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)*, pages 42–48. IEEE, 2020.

[53] Ashwani Kumar Malik, MA Ganaie, and Muhammad Tanveer. Graph embedded intuitionistic fuzzy weighted random vector functional link network. In *2022 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 293–299. IEEE, 2022.

[54] A.K. Malik and M. Tanveer. Graph embedded ensemble deep randomized network for diagnosis of alzheimer's disease. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, pages 1–13, 2022.

[55] Raveendra Pilli, Tripti Goel, R Murugan, M. Tanveer, and PN Suganthan. Kernel ridge regression-based randomized network for brain age classification and estimation. *IEEE Transactions on Cognitive and Developmental Systems*, pages 1–10, 2024.

[56] Tiangang Zhou, Jun Zhang, and Lin Chen. Neural correlation of "global-first" topological perception: Anterior temporal lobe. *Brain imaging and Behavior*, 2:309–317, 2008.

[57] Shuyin Xia, Yunsheng Liu, Xin Ding, Guoyin Wang, Hong Yu, and Yuoguo Luo. Granular ball computing classifiers for efficient, scalable and robust learning. *Information Sciences*, 483:136–152, 2019.

[58] Shuyin Xia, Daowan Peng, Deyu Meng, Changqing Zhang, Guoyin Wang, Elisabeth Giem, Wei Wei, and Zizhong Chen. A fast adaptive k-means

with no bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44:87–99, 2020.

[59] Shuyin Xia, Shaoyuan Zheng, Guoyin Wang, Xinbo Gao, and Binggui Wang. Granular ball sampling for noisy label classification or imbalanced classification. *IEEE Transactions on Neural Networks and Learning Systems*, 34(4):2144–2155, 2023.

[60] GQ Zhang, Q Liu, and LH JI. Multi-granularity sentiment classification method based on sequential three-way decisions. *Computer Science*, 45(12):153–159, 2018.

[61] Witold Pedrycz. Allocation of information granularity in optimization and decision-making models: towards building the foundations of granular computing. *European Journal of Operational Research*, 232(1):137–145, 2014.

[62] D Miao and S Hu. Uncertainty analysis based on granular computing. *J. Northwest Univ*, 4:44–47, 2019.

[63] Shuyin Xia, Guoyin Wang, Xinbo Gao, and Xiaoli Peng. GBSVM: Granular-ball support vector machine. *arXiv preprint arXiv:2210.03120*, 2022.

[64] Jinglin Xu, Junwei Han, Feiping Nie, and Xuelong Li. Re-weighted discriminatively embedded k-means for multi-view clustering. *IEEE Transactions on Image Processing*, 26(6):3016–3027, 2017.

[65] Lynn Houthuys, Rocco Langone, and Johan AK Suykens. Multi-view least squares support vector machines classification. *Neurocomputing*, 282:78–88, 2018.

[66] Ruxin Xu and Huiru Wang. Multi-view learning with privileged weighted twin support vector machine. *Expert Systems with Applications*, 206:117787, 2022.

[67] Maoshan Liu, Vasile Palade, and Zhonglong Zheng. Learning the consensus and complementary information for large-scale multi-view clustering. *Neural Networks*, page 106103, 2024.

[68] Shuicheng Yan, Dong Xu, Benyu Zhang, Hong-Jiang Zhang, Qiang Yang, and Stephen Lin. Graph embedding and extensions: A general framework

for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):40–51, 2006.

[69] M. A. Ganaie, M. Sajid, A. K. Malik, and M. Tanveer. Graph embedded intuitionistic fuzzy random vector functional link neural network for class imbalance learning. *IEEE Transactions on Neural Networks and Learning Systems*, 10.1109/TNNLS.2024.3353531:1–10, 2024.

[70] Dheeru Dua and Casey Graff. UCI machine learning repository. 2017.

[71] J Derrac, S Garcia, L Sanchez, and F Herrera. KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic and Soft Computing*, 17:255–287, 2015.

