Minimum Variance Embedding-Based Least-Squares Methods for One-Class Classification

MS (Research) Thesis

By

Pratik Kumar Mishra

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

JUNE 2020

Minimum Variance Embedding-Based Least-Squares Methods for One-Class Classification

A THESIS

Submitted in fulfillment of the

requirements for the award of the degree

of

Master of Science (Research)

by

Pratik Kumar Mishra

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

JUNE 2020

INDIAN INSTITUTE OF TECHNOLOGY INDORE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled **Minimum Variance Embedding-Based Least-Squares Methods for One-Class Classification** in the fulfillment of the requirements for the award of the degree of **Master of Science (Research)** and submitted in the **Department of Computer Science and Engineering, Indian Institute of Technology Indore,** is an authentic record of my own work carried out during the time period from July 2018 to June 2020 under the supervision of Dr. Aruna Tiwari, Associate Professor, Indian Institute of Technology Indore, Indore, India.

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

Signature of the Student (with date)
(Pratik Kumar Mishra)

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

Signature of the Supervisor of
MS (Research) thesis (with date)
(Dr. Aruna Tiwari)

Pratik Kumar Mishra has successfully given his MS (Research) Oral Examination held on ____________________

Signature of Chairperson (OEB) (with date)   Signature of Thesis Supervisor (with date)

Signature of Convener, DPGC (with date)   Signature of Head of Department (with date)

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my heartfelt gratitude to a number of persons who in one or the other way contributed by making this time as learnable, enjoyable, and bearable. At first, I would like to thank my supervisor **Dr. Aruna Tiwari**, who was a constant source of inspiration during my work. Without her constant guidance and research directions, this research work could not be completed. Her continuous support and encouragement has motivated me to remain streamlined in my research work.

I am thankful to **Dr. Kapil Ahuja** and **Dr. Amod C. Umarikar**, my research progress committee members for taking out some valuable time to evaluate my progress all these years. Their valuable comments and suggestions helped me to improve my work at various stages. I am also grateful to HOD of Computer Science for his help and support.

I am especially grateful to Dr. M. Tanveer for his insightful comments and encouragement, but also for the hard question which incented me to widen my research from various perspectives.

My sincere acknowledgement and respect to **Prof. Pradeep Mathur**, the former Director, and **Prof. Neelesh Kumar Jain**, the Officiating Director, Indian Institute of Technology Indore for providing me the opportunity to explore my research capabilities at Indian Institute of Technology Indore.

I would like to appreciate the guidance and support of my seniors Dr. Chandan Gautam and Mr. Vikas Chauhan. I would like to appreciate the fine company of my dearest colleagues and friends, especially, Siddharth Gupta, Shubham, Umar Aalam, Neha Sharma, Prarthi Jain, and Anagha Bhople.

I am also grateful to the institute staffs for their unfailing support and assistance, and the Government of India for funding the MS research.

I would like to express my heartfelt respect to my parents for their love, care and support they have provided to me throughout my life. Special thanks to my sister, and Masoom Rabbani, my mentor figure, as this thesis would not have been possible without the help of their support and encouragements.

Finally, I am thankful to all who directly or indirectly contributed, helped and supported me.

Pratik Kumar Mishra

To my family and friends

Abstract

In recent years, one-class classification (OCC) has been an area of extensive research for outlier or anomaly detection. OCC [1, 2, 3, 4] is different from conventional classification techniques. The conventional classification methods aim to classify the available data samples into one of many predetermined classes. However, OCC aims to predict whether a data sample belongs to the target class (or normal or positive class). The training is done using samples of only the target class. The samples that don't belong to the target class are termed as outliers. The approaches followed in OCC can be broadly classified [5, 6] into (i) Boundary framework-based (ii) Reconstruction framework-based (iii) Density framework-based. In the boundary framework-based approach, a one-class classifier tries to learn a boundary around the target class. In the reconstruction framework-based approach, the model utilizes the deviation in reconstruction error to differentiate the target class and outliers. In the density framework-based approach, assumptions are made regarding the density distribution of the data, and a threshold is set based on the density to identify the target class. In the thesis, we have utilized boundary and reconstruction-based approaches, and leveraged variance minimization to develop kernel regularized least-squares (KRL) based methods for OCC. The variance minimization helps to minimize the data dispersion of the target class and improves the generalization performance of the classifier. Variance minimization has been extensively used by researchers to develop binary [7, 8, 9] as well as multi-class [10, 11, 12] classifiers. Further, Mygdalis et. al. [13] exploited variance minimization to develop a boundary framework-based one-class classifier with kernel extreme learning machine at its base. However, variance minimization has not been explored for reconstruction framework-based KRL one-class classifiers in the past.

In this thesis, we propose two methods to leverage variance minimization for reconstruction and boundary framework-based KRL one-class classifiers. Both methods combine kernel learning and representation learning and utilize kernel autoencoders to learn essential information from the input data. The first method explores variance minimization for a reconstruction framework-based approach to OCC in a single-layer approach. In the second method, we have a multi-layer architecture composed of multiple reconstruction framework-based layers and a final boundary framework-based layer. We have utilized variance minimization at the first layer and used the boundary framework-based approach for OCC at the final layer. The contributions of this thesis involve the development of minimum variance KRL-based one-class classifiers by leveraging reconstruction and boundary frameworks in single-layer and multi-layer architectures. We meticulously analyze the performance of all the proposed methods and compare it with various state-of-the-art one-class classifiers using different performance evaluation criteria.

Keywords: Variance Minimization, One-Class Classification (OCC), Reconstruction Framework, Boundary Framework, Kernel Regularized Least-Squares (KRL).

List of Publications

Journals

Published:

1. Chandan Gautam, **Pratik K. Mishra**, Aruna Tiwari, Bharat Richhariya, Hari Mohan Pandey, Shuihua Wang, M. Tanveer, Alzheimer's Disease Neuroimaging Initiative. Minimum variance-embedded deep kernel regularized least squares method for one-class classification and its applications to biomedical data, *Neural Networks*, 123:191–216, 2020. doi: https://doi.org/10.1016/j.neunet.2019.12.001 (**IF: 5.785**)

Under Preparation:

1. **Pratik K Mishra**, Chandan Gautam, and Aruna Tiwari. Minimum variance embedded auto-associative kernel regularized least squares method for one-class classification. (To be submitted)

Contents

List of Figures

List of Tables

List of Abbreviations and Acronyms

1-NN 1-Nearest Neighbor

AN Adenosis

AD Alzheimer's Disease

CN Control Normal

DC Ductal Carcinoma

ELM Extreme Learning Machine

FA Fibroadenoma

KELM Kernel Extreme Learning Machine

k-NN k-Nearest Neighbor

KRL Kernel Regularized Least-squares

LSSVM Least Squares Support Vector Machine

LC Lobular Carcinoma

MST Minimum Spanning Tree

MC Mucinous Carcinoma

OCC One-class Classification

OCRF One Class Random Forests

OCSVM One-class Support Vector Machine

PC Papillary Carcinoma

PT Phyllodes Tumor

PCA Principal Component Analysis

RBF Radial Basis Function

sMRI structural Magnetic Resonance Image

SVDD Support Vector Data Description

SVM Support Vector Machine

TA Tubular Adenoma

Chapter 1

Introduction

Classification has been a frequently discussed problem for many decades. The type of classification can be binary, multi-class or one-class classification (OCC). In binary or multi-class classification, the aim of the classifier is to classify each sample into one of the many (or two for a binary classifier) predetermined classes. However, in OCC the samples of only one class is available in prior, and the challenge is to determine the classifying criteria for the samples of other unidentified classes using the information of only the given class. In the past, OCC has been frequently applied in various disciplines for novelty or outlier detection [14, 15, 16, 17, 18].

In this thesis, the focus is on OCC, where the task is to classify a sample as genuine or outlier. The class comprising of genuine samples is called the target (positive or normal) class, and the class comprising of outlier samples is called the outlier (negative or anomalous) class.

1.1 Background

In the traditional (binary or multi-class) classification, every sample is bound to be classified in one of the predefined classes. However, an issue arises when a test sample that doesn't belong to either of the predefined classes is encountered. Consider an example of a binary classifier that is trained to classify the samples into apples and mangoes. Even if the test sample is from a completely different domain, e.g., a cat, the classifier will always classify it as an apple or a mango, which is wrong in both cases. Hence, sometimes the task is not to allocate a test sample into one of the predefined classes but to decide if it belongs to a particular class. This brings the need for OCC.

The OCC problem fundamentally differs from the traditional (binary or multi-class) classification problem [5]. In binary classification, the information from the available samples of the two classes is used to construct a decision boundary during training. But, in OCC, it is assumed that the samples of only the target class are available during training. The objective is to construct a discrimination boundary around the target class such that it accepts as many target class objects as possible while minimizing the probability of accepting the outliers. Since OCC uses the information from only the target class, it is a challenge to determine how tight the boundary should fit in each of the directions around the target data. This makes the problem of OCC more challenging than the traditional binary classification problem. Generally, OCC is helpful in cases where the samples of the target class are abundant, but the other class has very few or no samples. There can be various reasons for the unavailability of data, like the difficulty of collection, high computational cost, infrequent event, lack of reproducibility, etc. OCC is particularly helpful for real-world scenarios, where collecting data for the outlier (or negative) class is much more difficult and expensive than the normal (or positive) class due to various reasons described previously. We provide cases of such real-world scenarios where OCC is quite helpful. One scenario is the detection of faults in a machine [19]. A classifier should be able to detect when a machine is showing irregular/faulty behavior. It is easy to gather data on normal working (positive class) of a machine. However, it is quite difficult to have sufficient data on the faulty behavior (negative class) as most of the faults may not have surfaced already. Additionally, we cannot wait for the faults to occur as it may involve high expense, danger to human operators, or machine malfunction. Another situation is identifying if a person is healthy or not [20]. In such a case, collecting data for a healthy case is quite easy as the characteristics of all the healthy persons are quite similar. However, it is quite difficult to collect information about all the unhealthy cases (negative class) in the world. In such a case, a classifier needs to be trained using the healthy (positive) class samples only, for which a one-class classifier is the best-suited solution as it needs samples of only one class. Building intrusion detection [21] systems for large secured networks is a well-known problem. It is quite easy to collect data on the everyday normal behavior of a network. However, it is impossible to collect data on every intrusion attack as the signature of not all the attacks are known. In such a case, a one-class classifier can be used to design an anomaly-based intrusion detection system, which leverages the normal behavior of a network to identify any new attack. Additionally, OCC has been applied to other real-world scenarios such as authorship verification [22], document classification [23], video surveillance [24, 25], fabric defect detection [26], and fMRI response [27].

The earliest work on OCC can be recorded back to Minter [28], who used the term 'single-class classification' to describe the learning of a Bayes classifier using labeled data from only the class of interest. Later, Moya et al. [29] originated the term 'one-class classification' in reference to the application of the classifier for target recognition. Over the years, other terms such as Novelty Detection [30], Outlier Detection [31], or Concept Learning [32] have been used as a result of different applications of OCC. Japkowicz [32] proposed an auto-association-based approach for OCC, which is a neural network-based approach and termed it as 'concept learning in the absence of counterexamples'. In the same year, kernel-based one-class classifiers were proposed [1, 33]. The kernel-based one-class classifiers can be broadly classified into two categories[5]: (i) reconstruction-based (ii) boundary-based. The reconstruction-based one-class classifiers try to reconstruct the input data at the output layer while keeping the essential information intact. The classification is performed based on the deviation in reconstruction error between the target and outlier data. Hoffmann [34] proposed a reconstruction-based one-class classifier leveraging the kernel principal component analysis as the base method. The boundary-based one-class classifiers aim to construct a boundary using the structure of the data. Schölkopf et al. [1] proposed one-class support vector machine (OCSVM) and Tax and Duin [35, 33] proposed support vector data description (SVDD). Both methods are boundary-based one-class classifiers with support vector machine (SVM) as a base method. The SVM-based classifiers are computationally expensive owing to their iterative nature of learning. Choi [36] and Leng et al. [37] further addressed this issue by proposing a least-squares-based one-class classifier with the least-squares SVM and kernel extreme learning machine (KELM) at the base, respectively. The KELM-based one-class classifier is further enabled with minimum variance embedding within its optimization problem [13]. Gautam et al. proposed a kernel learning-based autoencoder for OCC that detects outliers using the deviation in reconstruction error. Dai et al. [38] proposed a multi-layer KELM-based one-class classifier that leverages multiple reconstruction-based layers to minimize the reconstruction error, followed by a final OCC layer.

1.2 Motivation

A usual approach in OCC is to define a discrimination boundary around the target class data. However, the more data points spread out, the more difficult it becomes to determine how tight the boundary should fit in each of the directions. Over the years, efforts have been made to minimize the data dispersion in order to improve the efficiency of different one-class classifiers. Since variance is a measure of how far a dataset is spread out, variance minimization has shown promising results for different machine learning tasks. Researchers have applied variance minimization for binary [8, 9] as well as multi-class classification [10, 11]. Variance minimization has also been applied for OCC in the past [13] by developing a KELM-based one-class classifier that utilized minimum variance information for a boundary-based approach to OCC. However, variance minimization has not been explored for a reconstruction-based approach to OCC in the past. In this thesis, we venture to explore minimum variance embedding for a reconstruction-based approach to OCC. Hence, we propose minimum variance embedded single-layer and multi-layer one-class classifiers that utilizes both reconstruction and boundary-based frameworks by considering kernel regularized least-squares (KRL) as a base classifier.

1.3 Objectives

In this thesis, we aim to achieve the following objectives:

(1) To leverage minimum variance embedding to develop a reconstruction-based one-class classifier using kernel learning.

(2) To develop a model that utilizes the minimum variance embedding to minimize the data dispersion, and combine the advantages of the reconstruction-based framework and the boundary-based OCC approach in a single architecture.

(3) To explore the effectiveness of the proposed method for the identification of Alzheimer's and Breast Cancer diseases.

1.4 Thesis Contributions

A brief overview of our research contributions is provided below, and more details are available in the later chapters.

Contribution I: Minimum Variance Embedded Auto-associative KRL-based Method for One-class Classification

The variance minimization helps to take advantage of the underlying structural information of the data, leading to better classification. In the past, variance minimization was explored for kernel learning-based OCC following the boundary-based approach [13]. In this thesis, we have explored variance minimization by embedding minimum variance information in a reconstruction-based one-class classifier by developing the minimum variance embedded auto-associative KRL-based one-class classifier (VAAKRL). The proposed method leverages minimum variance embedding to exploit the structural information of the underlying data in a single-layer architecture and utilizes the reconstruction error to identify the outliers.

Contribution II: Minimum Variance Embedded Deep KRL-based Method for One-class Classification

We have further examined variance minimization in a multi-layer architecture by combining reconstruction-based and boundary-based frameworks. Thus, we have proposed the minimum variance embedded deep KRL-based one-class classifier (DKRLVOC). The proposed multi-layer architecture is formed by stacking multiple KRL-based autoencoders (reconstruction-based) sequentially with minimum variance embedding at the initial layer, followed by a KRL-based one-class classifier (boundary-based) at the final layer. The stacked autoencoders reconstruct the key information at the intermediate layers and enable better representation of data. The use of multiple reconstruction-based layers and the final boundary-based OCC layer enables the model to classify data more precisely.

Contribution III: Application of DKRLVOC for the identification of Alzheimer's and Breast Cancer Diseases

The proposed method, DKRLVOC, is applied for the identification of Alzheimer's and Breast Cancer diseases. We have utilized structural magnetic resonance imaging data[1] to train the one-class classifier to identify Alzheimer's disease, and histopathological image data [39] for learning the difference between cancerous and non-cancerous tumors in case of Breast Cancer disease.

1.5 Organization of the Thesis

This thesis is organized into six chapters. A summary of each chapter is provided below:

Chapter 1 (Introduction)

This chapter provides the background knowledge of OCC, the motivation behind our work, and the contributions of this thesis.

Chapter 2 (Literature Survey)

[1] adni.loni.usc.edu

This chapter provides a detailed literature survey on how variance minimization has been applied to different machine learning tasks, along with a survey on different KRL-based one-class classifiers. It also provides the details of different metrics used for performance evaluation of proposed methods.

Chapter 3 (Minimum Variance Embedded Auto-associative KRL-based Method for One-class Classification)

In this chapter, we propose the minimum variance embedded auto-associative KRL-based one-class classifier (VAAKRL). The proposed method incorporates the concept of minimum variance embedding with representation learning, and aims at minimizing the dispersion of the data and the reconstruction error, simultaneously. We experiment the proposed method on 14 benchmark datasets and compare their performance with different state-of-the-art one-class classifiers.

Chapter 4 (Minimum Variance Embedded Deep KRL-based Method for One-class Classification)

In this chapter, we propose the minimum variance embedded deep KRL-based one-class classifier (DKRLVOC). The proposed method uses multiple KRL-based autoencoders stacked in a sequential manner with a one-class classifier at the last layer. It leverages minimum variance embedding to minimize the data dispersion, and the multi-layer approach helps to combine both reconstruction and boundary frameworks in a single architecture. We experiment the proposed method on 24 benchmark datasets and compare their performance with various existing one-class classifiers.

Chapter 5 (Application of DKRLVOC: Identification of Alzheimer's and Breast Cancer Diseases)

In this chapter, we apply DKRLVOC for the detection of Alzheimer's disease using structural magnetic resonance imaging data and Breast Cancer disease using histopathological image data.

Chapter 6 (Conclusions and Future Work)

This chapter provides a brief description of the contributions of this thesis and the possible future scope of our work.

Chapter 2

Literature Survey

This chapter provides a detailed literature survey in four sections. Section 2.1 discusses the various existing approaches for OCC. Section 2.2 discusses the concept of variance and provides a survey on the existing work on variance minimization for different machine learning tasks. Section 2.3 discusses the kernel trick and the existing KRL-based one-class classifiers. Section 2.4 provides a brief review on autoencoder. Finally, Section 2.5 provides a survey on the various performance metrics that researchers have used for the analysis of the existing one-class classifiers and discusses the metrics used for performance evaluation of the proposed methods in this thesis.

2.1 One-Class Classification

Several methods have been proposed over the years to tackle the problem of OCC. These methods differ in their approach to exploit different characteristics of the data. In a broad sense, the exiting approaches for OCC can be classified into three types [5, 6]:

(1) **Density-based approach:** This approach involves making assumptions regarding the probability density of the data, followed by setting a threshold based on the density. The samples whose estimated probability is lesser than the threshold are classified as outliers. It assumes that the target class samples are very likely to appear in areas of high density. However, this approach requires a large number

of training samples. The approach is very advantageous when a probability model with low bias is assumed, and the sample size is sufficient.

(2) **Reconstruction-based approach:** The original purpose of reconstruction-based methods is to model the data. They are used to obtain a more effective representation of the data. The reconstructed data suffers from less noise as compared to the original data. When using the reconstruction-based approach for OCC, it is assumed that the outliers do not satisfy the assumptions about the target distribution. Hence, the encoded representation for the outlier data should be worse than the target data, and the reconstruction error for the outliers should be high. A one-class classifier obtains an empirical threshold using the training set. If the reconstruction error is less than the threshold, then the sample belongs to the target class, otherwise the outlier class.

(3) **Boundary-based approach:** In the boundary-based approach, a closed boundary around the target class is optimized. The samples that lie within the boundary belongs to the target class. This approach doesn't depend on the density distribution of the data. Most of the boundary-based methods stress towards a minimal volume solution. However, the extent of the minimal volume depends on the fit of the method to the data. In comparison to the density-based methods, the boundary-based approach requires less number of samples.

Further, in Section 2.2, we discuss the concept of variance and the existing works in the field of variance minimization.

2.2 Variance and its usage in Machine Learning

The section discusses the concept of variance, followed by a survey on variance minimization, which has been utilized in the past for various machine learning tasks.

2.2.1 Variance

A common approach of the supervised or semi-supervised machine learning models is to learn an estimate of the true underlying function f, denoted as $\hat{f}$, that best fits the data. Since f is unknown, such models use the data available during training and the associated target values to learn $\hat{f}$, which can further be used to estimate the target values of the unseen data. Variance can be referred to as the amount of change in $\hat{f}$ when we estimate it using different datasets [40]. Since the training data is used to estimate the $\hat{f}$, different training datasets will yield different $\hat{f}$. In ideal scenarios, the estimated function $\hat{f}$ should not vary too much with a change in training datasets. However, in case a method has high variance, then any small change in the training data can result in large changes in $\hat{f}$. In general, the methods which are more flexible, have higher variance. This can be understood from Figure 2.1. In the figure, the blue curve is more flexible and follows the data points very closely. It has high variance as any change in data points will result in considerable variation in estimate $\hat{f}$. However,

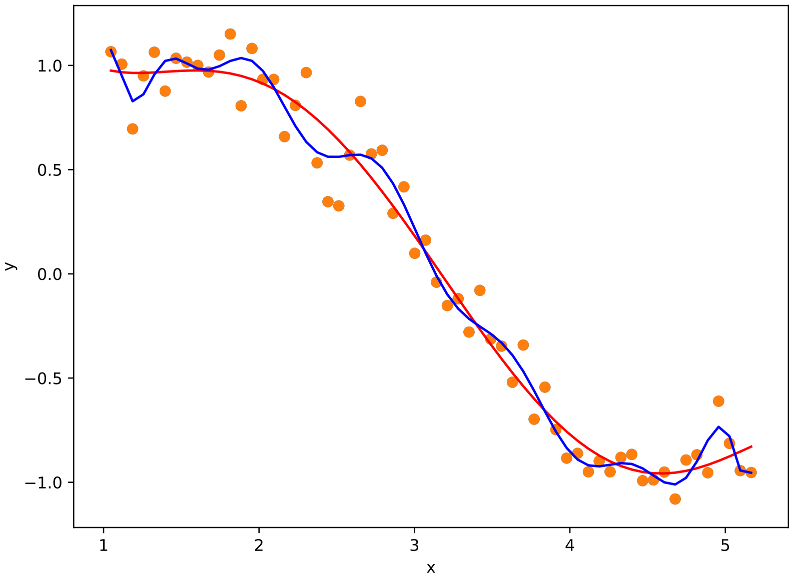


Figure 2.1: Two estimates of true function are shown. The blue curve follows the data points more closely relative to the red curve.

the red curve is relatively inflexible and has low variance, as moving any single data point will likely result in small changes in the position of the curve.

More formally, the variance is a measure of the spread of the data distribution [41]. It can be expressed as the average of squared differences of the network output. In this thesis, we leverage minimum variance embedding in reconstruction-based KRL one-class classifiers to minimize the dispersion of data. This forces the network output weight to emphasize in regions of low variance and improves the generalization performance of the classifier. Further, we discuss the mathematical formulation of variance for the KRL-based one-class classifiers [13]. Taking the training input as, $\mathbf{X} = \{\boldsymbol{x}_i \,|\, \boldsymbol{x}_i \in \mathbb{R}^d, i = 1, 2, ..., \mathcal{N}\}$, the variance of the network output is expressed as,

$$\begin{aligned}
\mathbf{V} &= \frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} \left(\widehat{O}_i - \overline{O}\right) \left(\widehat{O}_i - \overline{O}\right)^T, \\
&= \frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} \left((\boldsymbol{\beta})^T \boldsymbol{h}(\boldsymbol{x}_i) - (\boldsymbol{\beta})^T \overline{\mathcal{H}}\right) \left((\boldsymbol{\beta})^T \boldsymbol{h}(\boldsymbol{x}_i) - (\boldsymbol{\beta})^T \overline{\mathcal{H}}\right)^T, \\
&= (\boldsymbol{\beta})^T \left(\frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} \left(\boldsymbol{h}(\boldsymbol{x}_i) - \overline{\mathcal{H}}\right) \left(\boldsymbol{h}(\boldsymbol{x}_i) - \overline{\mathcal{H}}\right)^T\right) \boldsymbol{\beta}, \\
&= (\boldsymbol{\beta})^T \mathbf{V}_C \, \boldsymbol{\beta}, \qquad (2.1)
\end{aligned}$$

where, $\widehat{O}_i$ is the network output, and $\boldsymbol{h}(\boldsymbol{x}_i)$ is the non-linear feature mapping for a training sample $\boldsymbol{x}_i$. $\boldsymbol{\beta}$ is the network output weight, and $\overline{O} = \frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} \widehat{O}_i$ is the mean network output for all training samples. $\overline{\mathcal{H}} = \frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} \boldsymbol{h}(\boldsymbol{x}_i)$ is the mean vector of the samples in the non-linear feature space, and $\mathbf{V}_C$ is the class variance. Further, the class variance ($\mathbf{V}_C$) can be simplified as,

$$\begin{aligned}
\mathbf{V}_C &= \frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} (\boldsymbol{h}(\boldsymbol{x}_i) - \overline{\mathcal{H}})(\boldsymbol{h}(\boldsymbol{x}_i) - \overline{\mathcal{H}})^T \\
&= \frac{1}{\mathcal{N}} \mathcal{H} \left(\mathbb{I} - \frac{1}{\mathcal{N}} \mathbf{a}\mathbf{a}^T\right) (\mathcal{H})^T \\
&= \mathcal{H} \boldsymbol{\mathcal{M}} \, (\mathcal{H})^T, \qquad (2.2)
\end{aligned}$$

where, $\mathbb{I}$ is an identity matrix, $\mathbf{a}$ is a vector of ones, and $\mathcal{H} =$

$[\boldsymbol{h}(\mathbf{x}_1), \boldsymbol{h}(\mathbf{x}_2), \dots, \boldsymbol{h}(\mathbf{x}_{\mathcal{N}})]$. Further, the intra-class variance ($\mathbf{V}_S$) can be expressed as,

$$\mathbf{V}_S = \sum_{i=1}^{\mathcal{N}} \sum_{p=1}^{P} \frac{\mathcal{N}_p}{\mathcal{N}} \gamma_i^p (\boldsymbol{h}(\boldsymbol{x}_i) - \overline{\mathcal{H}})(\boldsymbol{h}(\boldsymbol{x}_i) - \overline{\mathcal{H}})^T, \tag{2.3}$$

where, $\mathcal{N}_p$ denotes the number of samples that belongs to the cluster p, and γ_i^p denotes if $\boldsymbol{x}_i$ belongs to cluster p or not. The data is grouped into sub-classes using a clustering method like k-means.

Further, in Section 2.2.2, we provide a survey on variance minimization applied for a range of machine learning applications.

2.2.2 Variance Minimization

Over the years, the researchers have tried to minimize the data dispersion to improve the generalization performance of the classifiers. Since variance is the measure of the spread of data distribution, minimizing the variance has shown promising results for different machine learning tasks. We provide a survey on variance minimization applied to solve different types of machine learning tasks. Warmuth and Kuzmin [42] designed two online variance minimization problems. In the first problem, they measured the variance along a probability vector associated with the problem of minimizing risk in the stock portfolio, while in the second problem, the variance was measured along an arbitrary direction. Additionally, they prove bounds on the total variance incurred by the online algorithms. Hofmann et. al. [43] proposed a method to overcome the slow convergence rate of stochastic gradient descent algorithms. They explored algorithms that can exploit the surrounding structure in the training data to leverage information about past stochastic gradients across data points and defined a family of stochastic gradient descent algorithms to minimize variance. Variance minimization has been employed for the task of regression with kernel learning. In the method proposed by Ormándi [44], weight variance minimization was integrated into the objective function of a least-squares SVM for the task of time series analysis. They intended to adjust the weight of the variance of the error in the kernel feature space. Xiaofei et. al. [45] proposed a variance minimization criterion for feature selection using laplacian regularization. They considered the feature selection problem in unsupervised learning scenarios and selected features such that the size of the parameter covariance matrix of the regularized regression model is minimized. They used trace and determinant operators to measure the size of the covariance matrix. Further, Jean et. al. [46] proposed a semi-supervised deep kernel learning regression model with unlabeled data that minimized predictive variance. The method takes advantage of representation learning-based neural networks and the probabilistic modeling power of gaussian processes. Variance minimization has been applied for the task of binary classification [7, 8, 9] with kernel learning by leveraging support vector machines (SVM). Wang et. al. [7] proposed a minimum class locality preserving variance SVM that aimed to perform binary classification and introduced the idea of locality preserving projections. Further, Chen et. al. [8] proposed a recursive projection twin SVM-based binary classifier via within-class variance minimization. They tried to separate the projected samples of one class from the other class by finding a weight vector direction for each class. They used within-class variance and the distance between the mean of projected class to measure the separability. Their idea was to search for a projection axis for each class such that the within-class variance of one class is minimized, and the projected samples of other class are scattered wide. Further, Ye et. al. [9] proposed the least-squares SVM-based binary classifier via maximum one-class within-class variance. They expected to keep the genuine geometric interpretation of generalized proximal SVM in the least-squares twin SVM. Researchers have applied variance minimization in the field of multi-class classification [10, 11, 12] as well. Ji and Han [10] proposed a variance minimization criterion-based multi-class classifier for active learning on graphs. They labeled the nodes such that the variance of unlabeled data distribution and the expected prediction error was minimized. Further, Iosifidis et. al. [11, 12] proposed a minimum variance-based extreme learning machine for the task of human action recognition by adopting shape and motion information and the Bag-of-Features-based action representation. Variance minimization has also been applied for one-class classification. Mygdalis et. al. [13] proposed a minimum variance embedded boundary-based one-class classifier for facial image analysis that used

KELM as a base classifier. The embedding improved the generalization performance of the classifier by minimizing the class variance. The minimum variance embedding has not been explored for the reconstruction-based approach for OCC in the past. In this thesis, we explore minimum variance embedding by developing single-layer and multi-layer reconstruction-based one-class KRL classifiers.

Since the proposed methods utilize kernel learning and are based on KRL, we discuss the kernel trick and the existing KRL-based one-class classifiers in Section 2.3.

2.3 Kernel Learning

Kernel learning has been employed in the past for different types of classification, namely, binary, multi-class, and one-class classification [47]. In this thesis, we focus on kernel learning for one-class classification. In 1999, Schölkopf et al. [1] proposed a kernel learning-based OCC model for novelty detection and coined it as one-class support vector machine (OCSVM). The proposed model was developed by taking SVM as the base classifier. Further, Tax and Duin proposed another kernel learning-based one-class classifier taking SVM as the base classifier, which is popularly known as support vector domain description [33] or support vector data description [35] (SVDD). Though both OCSVM and SVDD were developed by taking SVM as the base classifier, the working methodology of both the methods is quite different. OCSVM uses a hyperplane to separate the target class samples from the origin in the feature space. Further, the model maximizes the distance of the hyperplane from the origin. In contrast to the use of hyperplane in OCSVM, SVDD constructs a hypersphere around the target class data in the feature space. The radius of the hypersphere is minimized to enclose the maximum number of target class data points under the minimum radius. A one-class classifier is also called a data descriptor as it describes the characteristics of the data and performs classification based on the data description. The OCSVM and SVDD are domain-based one-class classifiers. They describe the boundary or domain of the target class data points and are insensitive to the underlying density of the data. Thus, they can be particularly helpful when the density distribution of the target class is unknown. However, they need a sufficient amount of samples from the target class during training to describe the domain of the target class.

Further, we discuss the kernel trick in Section 2.3.1, followed by a discussion on the existing KRL-based one-class classifiers in Section 2.3.2. Since the proposed methods in this thesis are based on the existing least-squares kernel autoencoder-based one-class classifier [48], we discuss its formulation in Section 2.3.3.

2.3.1 Kernel Trick

In the complex classification tasks, there are situations where it is not possible to linearly separate the available data in the original feature space. The kernel trick is used to tackle such situations. The kernel trick [49, 41] grants a mechanism to manipulate the linearly inseparable data in the original feature space, and project it in a higher-dimensional space. The projected data can become linearly separable in the higher dimensional space. Further, a hyperplane can be used to separate the projected data.

The kernel trick provides a link from linearity to non-linearity for algorithms that can be expressed in terms of dot products between two vectors. The data is transformed by mapping the input data onto a higher dimensional space. Hence, a linear algorithm that directly operates on the transformed data will actually behave non-linearly in the original input space. The kernel trick is based on the inner product of the samples mapped onto a feature space $\boldsymbol{h}(\boldsymbol{x})$.

Definition: A kernel function $K(\boldsymbol{x}, \boldsymbol{y})$ can be expressed as an inner product in feature space and is denoted as:

$$K(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{h}(\boldsymbol{x}).\boldsymbol{h}(\boldsymbol{y}) \tag{2.4}$$

where, $\boldsymbol{h}(.)$ is the non-linear feature mapping. A kernel function is used to generate a kernel or gram matrix. Any function can be treated as a kernel function if and only if it satisfies Mercer's theorem [50].

Mercer's Theorem: A symmetric function $K(\boldsymbol{x}, \boldsymbol{y})$ can be expressed as an inner product $K(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{h}(\boldsymbol{x}).\boldsymbol{h}(\boldsymbol{y})$, for some non-linear feature mapping $\boldsymbol{h}(.)$ if and only if $K(\boldsymbol{x}, \boldsymbol{y})$ is positive semidefinite.

The kernel trick is handy as we don't need to actually compute the mapping in equation 2.4. Any algorithm can leverage the kernel trick, under the condition that the algorithm can be expressed in terms of an inner product between two vectors. This is where we use the trick: *wherever an inner product is used within the algorithmic expression, it is replaced with a kernel function.* Hence, we can avoid to explicitly compute the mapping by using a kernel function. Thus, the data can be mapped onto a higher dimensional space without explicitly mapping the input points into this space.