[72] DR Musicant. NDC: normally distributed clustered datasets. *Computer Sciences Department, University of Wisconsin, Madison*, 1998.

[73] Mengjia Xu. Understanding graph embedding methods and their applications. *SIAM Review*, 63(4):825–853, 2021.

[74] Masashi Sugiyama. Dimensionality reduction of multimodal labeled data by local fisher discriminant analysis. *Journal of Machine Learning Research*, 8(5), 2007.

[75] Shuyin Xia, Xiaochuan Dai, Guoyin Wang, Xinbo Gao, and Elisabeth Giem. An efficient and adaptive granular-ball generation method in classification problem. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–13, 2022.

[76] Fan R. K. Chung. Spectral graph theory. 1996.

[77] Jing Zhao, Xijiong Xie, Xin Xu, and Shiliang Sun. Multi-view learning overview: Recent progress and new challenges. *Information Fusion*, 38:43–54, 2017.

[78] Chang Tang, Xinwang Liu, Xinzhong Zhu, En Zhu, Zhigang Luo, Lizhe Wang, and Wen Gao. Cgd: Multi-view clustering via cross-view graph diffusion. In *Proceedings of the AAAI conference on Artificial Intelligence*, volume 34, pages 5924–5931, 2020.

[79] Chang Tang, Xinzhong Zhu, Xinwang Liu, Miaomiao Li, Pichao Wang, Changqing Zhang, and Lizhe Wang. Learning a joint affinity graph for multiview subspace clustering. *IEEE Transactions on Multimedia*, 21(7):1724–1736, 2018.

[80] Meina Kan, Shiguang Shan, Haihong Zhang, Shihong Lao, and Xilin Chen. Multi-view discriminant analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1):188–194, 2015.

[81] Jingjing Li, Yue Wu, Jidong Zhao, and Ke Lu. Low-rank discriminant embedding for multiview learning. *IEEE Transactions on Cybernetics*, 47(11):3516–3529, 2016.

[82] Jason Farquhar, David Hardoon, Hongying Meng, John Shawe-Taylor, and Sandor Szedmak. Two view learning: Svm-2k, theory and practice. *Advances in Neural Information Processing Systems*, 18, 2005.

[83] Xijiong Xie and Shiliang Sun. Multi-view twin support vector machines. *Intelligent Data Analysis*, 19(4):701–712, 2015.

[84] Ning Chen, Jun Zhu, and Eric Xing. Predictive subspace learning for multi-view data: a large margin approach. *Advances in Neural Information Processing Systems*, 23, 2010.

[85] Jie Sun, Hamido Fujita, Yujiao Zheng, and Wenguo Ai. Multi-class financial distress prediction based on support vector machines integrated with the decomposition and fusion methods. *Information Sciences*, 559:153–170, 2021.

[86] Xiaobo Zhang, Yan Yang, Tianrui Li, Yiling Zhang, Hao Wang, and Hamido Fujita. Cmc: a consensus multi-view clustering model for predicting alzheimer's disease progression. *Computer Methods and Programs in Biomedicine*, 199:105895, 2021.

[87] Zhu Zhang, Xuan Wei, Xiaolong Zheng, and Daniel Dajun Zeng. Predicting product adoption intentions: An integrated behavioral model-inspired multiview learning approach. *Information & Management*, 58(7):103484, 2021.

[88] S Sathiya Keerthi and Shirish K Shevade. SMO algorithm for least-squares SVM formulations. *Neural Computation*, 15(2):487–507, 2003.

[89] Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013.

[90] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

[91] Alexandros Iosifidis, Anastasios Tefas, and Ioannis Pitas. Graph embedded extreme learning machine. *IEEE Transactions on Cybernetics*, 46(1):311–324, 2015.

[92] Kun Hu, Yingyuan Xiao, Wenguang Zheng, Wenxin Zhu, and Ching-Hsien Hsu. Multiview large margin distribution machine. *IEEE Transactions on Neural Networks and Learning Systems*, 2024, 10.1109/TNNLS.2023.3349142.