The proposed methods in this thesis utilize KRL as a base classifier; hence we discuss the existing KRL-based one-class classifiers in Section 2.3.2.

2.3.2 KRL-based One-class Classifiers

The SVM-based one-class classifiers are computationally expensive as they are involved in solving a quadratic optimization problem. Least-squares was introduced to SVM formulation to handle this issue and named as least-squares SVM (LSSVM) [51, 52, 36]. The idea of this least-squares is directly linked to the kernel regularized least-squares (KRL) [53, 54]. The LSSVM with zero bias is identical to KRL. This is evident from the formulation of LSSVM in equation (2.5), and KRL in equation (2.7) and (2.8). The optimization problem of another popular method, namely, extreme learning machine (ELM) [55] was reformulated for the kernel [56], leading to an identical optimization problem as KRL. The kernel formulation for ELM is referred to as kernel extreme learning machine (KELM). Below, we provide a brief analysis of the optimization problem of LSSVM, KELM, and KRL to facilitate a better understanding of the differences between these methods.

Given a training set $\{\boldsymbol{x}_i, y_i\}_{i=1,2,\ldots,\mathcal{N}}$, where $\boldsymbol{x}_i$ denotes i^{th} training sample, and y_i denotes the target value for the i^{th} sample. We provide a brief analysis on the optimization problem of LSSVM, KELM, and KRL as follows,

Optimization problem of LSSVM:

$$\begin{aligned} &\min_{\boldsymbol{\omega}, e_i} \frac{1}{2} \|\boldsymbol{\omega}\|^2 + \frac{C}{2} \sum_{i=1}^{N} \|e_i\|_2^2 \\ &s.t.\ \boldsymbol{\omega^T \phi_i} + b = y_i - e_i,\ i = 1, 2, ..., N, \end{aligned} \tag{2.5}$$

where, $\boldsymbol{\omega}$ is the weight coefficients, $\phi(.)$ is the mapping in the feature space, e_i denotes training error for the i^{th} sample, and C is a regularization parameter.

Optimization problem of KELM:

$$\begin{aligned} &\min_{\boldsymbol{\beta}, e_i} \frac{1}{2} \|\boldsymbol{\beta}\|^2 + \frac{C}{2} \sum_{i=1}^{N} \|e_i\|_2^2 \\ &s.t.\ \boldsymbol{\beta^T h(x_i)} = y_i - e_i,\ i = 1, 2, ..., N, \end{aligned} \tag{2.6}$$

where $\boldsymbol{\beta}$ is the network output weight (i.e., weight coefficients), and $h(.)$ is the mapping in KELM feature space. The notations used in the equation 2.6 is consistent with the KELM papers [57, 56].

Optimization problem of KRL can be represented in two ways:

First way,

$$\begin{aligned} &\min_{\boldsymbol{\beta}, e_i} \frac{1}{2} \|\boldsymbol{\beta}\|^2 + \frac{C}{2} \sum_{i=1}^{N} \|e_i\|_2^2 \\ &s.t.\ \boldsymbol{\beta^T h(x_i)} = y_i - e_i,\ i = 1, 2, ..., N. \end{aligned} \tag{2.7}$$

Second way,

$$\begin{aligned} &\min_{\boldsymbol{\beta}, e_i} \frac{C}{2} \|\boldsymbol{\beta}\|^2 + \frac{1}{2} \sum_{i=1}^{N} \|e_i\|_2^2 \\ &s.t.\ \boldsymbol{\beta^T h(x_i)} = y_i - e_i,\ i = 1, 2, ..., N. \end{aligned} \tag{2.8}$$

Comparing equations (2.7) and (2.8), we can find that in equation (2.7), C is associated with the error term, while in equation (2.8), C is associated with the weight term. C is the regularization parameter, and is responsible to control the trade-off between the square loss, and the norm of the output weight. Hence, C can be associated with either the error, or the weight term. It will not affect the solution. Further, comparing equations (2.6) and (2.7), the formulations for KELM and KRL are identical. Some researchers have used different notation for the feature mapping in the formulation of KRL; however, the optimization problem will not change just due to the use of different notation for the same thing. Hence, both KELM and KRL yield the same solution. When comparing equations (2.5) and (2.7), we find that the optimization problems of LSSVM and KRL differ in two aspects. The first difference is the notation used for the weights (i.e., $\boldsymbol{\omega}$ or $\boldsymbol{\beta}$) and the feature mapping (i.e., $\boldsymbol{\phi_i}$ or $\boldsymbol{h(x_i)}$). However, a difference in notation doesn't change the solution obtained. The second difference is the presence of the term b on the left side of the constraints of LSSVM in the equation (2.5). We can obtain the formulation of KRL from LSSVM by simply substituting b = 0 in the equation (2.5). Hence, we can conclude that KRL is equivalent to LSSVM without bias. From the above discussion, we can concur that mathematically KRL is equivalent to KELM and LSSVM without bias.

The proposed methods of this thesis are the variants of KRL and KELM. Since KRL is an older and more generic name compared to KELM, we use the name KRL instead of KELM in this thesis to describe the existing and the proposed methods.

The KRL-based models follow a non-iterative approach to learning by solving a linear system. Hence, they have received quite an attention from the researchers over the years. The KRL-based one-class classifiers can be classified into two categories, namely, (i) without Variance Minimization (ii) with Variance Minimization.

(1) **Without Variance Minimization:** In the past, researchers have employed KRL-based classifiers to solve OCC problems without incorporating minimum variance information. Leng et al. [37] proposed a one-class classifier with KELM as its base, which followed the boundary-based approach for OCC. Later, Gautam et al. [48] proposed a least-squares kernel autoencoder-based one-class classifier that followed the reconstruction-based approach for OCC. Dai et al. [38] further proposed a multi-layer framework-based one-class classifier that incorporated the advantages of both the reconstruction-based layers and the boundary-based approach to OCC in a single model. The KRL-based one-class classifiers have also

been applied for anomaly detection in gas turbine combustors [58] and videos [59].

(2) **With Variance Minimization:** In the past, minimum variance information has been leveraged with KRL-based classifiers to solve OCC tasks. Mygdalis et al. [13] proposed a minimum variance embedded KRL-based one-class classifier that followed a boundary-based approach to OCC and applied it for facial image analysis.

The proposed methods in this thesis are developed by embedding minimum variance information in the existing reconstruction-based OCC approach. This approach developed by Gautam et. al. [48] is referred to as the least-squares kernel autoencoder-based one-class classifier, to which we provide a brief discussion in Section 2.3.3.

2.3.3 Least-squares Kernel Autoencoder-based method for OCC

In the least-squares kernel autoencoder-based one-class classifier [48], the data at the input layer is used for reconstruction at the output layer using kernelized feature mapping. Being auto-associative in nature, the input and output layer is made of an equal number of nodes. $\mathbf{X} = \{\boldsymbol{x}_i \mid \boldsymbol{x}_i \in \mathbb{R}^d, i = 1, 2, ..., \mathcal{N}\}$ is taken as the training input. The training involves calculating the optimum output weight by solving the following optimization problem,

$$\begin{gathered}\min_{\boldsymbol{\beta}, \mathbf{e}_i} \frac{1}{2} ||\boldsymbol{\beta}||_F^2 + \frac{1}{2} C \sum_{i=1}^{\mathcal{N}} ||\mathbf{e}_i||_2^2 \\ s.t \;\; \boldsymbol{\beta}^T \boldsymbol{h}(\boldsymbol{x}_i) = \boldsymbol{x}_i - \mathbf{e}_i, \; i = 1, 2, ..., \mathcal{N},\end{gathered} \tag{2.9}$$

where, $\mathbf{e}_i$ is the reconstruction error, and $\boldsymbol{h}(\boldsymbol{x}_i)$ is the mapping in the feature space for a training sample $\boldsymbol{x}_i$. C acts as the trade-off between minimizing the output weight norm and the reconstruction error. $||\,.\,||_F$ refers to the frobenius norm. Solving equation (2.9), the optimum output weight ($\boldsymbol{\beta}$) is derived as,

$$\boldsymbol{\beta} = \mathcal{H}^T \left(\frac{1}{C}\mathbb{I} + \mathcal{H}\mathcal{H}^T \right)^{-1} \mathbf{X}^T, \tag{2.10}$$

where, $\mathcal{H} = [\boldsymbol{h}(\boldsymbol{x}_1), \boldsymbol{h}(\boldsymbol{x}_2), \dots, \boldsymbol{h}(\boldsymbol{x}_{\mathcal{N}})]$, and $\mathbb{I}$ is an identity matrix. The network output ($\widehat{\boldsymbol{O}}$) is derived as,

$$\widehat{\boldsymbol{O}} = \boldsymbol{h}(\boldsymbol{x})\boldsymbol{\beta}. \tag{2.11}$$

Further, the kernel matrix ($\boldsymbol{\Omega}$) is defined as,

$$\begin{aligned} &\boldsymbol{\Omega} = \mathcal{H}\mathcal{H}^T \\ &s.t.\ \ \Omega_{j,k} = \boldsymbol{h}(\boldsymbol{x}_j)\boldsymbol{h}(\boldsymbol{x}_k) = K(\boldsymbol{x}_j, \boldsymbol{x}_k),\ \ j,k = 1, \dots, \mathcal{N}, \end{aligned} \tag{2.12}$$

where, K is a kernel function. Using kernelized feature mapping, the equation (2.10) is rewritten as,

$$\boldsymbol{\beta} = \left(\frac{1}{C}\mathbb{I} + \boldsymbol{\Omega} \right)^{-1} \mathbf{X}^T. \tag{2.13}$$

The network output for the training data is then calculated as,

$$\widehat{\boldsymbol{O}} = \begin{bmatrix} K(\boldsymbol{x}, \boldsymbol{x}_1) \\ \vdots \\ K(\boldsymbol{x}, \boldsymbol{x}_{\mathcal{N}}) \end{bmatrix}^T \left(\frac{1}{C}\mathbb{I} + \boldsymbol{\Omega} \right)^{-1} \mathbf{X}^T. \tag{2.14}$$

After obtaining the network output, a threshold value is calculated based on the reconstructed data at the output. The threshold value helps to decide whether a sample belongs to the target or the outlier class. For the least-squares kernel autoencoder-based one-class classifier, the threshold value is determined as follows,

(1) The reconstruction error ($\boldsymbol{s}$) is calculated as,

$$s_i = \sum_{j=1}^{d} (\widehat{O}_{ij} - x_{ij})^2,\ i = 1, 2, \dots, \mathcal{N}. \tag{2.15}$$

(2) The error vector ($\boldsymbol{s}$) is then sorted in decreasing order and is denoted as, $\boldsymbol{s}_{dec}$.

In OCC, the threshold is calculated by assuming a certain percent of training samples as outliers. Primarily, the most deviant samples are dismissed as outliers, as they are the most far from the target class distribution. Hence, the threshold (θ) is calculated as,

$$\theta = \boldsymbol{s}_{dec}(\lfloor \delta * \mathcal{N} \rfloor), \tag{2.16}$$

where, $0 \leq \delta \leq 1$ is the fraction of dismissal of training samples, and $\mathcal{N}$ is the number of training samples.

During testing, for a test sample $\boldsymbol{x}_t$, the network output ($\widehat{O}_t$) is determined as,

$$\widehat{O}_t = \begin{bmatrix} K(\boldsymbol{x}_t, \boldsymbol{x}_1) \\ \vdots \\ K(\boldsymbol{x}_t, \boldsymbol{x}_{\mathcal{N}}) \end{bmatrix}^T \boldsymbol{\beta}. \tag{2.17}$$

The loss (s_t) is then calculated as,

$$s_t = \sum_{j=1}^{d} \left(\widehat{O}_{tj} - x_{tj} \right)^2 . \tag{2.18}$$

Finally, the classification is done using the following rule,

$$\begin{aligned} sign(\theta - s_t) = &\ 1, \quad \boldsymbol{x}_t \ belongs\ to\ target\ class, \\ & -1, \quad \boldsymbol{x}_t \ belongs\ to\ outlier\ class. \end{aligned} \tag{2.19}$$

Overall, we have observed in Section 2.3 that the concept of minimum variance embedding has not been explored for a reconstruction-based approach to OCC. Hence, in this thesis, we propose the minimum variance embedded KRL-based method that follows a reconstruction-based approach to OCC. Further, we explore minimum variance embedding for a multi-layer approach developed by stacking multiple KRL-based autoencoders in sequence. Below in Section 2.4, we briefly discuss the concept of autoencoder and the relevant existing work.

2.4 Autoencoder

Autoencoders have been quite a topic of interest in the past decade [60, 61]. Autoencoders are an unsupervised learning technique that uses neural networks to reconstruct the input at the output layer [62]. It learns the latent information of the input and reconstructs the essential information at the output layer. They leverage representation learning to learn a compressed knowledge representation of the original input. They are usually restricted to learn only certain aspects of the input, which often helps them to learn useful properties of the data. Autoencoders can be considered to be a special case of feedforward neural networks. Similar to feedforward networks, they can be trained with the same techniques, that is, mini-batch gradient descent following gradients computed by back-propagation.

The idea of autoencoders has existed for quite some time [63, 64, 62]. Researchers have employed autoencoders for various tasks viz., dimensionality reduction [65], semi-supervised learning [66], representation and multi-task learning [67], pattern generation [68], noise reduction [69], anomaly detection /OCC [32, 70, 71, 72], information retrieval for texts [73] and images [74, 75], transfer learning [76], and generating higher resolution images [77]. In this thesis, we have incorporated stacked KRL-based autoencoders for non-iterative learning to design single-layer and multi-layer one-class classifiers, which is discussed in detail in the later chapters.

Further, in Section 2.5, we provide a survey on the various performance metrics that researchers have used for the analysis of the existing one-class classifiers and discuss the metrics used for the performance evaluation of the proposed methods in this thesis.

2.5 Performance Metrics

Researchers have used various performance metrics for evaluating the performance of one-class classifiers. Cohen et. al. [78] used the specificity and sensitivity metrics for the performance evaluation in case of nosocomial infection detection using a

OCSVM. Manevitz and Yousef [23] used the F$_1$ score and accuracy metrics for one-class document classification with the help of neural networks. Further, Kemmler et. al. [79] used the specificity and sensitivity metrics for the automatic identification of novel bacteria using Raman spectroscopy and gaussian processes. Leng et. al. [37] proposed the one-class extreme learning machine and utilized the F$_1$ score for the performance evaluation of his proposed method. Further, Mygdalis et. al. [13] and Iosifidis et. al. [80] utilized the g-mean metric for performance evaluation of their proposed one-class classifiers. In this thesis, we have adopted the following metrics for performance evaluation to test the novel one-class classifiers,

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{2.20}$$

$$Precision\ (P) = \frac{TP}{TP + FP}, \tag{2.21}$$

$$Recall\ (R)\ or\ Sensitivity = \frac{TP}{TP + FN}, \tag{2.22}$$

$$F_1\ score = \frac{2\ P.R}{P + R}, \tag{2.23}$$

$$G - mean = \sqrt{P.R}. \tag{2.24}$$

Above, *FN*, *FP*, *TN*, and *TP* represent false negative, false positive, true negative, and true positive, respectively. Accuracy denotes the fraction of all correct measurements. Precision reflects the fraction of correct positive measurements among all the predicted positives. Recall indicates the fraction of correct positive measurements among the actual positives. F$_1$ score and G-mean are the harmonic mean and geometric mean of precision and recall, respectively.

In the case of imbalanced datasets, it is possible to obtain good accuracy by classifying any given sample to the majority class. Taking an example, suppose we have an imbalanced dataset where 90 samples belong to the positive class, and 10 samples belong to the negative class. Now, a model may classify all the negative class samples incorrectly to the positive class. In such a case, the accuracy will be determined as 90%, even if all the negative class samples are incorrectly classified. Hence, accuracy fails to give an unbiased score for the performance of a model when the data is imbalanced. Precision and recall provide a better understanding of the efficiency of a model in such cases. In order to obtain an equilibrium between precision and recall, F_1 score, and g-mean are mostly used [23, 37, 13, 80] when the data is imbalanced.

In this thesis, we use the F_1 score as the first evaluation metric as most of the datasets that we have used for the experiments are imbalanced in nature. Since we have to compare multiple classifiers on various datasets, we compute the mean of all F_1 scores ($\eta_{\mathbf{F_1}}$) over all the datasets by taking inspiration from an existing work [81]. We consider $\eta_{\mathbf{F_1}}$ as the final evaluation measure to rank the classifiers as per their performance. For reference purpose, we also present the results based on accuracy, g-mean, precision, and recall metrics.

Chapter 3

Minimum Variance Embedded Auto-associative Kernel Regularized Least-Squares Method for One-class Classification.

In this chapter, we propose the Minimum **V**ariance Embedded **A**uto-**A**ssociative **KRL**-based method for OCC (VAAKRL). The proposed method is inspired by another reconstruction-based method for OCC [48], which used KELM as a base classifier. In the past, minimum variance embedding was applied for the boundary-based method for OCC [13] but was never explored for the reconstruction-based OCC method. We incorporate the concept of minimum variance embedding [13] with representation learning and propose the VAAKRL method that follows a reconstruction-based approach to OCC. VAAKRL aims at minimizing the reconstruction error and considers the dispersion of the data at the same time. It forces the network output weights to focus in low-variance regions. We experiment with the proposed method on 14 benchmark datasets and compare their performance with different existing one-class classifiers based on various performance metrics, which we have discussed in Section 2.5 of Chapter 2. We discuss the details of the proposed method further in Section 3.1.

3.1 Proposed Method: VAAKRL

The proposed method VAAKRL, is a minimum variance embedded reconstruction framework-based method for OCC. We have shown the architecture of the proposed method in Figure 3.1. It leverages variance minimization using a single-layer KRL-based autoencoder. VAAKRL minimizes the variance and the reconstruction error at the same time using the proposed optimization criterion. This improves the performance of the model, resulting in better classification. We perform the training of VAAKRL by reconstructing the input at the output layer. As it is a reconstruction-based method, we empirically determine a threshold during training by using the reconstruction error. As we perform the training using only the target class data, the reconstruction error for the outlier data should be relatively high as compared to the target data. Further, we introduce the formulation of the proposed method.

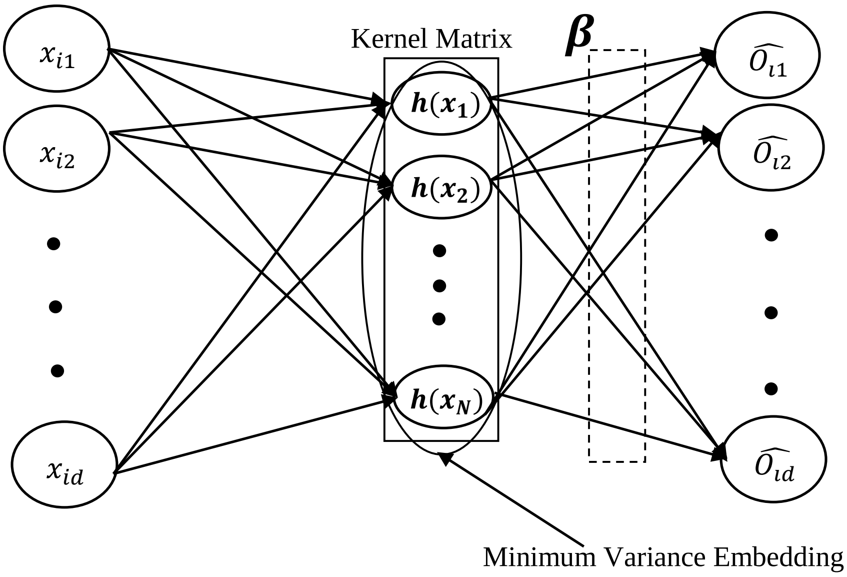

x_{i1}
x_{i2}
x_{id}
Kernel Matrix
$\boldsymbol{\beta}$
$\boldsymbol{h}(\boldsymbol{x_1})$
$\boldsymbol{h}(\boldsymbol{x_2})$
$\boldsymbol{h}(\boldsymbol{x_N})$
$\widehat{O_{\iota 1}}$
$\widehat{O_{\iota 2}}$
$\widehat{O_{\iota d}}$
Minimum Variance Embedding

Figure 3.1: Architecture of VAAKRL.

We consider the training input as, $\mathbf{X} = \{\boldsymbol{x}_i \mid \boldsymbol{x}_i \in \mathbb{R}^d, i = 1, 2, ..., \mathcal{N}\}$, where $\mathcal{N}$ refers to the number of training samples. Rewriting the expression of variance from

Section 2.2.1 of Chapter 2,

$$\begin{aligned}\mathbf{V} &= \frac{1}{\mathcal{N}}\sum_{i=1}^{\mathcal{N}}\left(\widehat{O}_i - \overline{O}\right)\left(\widehat{O}_i - \overline{O}\right)^T, \\ &= \frac{1}{\mathcal{N}}\sum_{i=1}^{\mathcal{N}}\left((\boldsymbol{\beta})^T \boldsymbol{h}(\boldsymbol{x}_i) - (\boldsymbol{\beta})^T\overline{\mathcal{H}}\right)\left((\boldsymbol{\beta})^T \boldsymbol{h}(\boldsymbol{x}_i) - (\boldsymbol{\beta})^T\overline{\mathcal{H}}\right)^T, \\ &= (\boldsymbol{\beta})^T\left(\frac{1}{\mathcal{N}}\sum_{i=1}^{\mathcal{N}}\left(\boldsymbol{h}(\boldsymbol{x}_i) - \overline{\mathcal{H}}\right)\left(\boldsymbol{h}(\boldsymbol{x}_i) - \overline{\mathcal{H}}\right)^T\right)\boldsymbol{\beta}, \\ &= (\boldsymbol{\beta})^T \mathbf{V}_C\, \boldsymbol{\beta}, \end{aligned} \tag{3.1}$$

where, $\widehat{O}_i$ is the network output, and $\boldsymbol{h}(\boldsymbol{x}_i)$ is the non-linear feature mapping for a training sample $\boldsymbol{x}_i$. $\boldsymbol{\beta}$ is the network output weight, and $\overline{O} = \frac{1}{\mathcal{N}}\sum_{i=1}^{\mathcal{N}}\widehat{O}_i$ is the mean network output for all training samples. $\overline{\mathcal{H}} = \frac{1}{\mathcal{N}}\sum_{i=1}^{\mathcal{N}}\boldsymbol{h}(\boldsymbol{x}_i)$ is the mean vector of the samples in the non-linear feature space, and $\mathbf{V}_C$ is the class variance. Further, the class variance ($\mathbf{V}_C$) can be simplified as,

$$\begin{aligned}\mathbf{V}_C &= \frac{1}{\mathcal{N}}\sum_{i=1}^{\mathcal{N}}(\boldsymbol{h}(\boldsymbol{x}_i) - \overline{\mathcal{H}})(\boldsymbol{h}(\boldsymbol{x}_i) - \overline{\mathcal{H}})^T \\ &= \frac{1}{\mathcal{N}}\mathcal{H}\left(\mathbb{I} - \frac{1}{\mathcal{N}}\mathbf{a}\mathbf{a}^T\right)(\mathcal{H})^T \\ &= \mathcal{H}\boldsymbol{\mathcal{M}}\,(\mathcal{H})^T, \end{aligned} \tag{3.2}$$

where, $\mathbb{I}$ is an identity matrix, $\mathbf{a}$ is a vector of ones, and $\mathcal{H} = [\boldsymbol{h}(\mathrm{x}_1), \boldsymbol{h}(\mathrm{x}_2), \ldots, \boldsymbol{h}(\mathrm{x}_{\mathcal{N}})]$.

After embedding minimum variance information, we propose the following optimization criterion to minimize the variance and the reconstruction error simultaneously.

$$\min_{\boldsymbol{\beta},\mathbf{e}_i}\frac{1}{2}\left|\left|(\boldsymbol{\beta})^T(\mathbf{V}_C + \lambda\mathbb{I})\,\boldsymbol{\beta}\right|\right|_F^2 + \frac{C}{2}\sum_{i=1}^{\mathcal{N}}\|\mathbf{e}_i\|_2^2 \tag{3.3}$$

$$s.t.\;\; (\boldsymbol{\beta})^T\boldsymbol{h}(\boldsymbol{x}_i) = \boldsymbol{x}_i - \mathbf{e}_i,\; i = 1, 2, ..., \mathcal{N},$$

where, $\mathbf{e}_i$ is the reconstruction error, and C is the regularization parameter. λ is the graph regularization parameter, and $||\,.\,||_F$ refers to frobenius norm. We substitute the equation (3.2) in equation (3.3), and derive the following expression using langrangian relaxation,

$$\mathcal{L} = \frac{1}{2}\left|\left|(\boldsymbol{\beta})^T\left(\mathcal{H}\boldsymbol{\mathcal{M}}\left(\mathcal{H}\right)^T + \lambda\mathbb{I}\right)\,\boldsymbol{\beta}\right|\right|_F^2 + \frac{C}{2}\sum_{i=1}^{\mathcal{N}}\|\mathbf{e}_i\|_2^2 - \sum_{i=1}^{\mathcal{N}}\alpha_i\left((\boldsymbol{\beta})^T\boldsymbol{h}\left(\boldsymbol{x}_i\right) - \boldsymbol{x}_i + \mathbf{e}_i\right), \tag{3.4}$$

where, $\alpha = \{\alpha_i\}$, $i = 1, 2, ..., \mathcal{N}$, is a langrangian multiplier. Next, we perform further computations as follows:

$$\frac{\partial\mathcal{L}}{\partial\boldsymbol{\beta}} = 0 \implies \boldsymbol{\beta} = \alpha\mathcal{H}\left(\mathcal{H}\boldsymbol{\mathcal{M}}\left(\mathcal{H}\right)^T + \lambda\mathbb{I}\right)^{-1}, \tag{3.5}$$

$$\frac{\partial\mathcal{L}}{\partial\mathbf{e}_i} = 0 \implies \mathbf{E} = \frac{\alpha}{C}, \tag{3.6}$$

$$\frac{\partial\mathcal{L}}{\partial\alpha} = 0 \implies \alpha = C\left(\mathbf{X} - (\boldsymbol{\beta})^T\mathcal{H}\right). \tag{3.7}$$

We substitute the equation (3.7) in equation (3.5) to derive the output weight as,

$$\boldsymbol{\beta} = \mathcal{H}\left(\mathcal{H}\left(\mathcal{H}\right)^T + \frac{\mathcal{H}\boldsymbol{\mathcal{M}}\left(\mathcal{H}\right)^T}{C} + \frac{\lambda}{C}\mathbb{I}\right)^{-1}\mathbf{X}. \tag{3.8}$$

The network output is further expressed as,

$$\widehat{\boldsymbol{O}} = \boldsymbol{h(x)\beta}. \tag{3.9}$$

Next, we define a kernel matrix $\boldsymbol{\mathit{\Omega}}$ as,

$$\begin{aligned} &\boldsymbol{\mathit{\Omega}} = \mathcal{H}\mathcal{H}^T \\ &s.t.\;\; \mathit{\Omega}_{j,k} = \boldsymbol{h}(\boldsymbol{x}_j)\boldsymbol{h}(\boldsymbol{x}_k) = K(\boldsymbol{x}_j, \boldsymbol{x}_k),\;\; j,k = 1, ..., \mathcal{N}, \end{aligned} \tag{3.10}$$

where, K is a kernel function. We use the kernel mapping to rewrite the equation

(3.8), and calculate the output weight ($\boldsymbol{\beta}$) as,

$$\boldsymbol{\beta} = \left(\boldsymbol{\Omega} + \frac{\boldsymbol{\mathcal{M}}\boldsymbol{\Omega}}{C} + \frac{\lambda}{C}\mathbb{I}\right)^{-1}\mathbf{X}. \tag{3.11}$$

Finally, using kernel mapping we rewrite the equation (3.9), and calculate the network output for the training data as,

$$\widehat{\boldsymbol{O}} = \begin{bmatrix} K(\boldsymbol{x}, x_1) \\ \vdots \\ K(\boldsymbol{x}, x_{\mathcal{N}}) \end{bmatrix}^T \left(\boldsymbol{\Omega} + \frac{\boldsymbol{\mathcal{M}}\boldsymbol{\Omega}}{C} + \frac{\lambda}{C}\mathbb{I}\right)^{-1}\mathbf{X}. \tag{3.12}$$

Since VAAKRL follows a reconstruction-based approach to OCC, the degree of deviation in reconstruction error is used to classify the samples into the target or outlier class. Since the outlier samples don't follow the target class distribution, it is assumed that they have a high reconstruction error relative to the target class samples. This assumption helps to decide the threshold (θ) as follows,

(1) We calculate the loss $\boldsymbol{s}$ using the following loss function,

$$s_i = \sum_{j=1}^{d}(\widehat{O}_{ij} - x_{ij})^2,\ i = 1, 2, ..., \mathcal{N}. \tag{3.13}$$

(2) We sort the loss vector ($\boldsymbol{s}$) in decreasing order and denote it as $\boldsymbol{s}_{dec}$. A certain fraction of training data is dismissed as outliers to determine the threshold in OCC. The most deviant samples are dismissed as outliers first, as they have the highest reconstruction error. Hence, we calculate the threshold (θ) as,

$$\theta = \boldsymbol{s}_{dec}(\lfloor \delta * \mathcal{N} \rfloor), \tag{3.14}$$

where, $0 \leq \delta \leq 1$ is the fraction of dismissal. δ controls what fraction of the training data is dismissed as outliers.

Assuming, $\mathbf{X} = [20.8\ \ 31.7\ \ 15.6\ \ 14.5]$, $C = 0.5$ and $\lambda = 1$, we present the following illustrative example of the steps involved during training,

(1) Using 3.10, we determine the kernel matrix as $\boldsymbol{\Omega} = \begin{bmatrix} 1 & 0.2 & 0.7 & 0.6 \\ 0.2 & 1 & 0.06 & 0.04 \\ 0.7 & 0.06 & 1 & 0.9 \\ 0.6 & 0.04 & 0.9 & 1 \end{bmatrix}$.

(2) Using 3.11, we determine the output weight as $\boldsymbol{\beta} = \begin{bmatrix} 3.2 & 0.1 & 1.1 & 1.003 \\ -0.2 & 3.2 & -0.6 & -0.6 \\ 1.2 & -0.1 & 3.6 & 1.6 \\ 1.1 & -0.1 & 1.6 & 3.6 \end{bmatrix}$.

(3) Using 3.12, we determine the network output as $\widehat{\boldsymbol{O}} = [4.2 \;\; 10.9 \;\; 2.3 \;\; 2.1]$.

(4) Using 3.14 and taking $\delta = 0.25$, we determine the threshold as $\theta = 106.3$.

During testing, for a test sample $\boldsymbol{x}_t$, we calculate the test output $\widehat{O}_t$ as,

$$\widehat{O}_t = \begin{bmatrix} K(\boldsymbol{x}_t, \boldsymbol{x}_1) \\ \vdots \\ K(\boldsymbol{x}_t, \boldsymbol{x}_{\mathcal{N}}) \end{bmatrix}^T \boldsymbol{\beta}. \tag{3.15}$$

Further, we calculate the test sample loss (s_t) as,

$$s_t = \sum_{j=1}^{d} \left(\widehat{O}_{tj} - x_{tj} \right)^2. \tag{3.16}$$

Finally, we perform OCC using the following decision rule,

$$\begin{aligned} sign(\theta - s_t) = 1, &\quad \boldsymbol{x}_t \textit{ belongs to target class}, \\ -1, &\quad \boldsymbol{x}_t \textit{ belongs to outlier class}. \end{aligned} \tag{3.17}$$

We briefly provide the implementation steps for the proposed method in Algorithm 3.1. Further, in Section 3.2, we present the experimental results for VAAKRL on different datasets and compare them with various exiting state-of-the-art one-class classifiers.

Algorithm 3.1 VAAKRL

```
Given:
Training dataset: $\mathbf{X}$, Regularization parameter: C, Graph regularization parameter: $\lambda$, Fraction of dismissal: $\delta$
Training:
1: Calculate kernel matrix $\boldsymbol{\Omega}$ using 3.10.
2: Calculate output weight $\boldsymbol{\beta}$ using (3.11).
3: Calculate network output $\widehat{\boldsymbol{O}}$ using (3.12).
4: Calculate threshold $\theta$ using (3.14).
Testing:
1: For test sample $\boldsymbol{x}_t$, calculate network output $\widehat{O}_t$ using (3.15).
2: Classify $\boldsymbol{x}_t$ using (3.17).
```

3.2 Experiments

Matlab R2016a is used for all the trials running on a PC with Intel Core i5 3.10 GHz CPU, 32 GB RAM. We have conducted experiments on 14 UCI benchmark datasets. The datasets have been downloaded from the UCI Machine Learning Repository [82], and the website of TU Delft[1] made available by Tax and Duin [83] in the preprocessed form for OCC. Tax and Duin [83] obtained the one-class datasets from the multi-class datasets by taking one of the classes as target and the rest of the classes as outliers. We have followed the same approach. We present the specifications of the one-class datasets, along with the associated target class in Table 3.1. The samples with any missing feature values have been removed. We have normalized all the features with a mean 0 and standard deviation 1 using z-score. We have used 50% of the target and outlier class samples for 5-fold cross-validation, and the other 50% as the test set. It is important to note that we have used samples from only the target class to train the model. The optimal parameters is selected using 5-fold cross-validation from a range of values. The regularization parameter C is selected from the range $\{2^{-5}, 2^{-4}, ..., 2^{5}\}$.

[1] `http://homepage.tudelft.nl/n9d04/occ/`

S.no.	**Datasets**	**#Total Samples**	**#Target**	**#Outlier**	**#Features**	**Target Class**
1	Biomed*	194	127	67	5	Healthy
2	Breast Cancer*	699	241	458	9	Malignant
3	Breast Tissue	106	18	88	9	Mastopathy
4	Caesarian	80	46	34	5	1
5	Cardiotocography	2126	176	1950	22	Pathologic
6	Colposcopy	97	82	15	62	Good
7	Diabetic Retinopathy	1151	540	611	19	Normal
8	Heart Cleveland	297	160	137	13	Absent
9	Hepatitis*	155	123	32	19	Normal
10	Imports*	159	71	88	25	Low Risk
11	Sonar*	208	97	111	60	Rocks
12	SPECT Heart*	349	254	95	44	Abnormal
13	Waveform*	900	300	600	21	1
14	Wine*	178	48	130	13	3

* Obtained from website of TU Delft [83]. Rest are from UCI Machine Learning Repository [82].

Table 3.1: Specification of one-class datasets.

The graph regularization parameter λ is taken as 1 in all the experiments. The number of clusters k for k-means clustering is selected from the range $\{2, ..., 10\}$. The fraction of dismissal of outliers δ is selected from the range $\{1\%, 5\%, 10\%\}$. All the methods employ the Radial Basis Function (RBF) kernel, which can be calculated for data points $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ as follows:

$$\boldsymbol{k(x_i, x_j)} = exp\left(-\frac{\|\boldsymbol{x_i} - \boldsymbol{x_j}\|_2^2}{2\sigma^2}\right), \tag{3.18}$$

where, we have used the mean of the euclidean distance across different training samples to obtain σ. All the existing and proposed one-class classifiers have been implemented and tested in the same environment to ensure a fair comparison.

We compare the performance of the proposed method, VAAKRL, with 14 existing one-class classifiers, namely, One Class Random Forests (OCRF) [84], Principal Component Analysis (PCA) [85], Naive Parzen density estimation [86], k-means [87], 1-Nearest Neighbor (1-NN) [88], k-Nearest Neighbor (k-NN) [89], Autoencoder neural network or Multi-layer Perceptron [5], k-centers [90], Support Vector Data Description (SVDD) [35], One Class Support Vector Machine (OCSVM) [1], Minimum Spanning Tree-based one-class classifier (MST) [91], OCKELM [37], VOCKELM [13], and

AAKELM [48]. The motivation behind choosing the existing one-class classifiers for comparison purposes is based upon the fact that they have been used as benchmark classifiers frequently in the past [84, 91] and are regarded as the standard classifiers in the field of OCC [15]. OCSVM is implemented using the LIBSVM library [92], while the implementation of other existing methods is taken from ddtoolbox [93].

We present the F_1 scores for VAAKRL and the existing one-class classifiers for different datasets in Tables 3.2 and 3.3. Due to the limitation in page width, the results are divided into two tables. The first row in the tables lists the name of classifiers, the first column lists the datasets, and the last row lists the mean F_1 score (η_{F_1}) for each classifier. We consider η_{F_1} as the final evaluation measure to rank the classifiers as per their performance. VAAKRL achieves the highest η_{F_1} over all the datasets (highlighted in bold red in Table 3.3) in comparison to other one-class classifiers, with a significant difference of 6.95% in the case of non-kernel-based methods. Also, it can be noted that VAAKRL obtains the highest F_1 score for an overwhelming 13 out of 14 datasets except Colposcopy, with a few other classifiers obtaining identical values for some datasets. VAAKRL achieves this by reducing the variance and minimizing the reconstruction error using the minimum variance embedded KRL-based autoencoder. The reconstruction property helps to learn essential features from noisy input data, and the minimum variance embedding in VAAKRL helps in better separation of outliers. They grant VAAKRL a boost in performance over other one-class classifiers.

For reference purpose, we also present the experimental results based on accuracy, g-mean, precision, and recall metrics for VAAKRL along with other KRL-based one-class classifiers in Figure 3.2. VAAKRL achieves the highest accuracy for 12 out of 14 datasets. VOCKELM and OCKELM score the highest accuracy for the remaining 2 datasets, namely, Breast Tissue and Colposcopy. In terms of g-mean, VAAKRL scores highest for 12 out of 14 datasets. OCKELM scores the highest g-mean for Colposcopy, while AAKELM scores the highest g-mean for Diabetic Retinopathy. VAAKRL achieves the highest precision for 11 datasets and the highest recall for 7 datasets. The efficiency of VAAKRL is evident from the observation that it performs overwhelmingly better than other methods by scoring the highest accuracy, g-mean,

	OCRF	**Naive**	**k-means**	**1-NN**	**k-NN**	**Autoencoder**	**PCA**	**MST**
	[84]	**Parzen[86]**	**[87]**	**[88]**	**[89]**	**[5]**	**[85]**	**[91]**
Biomed	79.25	93.75	91.18	92.42	89.86	88.37	94.03	91.18
Breast Cancer	51.17	88.33	90.4	46.4	48.91	57.07	50.11	88.12
Breast Tissue	29.03	50	48.48	43.24	48.48	38.71	44.44	48.48
Caesarian	73.02	70.97	73.02	73.02	73.02	64.29	70.97	73.02
Cardiotocography	15.29	39.07	25.81	39.39	34.08	55.51	35.58	34.08
Colposcopy	87.06	81.08	89.66	89.66	90.91	88.37	86.75	90.91
Diabetic Retinopathy	63.91	65.91	65.17	66	65.56	66.84	66.58	66
Heart Cleveland	70.18	76.3	65.19	67.01	69.79	73.68	72.99	68.78
Hepatitis	88.41	86.18	88.41	88.41	88.41	88.89	88.89	88.41
Imports	61.4	63.83	64.08	65.26	63.55	60.87	55.88	61.95
Sonar	66.67	63.72	64	62.86	63.95	67.63	62.02	63.95
SPECT Heart	84.39	79.58	84.39	79.17	84.39	84.39	80.41	84.39
Waveform	50	78.26	73.6	74.41	73.25	70.8	72.33	73.33
Wine	42.48	93.62	62.3	59.74	62.16	59.15	43.14	61.33
η_{F_1}	61.59	73.61	70.41	67.64	68.31	68.9	66.01	71

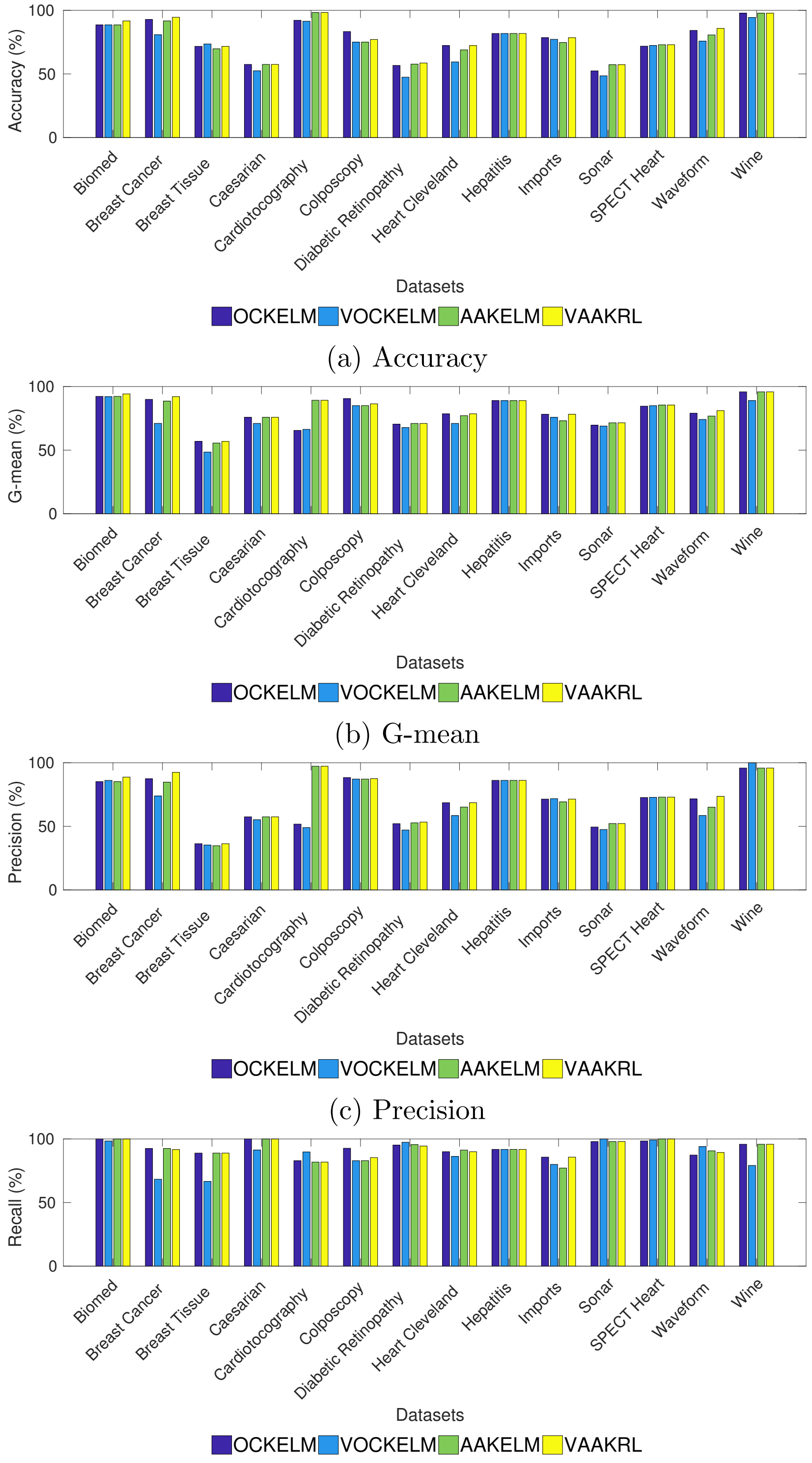
Table 3.2: Performance in terms of F_1 score over 14 one-class datasets.

	k-centers	**Kernel-based methods**					
	[90]	**OCSVM**	**SVDD**	**OCKELM**	**VOCKELM**	**AAKELM**	**VAAKRL**
		[1]	**[35]**	**[37]**	**[13]**	**[48]**	
Biomed	88.57	91.85	90.63	91.97	91.85	91.97	94.03
Breast Cancer	46.09	84.43	81.22	89.88	71	88.45	92.05
Breast Tissue	41.67	50	45.45	51.61	46.15	50	51.61
Caesarian	67.8	62.07	63.16	73.02	68.85	73.02	73.02
Cardiotocography	26.76	17.17	19.15	63.76	63.45	88.89	88.89
Colposcopy	89.66	86.75	82.05	90.48	85	85	86.42
Diabetic Retinopathy	65.2	65.06	65.17	67.37	63.53	67.98	68.18
Heart Cleveland	64.97	66.67	67.33	77.84	69.7	76.04	77.84
Hepatitis	88.41	84.85	86.57	88.89	88.89	88.89	88.89
Imports	68.13	70	55.74	77.92	75.68	72.97	77.92
Sonar	67.65	64.18	63.25	65.73	64.43	68.12	68.12
SPECT Heart	84	81.45	77.27	83.61	84	84.39	84.39
Waveform	71.2	79.14	75.78	78.68	72.12	75.77	80.72
Wine	54.55	59.7	63.64	95.83	88.37	95.83	95.83
η_{F_1}	66.05	68.81	66.89	78.33	73.79	79.09	**80.57**

Table 3.3: Performance in terms of F_1 score over 14 one-class datasets (continued).

and precision for 12,12,11 datasets, respectively.

We present the variation in F_1 score with the regularization parameter (C) and kernel parameter (σ) for the KRL-based methods for the Cardiotocography dataset in Figure 3.3. The difference in the behavior of the boundary-based methods, OCKELM and VOCKELM, and the reconstruction-based methods, AAKELM and VAAKRL, is clearly visible in the figure. In the case of OCKELM and VOCKELM, with an



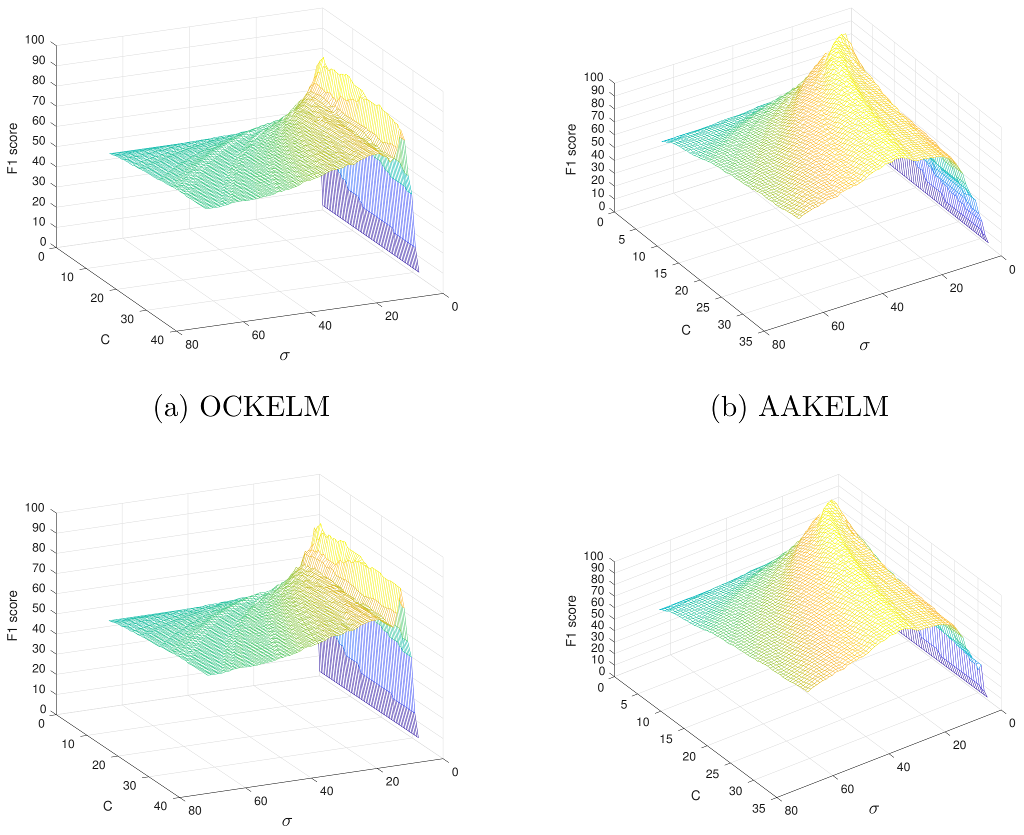
(a) Accuracy

(b) G-mean

(c) Precision

(d) Recall

Figure 3.2: Accuracy, G-mean, Precision and Recall plots for different datasets for VAAKRL and existing KRL-based one-class classifiers.



(a) OCKELM

(b) AAKELM

(c) VOCKELM

(d) VAAKRL

Figure 3.3: Variation of F_1 score with regularization parameter (C) and kernel size (σ) for the Cardiotocography dataset.

increase in the value of σ, the value of the F_1 score becomes increasingly constant. Also, for a constant σ, there is a little variation in the F_1 score for varying value of C. In contrast, for AAKELM and VAAKRL, the F_1 score becomes more and more stable, with an increase in σ and a decrease in C. Also, it can be observed that there is an evident variation in the F_1 score for high values of C and σ.

In OCC, the decision criteria is set by taking a portion of data as outliers (δ) during training time. For reference, we present the variation of F_1 scores of the different KRL-based methods across different values of fraction of dismissal, namely $\delta = 1\%$, 5%, 10%, in Figure 3.4. The observations are noted as follows,

(1) For $\delta = 1\%$, VAAKRL achieves the highest F_1 score for 8 datasets. Also, VAAKRL

displays a clear advantage over other methods for 4 datasets, namely, Biomed, Breast Cancer, Imports, and Waveform.

(2) For $\delta = 5\%$, VAAKRL achieves the highest F_1 score for 10 datasets. VAAKRL performs better than the other methods by a wide margin for 2 datasets, namely, Hepatitis, and Waveform.

(3) For $\delta = 10\%$, VAAKRL achieves the highest F_1 score for 9 datasets and performs better than the other methods by a wide margin for 3 datasets, namely, Diabetic Retinopathy, Imports, and Waveform.

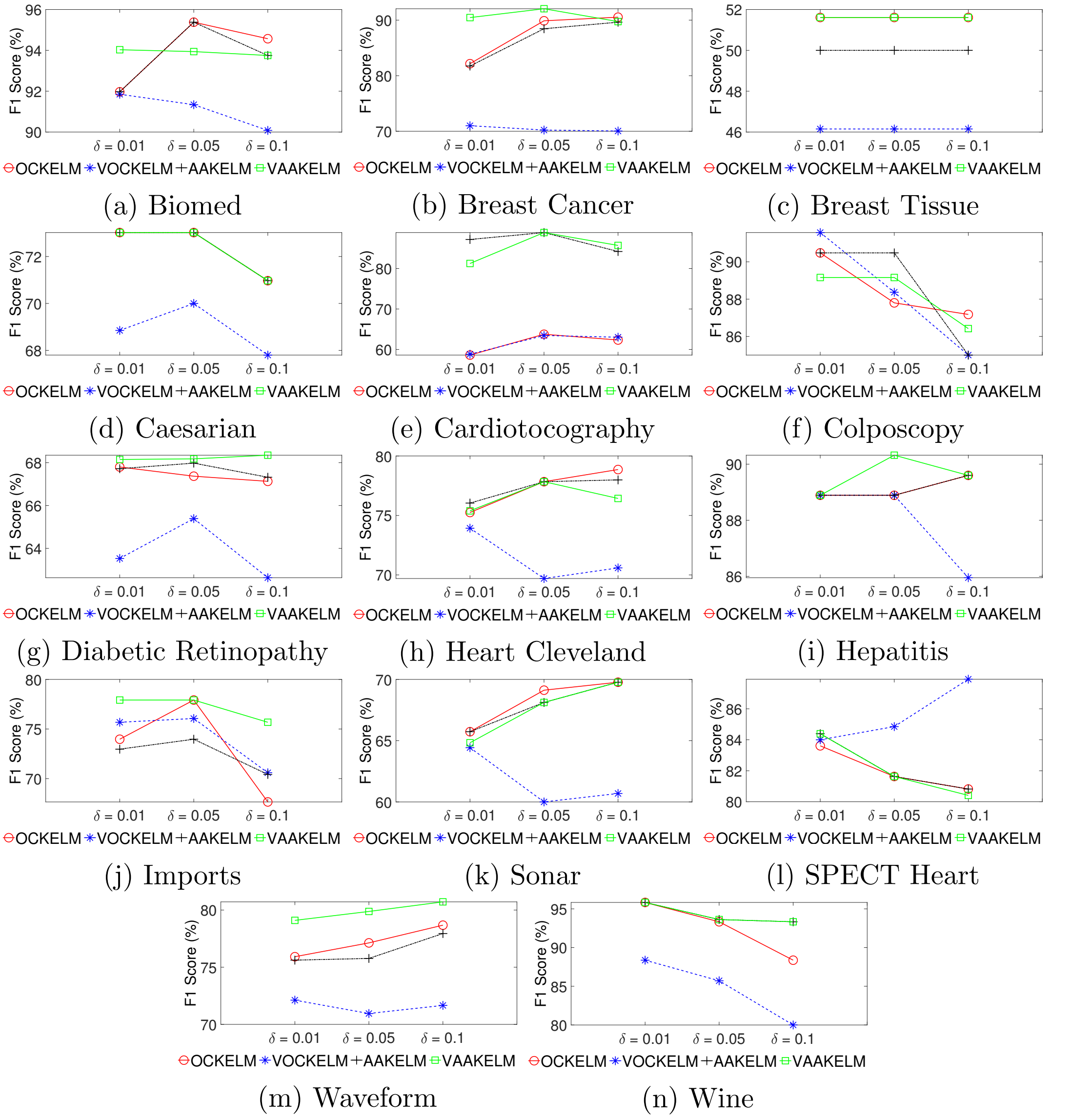
(4) For 7 datasets, VAAKRL scores the highest across all δ values, while showing a clear advantage in performance for 2 datasets, namely, Imports, and Waveform.

From the above observations (1,2,3), it can be inferred that VAAKRL generally outperforms other methods for the same value of δ. Also, using a small value of δ usually gives better results.

When comparing methods, computational complexity is a crucial performance metric. The training time spent on different OCC methods is recorded in Table 3.4. The training times of all the classifiers, except OCSVM, are recorded on the MATLAB platform. As OCSVM has used the Mex C++ compiler and not the same environ-

	Biomed	**Breast Cancer**	**Breast Tissue**	**Caesarian**	**Cardiotocography**	**Colposcopy**	**Diabetic Retinopathy**	**Heart Cleveland**
OCRF [84]	0.1163	0.2582	0.0561	0.0658	0.3968	1.1940	0.9092	0.2087
Naive Parzen [86]	1.9521	0.4376	0.1187	0.0714	0.3172	0.6763	0.8659	0.1581
k-means [87]	0.0885	0.0222	0.0197	0.0155	0.0245	0.0147	0.0244	0.0155
1-NN [88]	0.0517	0.0265	0.0145	0.0135	0.0205	0.0131	0.0270	0.0139
k-NN [89]	0.0241	0.0242	0.0145	0.0132	0.0210	0.0131	0.0247	0.0136
Autoencoder [5]	3.9490	0.5733	0.3077	0.2006	2.7101	0.5027	3.1774	2.2721
PCA [85]	0.1520	0.0708	0.0887	0.0460	0.0735	0.0466	0.0371	0.0361
MST [91]	0.0186	0.0206	0.0145	0.0135	0.0211	0.0135	0.0249	0.0196
k-centers [90]	0.3926	0.2027	0.1885	0.1586	0.2012	0.1698	0.2374	0.1939
SVDD [35]	0.1299	0.0281	0.0206	0.0189	0.0658	0.0121	0.0351	0.0125
OCKELM [37]	0.0202	0.0140	0.0036	0.0027	0.0144	0.0023	0.0204	0.0044
VOCKELM [13]	0.0126	0.0104	0.0032	0.0020	0.0123	0.0023	0.0210	0.0048
AAKELM [48]	0.0119	0.0130	0.0034	0.0023	0.0144	0.0025	0.0233	0.0043
VAAKRL	0.0126	0.0122	0.0031	0.0023	0.0151	0.0025	0.0246	0.0038

Table 3.4: Training time (in secs) for different one-class classifiers.



(a) Biomed (b) Breast Cancer (c) Breast Tissue

(d) Caesarian (e) Cardiotocography (f) Colposcopy

(g) Diabetic Retinopathy (h) Heart Cleveland (i) Hepatitis

(j) Imports (k) Sonar (l) SPECT Heart

(m) Waveform (n) Wine

Figure 3.4: Variation of F_1 score with fraction of dismissal (δ) for different datasets for VAAKRL and existing KRL-based one-class classifiers.

ment as other classifiers, we have not included OCSVM in Table 3.4. In the table, it can be observed that the training time of VAAKRL is similar to OCKLEM, VOCKELM, and AAKELM. This is expected as all four of them are non-iterative in nature. However, VAAKRL mostly records the least training time among all other one-class classifiers (i.e., all classifiers except OCKLEM, VOCKELM, and AAKELM) owing to its non-iterative approach to learning.

3.3 Summary

In this chapter, we proposed the minimum variance embedded KRL-based autoencoder for OCC. It is a single-layer method and follows a reconstruction-based approach to OCC. The minimum variance embedding reduces the variance of the target class data and forces the network output weights to emphasize in regions of low variance. The proposed method uses reconstruction error to define a threshold criterion to decide the membership of a data sample. The KRL-based autoencoder utilizes representation learning to build an effective representation of the data at the output layer. The proposed method consumes less training time in comparison to the existing iterative learning-based one-class classifiers owing to its non-iterative approach to learning. The proposed method was experimented on 14 benchmark datasets from various disciplines and the outcomes were compared with 14 existing one-class classifiers. VAAKRL achieved the highest η_{F_1} in comparison to other one-class classifiers and outperformed the non-kernel-based one-class classifiers by a significant margin of more than 6.9% in terms of η_{F_1}. VAAKRL has yielded a slightly better η_{F_1} value compared to OCKELM (boundary framework-based) and AAKELM (reconstruction framework-based). Therefore, it is very difficult to declare any single framework-based one-class classifier as the best classifier.

We further try to improve the performance of the KRL-based one-class classifier. We combine the concept of the boundary-based and reconstruction-based frameworks in a multi-layer architecture, and incorporate minimum variance embedding at the first layer. We explore this OCC architecture in the next chapter.

Chapter 4

Minimum Variance Embedded Deep Kernel Regularized Least-Squares Method for One-class Classification

To enhance the performance of the proposed one-class classifier in the previous chapter, we develop the Minimum **V**ariance Embedded **D**eep **KRL**-based method for **O**ne-class **C**lassification (DKRLVOC) in this chapter. A multi-layer KRL-based one-class classifier was proposed in the past [38], but it didn't utilize the minimum variance information within its architecture. DKRLVOC leverages the minimum variance embedding to minimize the data dispersion of the target class and force the network output weights to emphasize in areas of low variance. The multi-layer approach helps to combine both the reconstruction-based and boundary-based frameworks in a single architecture, hence we refer the proposed architecture as a deep architecture. The reconstruction-based framework helps to learn an effective representation of the data by reconstructing the key input features at the output. The boundary-based framework defines a one-class boundary around the target class using the structural information of the dataset. We have experimented with the proposed method on 24 benchmark datasets and compared their performance with different state-of-the-art one-class classifiers based on various performance metrics. We discuss the details of the proposed method further in Section 4.1.

4.1 Proposed Method: DKRLVOC

The proposed method, DKRLVOC, follows a deep architecture and minimizes variance at the first layer to achieve better separation of outliers. We present a schematic representation of the proposed architecture in Figure 4.1. It consists of mainly three types of layers viz., (i) minimum **v**ariance-embedded **KRL**-based **A**uto**e**ncoder (KRLVAE) (ii) **KRL**-based **A**uto**e**ncoder (KRLAE) (iii) **KRL**-based **OCC** layer (KRLOC). The layers are stacked sequentially to form a deep architecture. The first layer is the KRLVAE layer and is responsible for minimizing the variance, the norm of output weight, and the reconstruction error. The KRLVAE layer is followed by multiple KRLAE layers, which are used to extract meaningful information from the data. These KRLAE layers are responsible for representation learning and help to learn an effective representation of the data. The last layer is the KRLOC layer, which is a boundary framework-based OCC layer and is responsible for learning a boundary around the target class. We have performed the minimum variance embedding only at the first layer as we observed that minimizing variance in successive layers leads to loss of pattern between the samples, and hence poor generalization performance. Further, we introduce the formulation of the proposed method.

We represent the training data as, $\mathbf{X}^{(1)} = \{\boldsymbol{x}_i^{(1)} | \boldsymbol{x}_i^{(1)} \in \mathbb{R}^d, i = 1, 2, ..., \mathcal{N}\}$, where $\mathcal{N}$ refers to the number of training samples. The first Q layers of DKRLVOC is composed of KRL-based autoencoder. We refer the input of the q^{th} layer as, $\mathbf{X}^{(q)} = \{\boldsymbol{x}_i^{(q)} \,|\, \boldsymbol{x}_i^{(q)} \in \mathbb{R}^d, i = 1, 2, ..., \mathcal{N}, q = 1, 2, ..., Q\}$. Further, we refer the input of the final OCC layer as, $\mathbf{X}^{(f)} = \{\boldsymbol{x}_i^{(f)} \,|\, \boldsymbol{x}_i^{(f)} \in \mathbb{R}^d, i = 1, 2, ..., \mathcal{N}\}$. KRLVAE is the first layer (i.e., $q = 1$), while the subsequent $Q - 1$ layers are KRLAE layers (i.e., $q = 2, ..., Q$). The final OCC layer is the KRLOC layer. The encoded output of one layer is fed as input to the next layer.

We propose the following optimization criterion at the first layer (KRLVAE) to

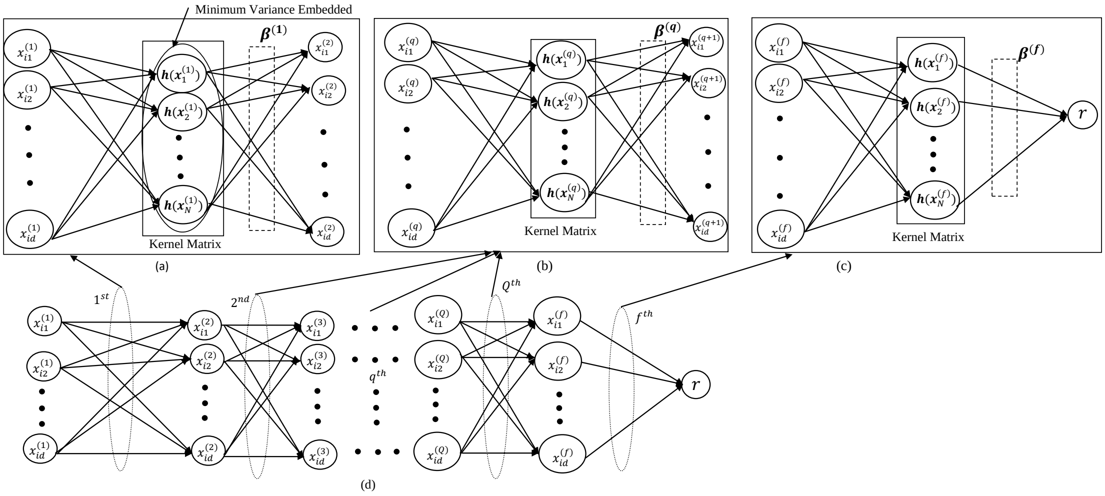

Minimum Variance Embedded
$\boldsymbol{\beta}^{(1)}$
$x_{i1}^{(1)}$
$x_{i2}^{(1)}$
$x_{id}^{(1)}$
$\boldsymbol{h}(\boldsymbol{x}_1^{(1)})$
$\boldsymbol{h}(\boldsymbol{x}_2^{(1)})$
$\boldsymbol{h}(\boldsymbol{x}_N^{(1)})$
$x_{i1}^{(2)}$
$x_{i2}^{(2)}$
$x_{id}^{(2)}$
Kernel Matrix
(a)
$\boldsymbol{\beta}^{(q)}$
$x_{i1}^{(q)}$
$x_{i2}^{(q)}$
$x_{id}^{(q)}$
$\boldsymbol{h}(\boldsymbol{x}_1^{(q)})$
$\boldsymbol{h}(\boldsymbol{x}_2^{(q)})$
$\boldsymbol{h}(\boldsymbol{x}_N^{(q)})$
$x_{i1}^{(q+1)}$
$x_{i2}^{(q+1)}$
$x_{id}^{(q+1)}$
Kernel Matrix
(b)
$\boldsymbol{\beta}^{(f)}$
$x_{i1}^{(f)}$
$x_{i2}^{(f)}$
$x_{id}^{(f)}$
$\boldsymbol{h}(\boldsymbol{x}_1^{(f)})$
$\boldsymbol{h}(\boldsymbol{x}_2^{(f)})$
$\boldsymbol{h}(\boldsymbol{x}_N^{(f)})$
r
Kernel Matrix
(c)
1^{st}
2^{nd}
q^{th}
Q^{th}
f^{th}
$x_{i1}^{(3)}$
$x_{i2}^{(3)}$
$x_{id}^{(3)}$
$x_{i1}^{(Q)}$
$x_{i2}^{(Q)}$
$x_{id}^{(Q)}$
r
(d)

Figure 4.1: Architecture of DKRLVOC. (a) Encoded output of KRLVAE layer is fed as input to next KRLAE layer. (b) Encoded output of each KRLAE is fed as input to the subsequent KRLAE layer. (c) KRLOC layer takes encoded output of the last KRLAE layer as input. (d) Shows arrangement of different layers.

minimize the data dispersion and the reconstruction error.

$$\begin{aligned} &\min_{\boldsymbol{\beta}^{(1)},\mathbf{e}_i^{(1)}} \frac{1}{2}Tr\left(\left(\boldsymbol{\beta}^{(1)}\right)^T(\mathbf{V}_C+\lambda\mathbb{I})\,\boldsymbol{\beta}^{(1)}\right)+\frac{C^{(1)}}{2}\sum_{i=1}^{\mathcal{N}}\left\|\mathbf{e}_i^{(1)}\right\|_2^2 \\ &s.t.\ \left(\boldsymbol{\beta}^{(1)}\right)^T\boldsymbol{h}\left(\boldsymbol{x}_i^{(1)}\right)=\boldsymbol{x}_i^{(1)}-\mathbf{e}_i^{(1)},\ i=1,2,...,\mathcal{N}, \end{aligned} \tag{4.1}$$

where, $\mathbf{e}_i^{(1)}$ is the reconstruction error, and $\boldsymbol{h}\left(\boldsymbol{x}_i^{(1)}\right)$ is the non-linear feature mapping for input $\boldsymbol{x}_i^{(1)}$. $\boldsymbol{\beta}^{(1)}$ is the output weight at the first layer. $C^{(1)}$ acts as a trade-off to minimize the norm of output weight and the reconstruction error. λ is used to control the degree of regularization for variance and referred to as the graph regularization parameter. $\mathbf{V}_C$ is the class variance. We have discussed the mathematical formulation for $\mathbf{V}_C$ in Section 3.1 of Chapter 3. We substitute the expression of $\mathbf{V}_C$ from equation

(3.2) in equation (4.1), and obtain the langrangian relaxation for equation (4.1) as,

$$\begin{aligned}\mathcal{L}_{KRLVAE} =& \frac{1}{2}Tr\left(\left(\boldsymbol{\beta}^{(1)}\right)^T\left(\mathcal{H}^{(1)}\boldsymbol{\mathcal{M}}\left(\mathcal{H}^{(1)}\right)^T+\lambda\mathbb{I}\right)\boldsymbol{\beta}^{(1)}\right)+\frac{C^{(1)}}{2}\sum_{i=1}^{\mathcal{N}}\left\|\mathbf{e}_i^{(1)}\right\|_2^2 \\ &-\sum_{i=1}^{\mathcal{N}}\alpha_i^{(1)}\left(\left(\boldsymbol{\beta}^{(1)}\right)^T\boldsymbol{h}\left(\boldsymbol{x}_i^{(1)}\right)-\boldsymbol{x}_i^{(1)}+\mathbf{e}_i^{(1)}\right),\end{aligned} \tag{4.2}$$

where, $\alpha^{(1)} = \{\alpha_i^{(1)}\}$, $i = 1, 2, ..., \mathcal{N}$ is the langrangian multiplier at the first layer. Next, we perform further computations as follows:

$$\frac{\partial\mathcal{L}_{KRLVAE}}{\partial\boldsymbol{\beta}^{(1)}}=0 \implies \boldsymbol{\beta}^{(1)}=\alpha^{(1)}\mathcal{H}^{(1)}\left(\mathcal{H}^{(1)}\boldsymbol{\mathcal{M}}\left(\mathcal{H}^{(1)}\right)^T+\lambda\mathbb{I}\right)^{-1}, \tag{4.3}$$

$$\frac{\partial\mathcal{L}_{KRLVAE}}{\partial\mathbf{e}_i^{(1)}}=0 \implies \mathbf{E}^{(1)}=\frac{\alpha^{(1)}}{C^{(1)}}, \tag{4.4}$$

$$\frac{\partial\mathcal{L}_{KRLVAE}}{\partial\alpha^{(1)}}=0 \implies \alpha^{(1)}=C^{(1)}\left(\mathbf{X}^{(1)}-\left(\boldsymbol{\beta}^{(1)}\right)^T\mathcal{H}^{(1)}\right). \tag{4.5}$$

Substituting equation (4.5) in equation (4.3) we get the following expression for output weight,

$$\boldsymbol{\beta}^{(1)}=\mathcal{H}^{(1)}\left(\mathcal{H}^{(1)}\left(\mathcal{H}^{(1)}\right)^T+\frac{\mathcal{H}^{(1)}\boldsymbol{\mathcal{M}}\left(\mathcal{H}^{(1)}\right)^T}{C^{(1)}}+\frac{\lambda}{C^{(1)}}\mathbb{I}\right)^{-1}\mathbf{X}^{(1)}. \tag{4.6}$$

Further, we define a kernel matrix $\boldsymbol{\Omega}^{(1)}$ as,

$$\begin{aligned}&\boldsymbol{\Omega}^{(1)}=\mathcal{H}^{(1)}\left(\mathcal{H}^{(1)}\right)^T \\ &s.t.\ \Omega_{j,k}^{(1)}=\boldsymbol{h}\left(\boldsymbol{x}_j^{(1)}\right)\boldsymbol{h}\left(\boldsymbol{x}_k^{(1)}\right)=K\left(\boldsymbol{x}_j^{(1)},\boldsymbol{x}_k^{(1)}\right),\ j,k=1,...,\mathcal{N},\end{aligned} \tag{4.7}$$

where, K is a kernel function. Using this kernel mapping, we can rewrite equation (4.6) to obtain the final expression for output weight $\boldsymbol{\beta}^{(1)}$ as,

$$\boldsymbol{\beta}^{(1)}=\left(\boldsymbol{\Omega}^{(1)}+\frac{\boldsymbol{\mathcal{M}}\boldsymbol{\Omega}^{(1)}}{C^{(1)}}+\frac{\lambda}{C^{(1)}}\mathbb{I}\right)^{-1}\mathbf{X}^{(1)}. \tag{4.8}$$

In DKRLVOC, the encoded output of one layer is fed as input to the succeeding layer.

Hence, the input to the second (i.e., KRLAE) layer is calculated as,

$$\mathbf{X}^{(2)} = \begin{bmatrix} K(\boldsymbol{x}^{(1)}, \boldsymbol{x_1}) \\ \vdots \\ K(\boldsymbol{x}^{(1)}, \boldsymbol{x_{\mathcal{N}}}) \end{bmatrix}^T \left(\boldsymbol{\Omega}^{(1)} + \frac{\boldsymbol{\mathcal{M}} \boldsymbol{\Omega}^{(1)}}{C^{(1)}} + \frac{\lambda}{C^{(1)}} \mathbb{I} \right)^{-1} \mathbf{X}^{(1)}. \tag{4.9}$$

After that, we use $(Q-1)$ KRLAE layers to learn meaningful information from the data. We obtain the optimum output weight $\boldsymbol{\beta}^{(q)}$ by using the following optimization criterion,

$$\begin{aligned} &\min_{\boldsymbol{\beta}^{(q)}, \mathbf{e}_i^{(q)}} \ \frac{1}{2} \left\| \boldsymbol{\beta}^{(q)} \right\|_F^2 + \frac{C^{(q)}}{2} \sum_{i=1}^{\mathcal{N}} \left\| \mathbf{e}_i^{(q)} \right\|_2^2 \\ &s.t. \ \left(\boldsymbol{\beta}^{(q)} \right)^T \boldsymbol{h} \left(\boldsymbol{x}_i^{(q)} \right) = \boldsymbol{x}_i^{(q)} - \mathbf{e}_i^{(q)}, \ i = 1, 2, ..., \mathcal{N}, \ q = 2, 3, ..., Q, \end{aligned} \tag{4.10}$$

where, $C^{(q)}$ acts as the regularization parameter at the q^{th} layer, and $\mathbf{e}_i^{(q)}$ is the reconstruction error for the input $\boldsymbol{x}_i^{(q)}$. Further, we solve equation (4.10) using langrangian relaxation as follows,

$$\mathcal{L}_{KRLAE} = \frac{1}{2} \left\| \boldsymbol{\beta}^{(q)} \right\|_F^2 + \frac{C^{(q)}}{2} \sum_{i=1}^{\mathcal{N}} \left\| \mathbf{e}_i^{(q)} \right\|_2^2 - \sum_{i=1}^{\mathcal{N}} \alpha_i^{(q)} \left(\left(\boldsymbol{\beta}^{(q)} \right)^T \boldsymbol{h} \left(\boldsymbol{x}_i^{(q)} \right) - \boldsymbol{x}_i^{(q)} + \mathbf{e}_i^{(q)} \right), \tag{4.11}$$

where, $\alpha^{(q)} = \{\alpha_i^{(q)}\}$, $i = 1, 2, ..., \mathcal{N}$ is the langrangian multiplier at the q^{th} layer. Next, we obtain the following derivatives,

$$\frac{\partial \mathcal{L}_{KRLAE}}{\partial \boldsymbol{\beta}^{(q)}} = 0 \implies \boldsymbol{\beta}^{(q)} = \alpha^{(q)} \mathcal{H}^{(q)}, \tag{4.12}$$

$$\frac{\partial \mathcal{L}_{KRLAE}}{\partial \mathbf{e}_i^{(q)}} = 0 \implies \mathbf{E}^{(q)} = \frac{\alpha^{(q)}}{C^{(q)}}, \tag{4.13}$$

$$\frac{\partial \mathcal{L}_{KRLAE}}{\partial \alpha^{(q)}} = 0 \implies \alpha^{(q)} = C^{(q)} \left(\mathbf{X}^{(q)} - \left(\boldsymbol{\beta}^{(q)} \right)^T \mathcal{H}^{(q)} \right). \tag{4.14}$$

We obtain the expression for output weight $\boldsymbol{\beta}^{(q)}$ by substituting equation (4.14) in equation (4.12),

$$\boldsymbol{\beta}^{(q)} = \mathcal{H}^{(q)} \left(\frac{1}{C^{(q)}} \mathbb{I} + \mathcal{H}^{(q)} \left(\mathcal{H}^{(q)} \right)^T \right)^{-1} \mathbf{X}^{(q)}. \tag{4.15}$$

Using kernelized feature mapping similar to equation (4.7), we rewrite equation (4.15) to obtain the final expression for output weight $\boldsymbol{\beta}^{(q)}$ as,

$$\boldsymbol{\beta}^{(q)} = \left(\frac{1}{C^{(q)}}\mathbb{I} + \boldsymbol{\Omega}^{(q)}\right)^{-1}\mathbf{X}^{(q)}. \tag{4.16}$$

The input to the $(q+1)^{th}$ layer is expressed as,

$$\mathbf{X}^{(q+1)} = \begin{bmatrix} K(\boldsymbol{x}^{(q)}, \boldsymbol{x_1}) \\ \vdots \\ K(\boldsymbol{x}^{(q)}, \boldsymbol{x_{\mathcal{N}}}) \end{bmatrix}^T \left(\frac{1}{C^{(q)}}\mathbb{I} + \boldsymbol{\Omega}^{(q)}\right)^{-1}\mathbf{X}^{(q)}, \;\; q = 2, 3, ..., Q. \tag{4.17}$$

Here, $\mathbf{X}^{(Q+1)}$ is used to refer $\mathbf{X}^{(f)}$, which is the input to the final layer (KRLOC). At the final layer, the output weight $\boldsymbol{\beta}^{(f)}$ is derived using the following optimization problem,

$$\begin{aligned} &\min_{\boldsymbol{\beta}^{(f)}, e_i^{(f)}} \; \frac{1}{2}\left\|\boldsymbol{\beta}^{(f)}\right\|_2^2 + \frac{C^{(f)}}{2}\sum_{i=1}^{\mathcal{N}}\left\|e_i^{(f)}\right\|_2^2 \\ &s.t. \;\; \left(\boldsymbol{\beta}^{(f)}\right)^T \boldsymbol{h}\left(\boldsymbol{x}_i^{(f)}\right) = r - e_i^{(f)}, \; i = 1, 2, ..., \mathcal{N}, \end{aligned} \tag{4.18}$$

where, $e_i^{(f)}$ is the training error for input $\boldsymbol{x}_i^{(f)}$. r is a real number, referred to as the target class. It is generally taken as 1. Solving the above minimization problem in a similar way as equation (4.10), the output weight $\boldsymbol{\beta}^{(f)}$ is derived as,

$$\boldsymbol{\beta}^{(f)} = \left(\frac{1}{C^{(f)}}\mathbb{I} + \boldsymbol{\Omega}^{(f)}\right)^{-1}\mathbf{r}, \tag{4.19}$$

where, $\mathbf{r} = [r, ..., r]^T \in R^{\mathcal{N}}$. Finally, the network output of DKRLVOC at the time of training is calculated as,

$$\widehat{\boldsymbol{O}} = \begin{bmatrix} K(\boldsymbol{x}^{(f)}, \boldsymbol{x_1}) \\ \vdots \\ K(\boldsymbol{x}^{(f)}, \boldsymbol{x_{\mathcal{N}}}) \end{bmatrix}^T \left(\frac{1}{C^{(f)}}\mathbb{I} + \boldsymbol{\Omega}^{(f)}\right)^{-1}\mathbf{r}. \tag{4.20}$$

The training samples are used to determine the threshold (θ) as follows,

(1) We calculate the distance between the network output $\widehat{O}_i$ and the target label r for each training sample $\boldsymbol{x}_i$ as,

$$s(i) = \left|\widehat{O}_i - r\right|. \tag{4.21}$$

(2) Further, we sort the vector $\boldsymbol{s}$ in decreasing order and denote it as $\boldsymbol{s_{dec}}$. We dismiss a small percentage of training data as outliers. The samples having maximum distance from the target class are treated as outliers first, as they deviate most from the target class distribution. The threshold is then decided as,

$$\theta = \boldsymbol{s_{dec}}\left(\lfloor \delta \, * \, \mathcal{N} \rfloor\right), \; 0 \leq \delta \leq 1, \tag{4.22}$$

where, δ is the fraction of dismissal.

For each test sample $\boldsymbol{x}_t$, we calculate the input $\boldsymbol{x}_t^{(q+1)}$ for the subsequent layers as follows,

$$\boldsymbol{x}_t^{(q+1)} = \begin{bmatrix} K(\boldsymbol{x}_t^{(q)}, \boldsymbol{x_1}) \\ \vdots \\ K(\boldsymbol{x}_t^{(q)}, \boldsymbol{x_{\mathcal{N}}}) \end{bmatrix}^T \boldsymbol{\beta}^{(q)}, \;\; q = 1, 2, ..., Q. \tag{4.23}$$

Here, $\boldsymbol{x}_t^{(Q+1)}$ is used to refer $\boldsymbol{x}_t^{(f)}$, which is the test input to the final layer. Further, the test network output is calculated as,

$$\widehat{\boldsymbol{O}}_t = \begin{bmatrix} K(\boldsymbol{x}_t^{(f)}, \boldsymbol{x_1}) \\ \vdots \\ K(\boldsymbol{x}_t^{(f)}, \boldsymbol{x_{\mathcal{N}}}) \end{bmatrix}^T \boldsymbol{\beta}^{(f)}. \tag{4.24}$$

Finally, we calculate the distance of the test network output from the target class as,

Algorithm 4.1 DKRLVOC

```
Given:
Training dataset: $\mathbf{X}^{(1)}$, Number of KRL Autoencoder layers: Q, Regularization parameter: $C^{(q)}$ for layer $q = 1, ..., Q$ and $C^{(f)}$ for final layer, Graph regularization parameter: $\lambda$, Fraction of dismissal: $\delta$
Training:
for $q = 1, 2, ..., Q$ layers do
    if $q == 1$ then
        Calculate kernel matrix $\boldsymbol{\Omega}^{(1)}$ using 4.7 and output weight $\boldsymbol{\beta}^{(1)}$ using (4.8).
        Calculate input, $\mathbf{X}^{(2)}$, for the second layer using (4.9).
    else
        Calculate kernel matrix $\boldsymbol{\Omega}^{(q)}$, followed by output weight $\boldsymbol{\beta}^{(q)}$ using (4.16).
        Calculate input, $\mathbf{X}^{(q+1)}$, using (4.17).    ▷ $\mathbf{X}^{(Q+1)}$ refers $\mathbf{X}^{(f)}$
    end if
end for
Calculate kernel matrix $\boldsymbol{\Omega}^{(f)}$, followed by output weight $\boldsymbol{\beta}^{(f)}$ using (4.19).
Calculate network output $\widehat{\boldsymbol{O}}$ using (4.20).
Calculate threshold $\theta$ using (4.22).
Testing:
for $q = 1, 2, ..., Q$ layers do
    Calculate test input, $\boldsymbol{x}_t^{(q+1)}$, using (4.23).    ▷ $\boldsymbol{x}_t^{(Q+1)}$ refers $\boldsymbol{x}_t^{(f)}$
end for
Calculate network output $\widehat{\boldsymbol{O}}_t$ using (4.24).
Calculate distance $s_t$ and classify $\boldsymbol{x}_t$ using (4.25).
```

$s_t = \left| \widehat{\boldsymbol{O}}_t - \mathbf{r} \right|$, and perform OCC based on the following decision function,

$$\begin{aligned} sign(\theta - s_t) = 1, &\quad \boldsymbol{x}_t \, belongs \, to \, target \, class, \\ -1, &\quad \boldsymbol{x}_t \, belongs \, to \, outlier \, class. \end{aligned} \tag{4.25}$$

We provide the implementation steps for the proposed method in Algorithm 4.1. Further, in Section 4.2, we discuss the experimental results for DKRLVOC on different datasets and compare the results with various exiting state-of-the-art one-class classifiers.

4.2 Experiments

Matlab R2016a is used for all the trials running on a PC with Intel Core i5 3.10 GHz CPU, 32 GB RAM. In the experiments, we have used a three-layered ($Q = 2$) DKRLVOC architecture, that is, the initial KRLVAE layer, the intermediate KRLAE layer, and the final KRLOC layer. We have limited the number of layers to three, as we found that with an increase in the number of layers, there is a loss of pattern between the samples. This leads to poor generalization performance. In the experiments, we have taken the value of the graph regularization parameter (λ) as 1. We have used 5-fold cross-validation to select the optimal parameters from a range of values as outlined further. The regularization parameters $\mathrm{C}^{(q)}$ and $\mathrm{C}^{(f)}$ is selected from the range $\{2^{-5}, 2^{-4}, ..., 2^{5}\}$. The value of k for k-means clustering is selected from the range $\{1, 2, ..., 10\}$. k-means clustering is used to group data into sub-classes. The value of the fraction of dismissal (δ) is selected from the range $\{0.01, 0.05, 0.1\}$. All the methods employ the RBF kernel, which can be calculated for data points $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ as follows:

$$\boldsymbol{k(x_i, x_j)} = exp\left(-\frac{\|\boldsymbol{x_i} - \boldsymbol{x_j}\|_2^2}{2\sigma^2}\right), \tag{4.26}$$

where, we use the mean of the euclidean distance across different training samples to obtain σ. All the existing and proposed one-class classifiers have been implemented and tested in the same environment to ensure a fair comparison. The source code of DKRLVOC is available at Github.[1].

We have conducted experiments on 24 UCI benchmark one-class datasets, among which there are 14 small-size datasets and 10 medium-size datasets. The datasets have been downloaded from the UCI Machine Learning Repository [82]. The medium-size one-class datasets have been obtained from the multi-class optical digit dataset, comprised of 10 classes. We have followed an existing approach [83] to convert the multi-class dataset into one-class datasets. We have done this by taking one of the classes as target and the rest of the classes as outliers in an iterative manner. In this way, we were able to generate 10 medium-size one-class datasets. The samples with

[1] `https://github.com/PratikMishra/Deep-Kernel-Learning-for-One-class-Classification`

S.no.	**Datasets**	**#Total Samples**	**#Target**	**#Outlier**	**#Features**	**Target Class**
1	Arrhythmia	420	183	237	278	Abnormal
2	Biomed	194	67	127	5	Diseased
3	Breast Cancer[1]	699	458	241	9	Benign
4	Caesarian	80	34	46	5	0
5	Cancer[2]	198	151	47	33	Non Recurring
6	Cardiotocography	2126	176	1950	22	Pathologic
7	Colposcopy[3]	97	82	15	62	Good
8	Cryotherapy	90	48	42	6	1
9	Hepatitis	155	123	32	19	Normal
10	SPECT Heart	349	254	95	44	Abnormal
11	Survival	306	225	81	3	Greater than 5 year
12	Glass Building	214	76	138	9	Non float
13	Ionosphere	351	126	225	34	Bad
14	Iris	150	50	100	4	Setosa

[1] Refers to Wisconsin Breast Cancer UCI dataset.

[2] Refers to Wisconsin Prognostic Breast Cancer UCI dataset.

[3] Colposcopy dataset with modality hinselmann is used for experimental purpose.

Table 4.1: Specification of small-size one-class datasets.

any missing feature values have been removed. We have normalized all the features with a mean 0 and standard deviation 1 using z-score. We have used 50% of the target and outlier class samples for 5-fold cross-validation and the other 50% as the test set. It is important to note that we have used samples from only the target class to train the model.

Further, we have divided this section into two parts on the basis of the size of datasets used for the experiments. In Section 4.2.1 and 4.2.2, we provide experimental results and performance evaluations on small-size and medium-size datasets, respectively.

4.2.1 Small-size datasets

We have conducted experiments on 14 small-size UCI benchmark one-class datasets. The specifications of these datasets is provided in Table 4.1. We have compared the performance of DKRLVOC with 14 existing state-of-the-art one-class classifiers over 14 datasets in Tables 4.2 and 4.3. Due to the limitation in page width, the results are divided into two tables. The upper rows in the tables list the name of classifiers, the first column lists the datasets, and the last row lists the mean F_1

	OCRF [84]	Naive Parzen[86]	k-means [87]	1-NN [88]	k-NN [89]	Autoencoder [5]	PCA [85]	MST [91]	k-centers [90]
Arrhythmia	60.67	60.67	60.67	58.98	60.67	58.5	57.44	60.67	59.73
Biomed	51.16	45.53	50	48.82	50	38.26	48.54	50	44.04
Breast Cancer	79.24	90.95	94.98	92.99	95.69	95.48	93.51	95.69	95.28
Caesarian	59.65	59.46	53.66	51.16	51.16	53.66	60.38	51.16	50
Cancer	82.42	73.83	86.39	82.58	86.39	79.49	79.75	86.39	84.34
Cardiotocography	15.29	39.07	25.81	39.39	34.08	55.51	35.58	34.08	26.76
Colposcopy	87.06	81.08	89.66	89.66	90.91	88.37	86.75	90.91	89.66
Cryotherapy	69.57	77.55	69.39	73.68	71.64	71.11	77.78	80	74.07
Hepatitis	88.41	86.18	88.41	88.41	88.41	88.89	88.89	88.41	88.41
SPECT Heart	84.39	79.58	84.39	79.17	84.39	84.39	80.41	84.39	84
Survival	85.17	83.33	84.94	83.74	83.87	83.79	83.27	82.95	81.36
Glass Building	52.41	51.49	53.1	55.36	52.34	58.41	55.93	52.34	55.1
Ionosphere	56.11	62.34	51.69	51.69	52.32	52.94	41.62	52.32	52.32
Iris	50	78.05	93.62	91.3	95.83	93.62	80.95	97.96	88.89
η_{F_1}	65.83	69.22	70.48	70.5	71.26	71.6	69.34	71.95	69.57

Table 4.2: Performance in terms of F_1 score over 14 small-size datasets.

	OCSVM [1]	SVDD [35]	KRL-based methods			
			Single-layer		Multi-layer	
			OCKELM [37]	VOCKELM [13]	ML-OCKELM [38]	DKRLVOC
Arrhythmia	57.04	56.12	59.46	58.95	60.67	60.67
Biomed	47.79	48.21	47.37	53.06	47.71	53.45
Breast Cancer	93.85	93.74	95.28	76.68	95.26	95.96
Caesarian	62.86	55.17	53.66	61.11	59.65	62.96
Cancer	84.66	83.02	84.02	86.05	84.02	86.71
Cardiotocography	17.17	19.15	63.76	63.45	69.79	70.53
Colposcopy	86.75	82.05	90.48	85	89.66	92.13
Cryotherapy	75.47	75.47	77.78	75.56	78.43	80
Hepatitis	84.85	86.57	88.89	88.89	87.22	89.39
SPECT Heart	81.45	77.27	83.61	84	84	84.39
Survival	79.01	80.17	83.72	81.57	84.25	85.49
Glass Building	58.59	59.18	56.25	58.33	54.55	62.5
Ionosphere	46.7	42.53	52.94	53.45	60.42	66.29
Iris	68.42	64.86	83.72	93.62	100	100
η_{F_1}	67.47	65.97	72.92	72.84	75.4	**77.89**

Table 4.3: Performance in terms of F_1 score over 14 small-size datasets (continued).

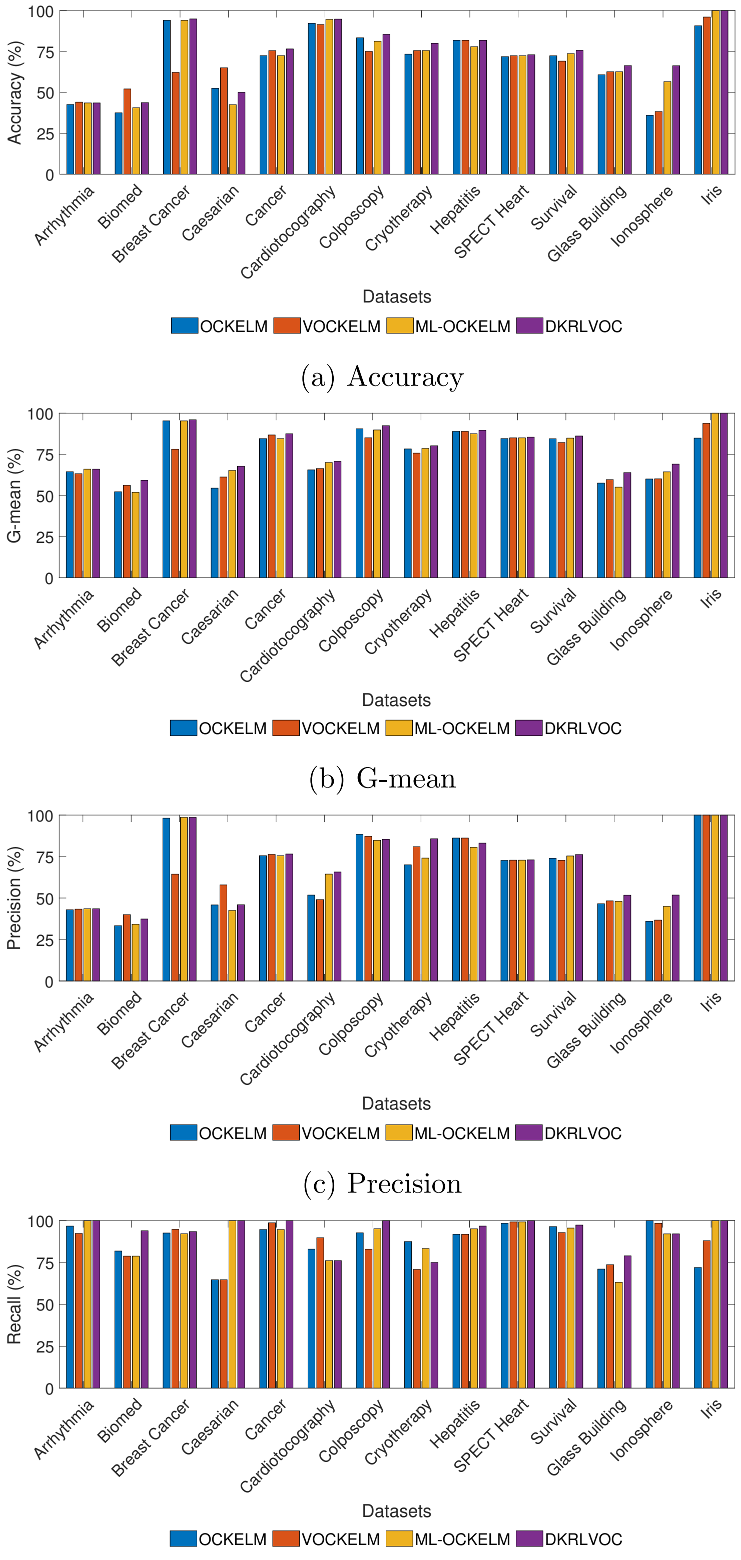
score (η_{F_1}) for each classifier. Since we have to compare multiple classifiers on various datasets, we compute mean of all F_1 scores (η_{F_1}) over all datasets by taking inspiration from an existing work [81]. We consider η_{F_1} as the final evaluation measure to rank the classifiers as per their performance. DKRLVOC achieves the highest η_{F_1} over all the datasets (highlighted in bold red in Table 4.3) in comparison to other one-class classifiers, with a significant difference of 5.94% when compared with non-KRL-based methods. When compared to single-layer KRL-based methods, there is an improvement of 4.97% in terms of η_{F_1}. Also, it can be noted that DKRLVOC obtains the highest F_1 score for all 14 datasets, with a few classifiers obtaining identical values for some datasets. DKRLVOC achieves this by reducing the variance and minimizing the reconstruction error using multiple KRL-based autoencoders stacked sequentially. The minimum variance embedding helps to minimize the data dispersion, and the reconstruction property helps to learn the essential features from the input data. They grant DKRLVOC a boost in performance over other one-class classifiers.

For reference purpose, we also present the experimental results based on accuracy, g-mean, precision, and recall metrics for DKRLVOC along with other KRL-based methods in Figure 4.2. DKRLVOC achieves the highest accuracy for 11 out of 14 datasets. Further, DKRLVOC scored highest g-mean, precision, and recall for 14, 10, and 10 datasets, respectively, as compared to other single-layer and multi-layer KRL-based classifiers. The efficiency of DKRLVOC is evident from the observation that it performs overwhelmingly better than the other methods by scoring the highest accuracy, g-mean, precision, and recall for 11, 14, 10, and 10 datasets, respectively. In OCC, the decision criteria is set by taking a portion of data as outliers (δ) during training time. For reference, we present the variation of F_1 scores of the different KRL-based methods across different values of fraction of dismissal, namely $\delta = \{1\%, 5\%, 10\%\}$, in Figure 4.3. It can be observed that DKRLVOC performs better than the other methods mostly for $\delta = 1\%$. Hence, DKRLVOC gives a better performance for small values of δ.

Further, in Section 4.2.2, we present the performance evaluation of DKRLVOC for medium-size datasets.

4.2.2 Medium-size datasets

We have conducted experiments on 10 medium-size one-class datasets. The datasets are obtained from the multi-class optical digit dataset using the method described in Section 4.2. The specifications of these one-class datasets is provided in Table 4.4.



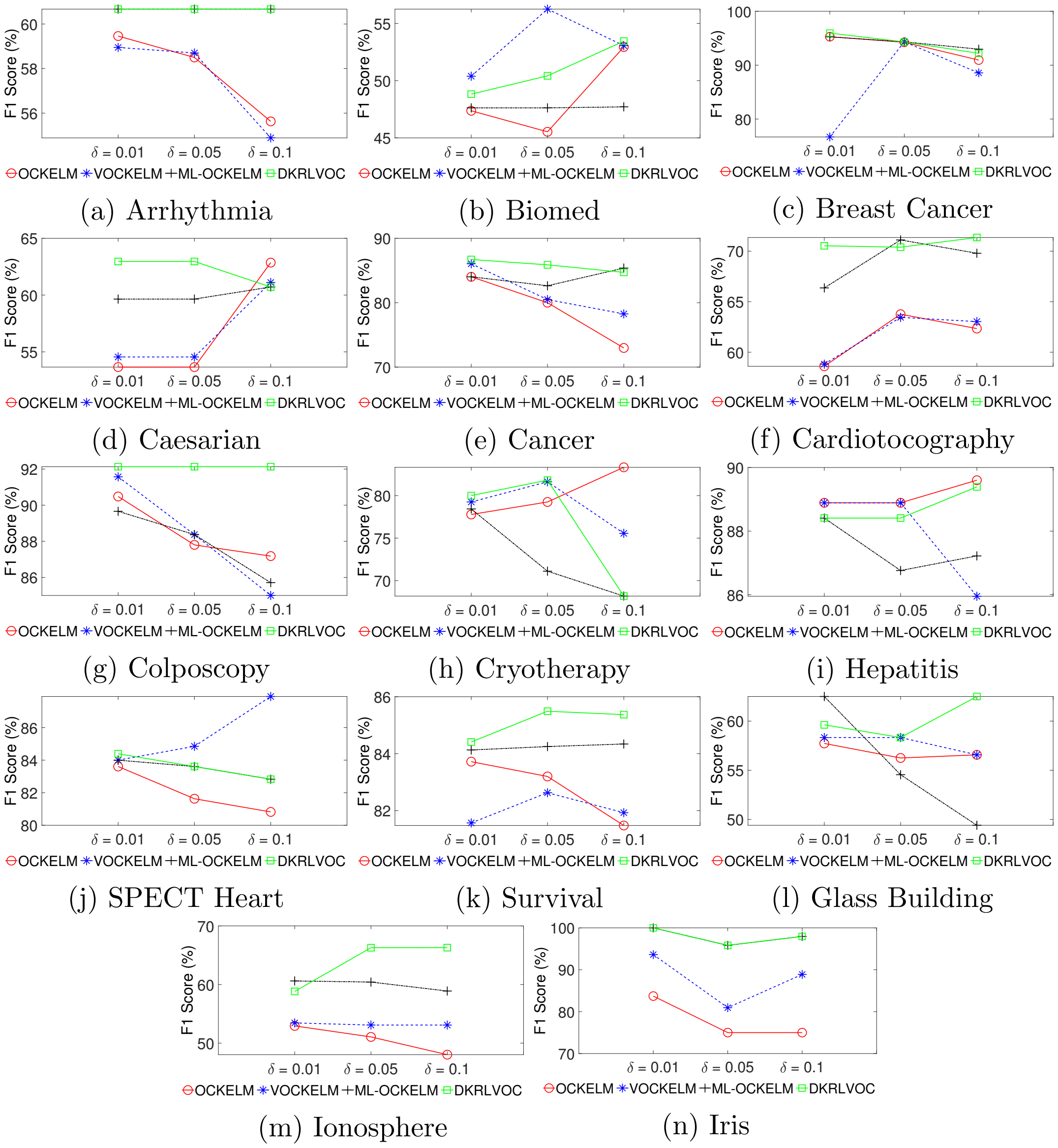
(a) Accuracy

(b) G-mean

(c) Precision

(d) Recall

Figure 4.2: Accuracy, G-mean, Precision and Recall plots for DKRLVOC and existing KRL-based one-class classifiers for small-size datasets.



(a) Arrhythmia

(b) Biomed

(c) Breast Cancer

(d) Caesarian

(e) Cancer

(f) Cardiotocography

(g) Colposcopy

(h) Cryotherapy

(i) Hepatitis

(j) SPECT Heart

(k) Survival

(l) Glass Building

(m) Ionosphere

(n) Iris

Figure 4.3: Variation of F_1 score with fraction of dismissal (δ) for DKRLVOC and existing KRL-based one-class classifiers for small-size datasets.

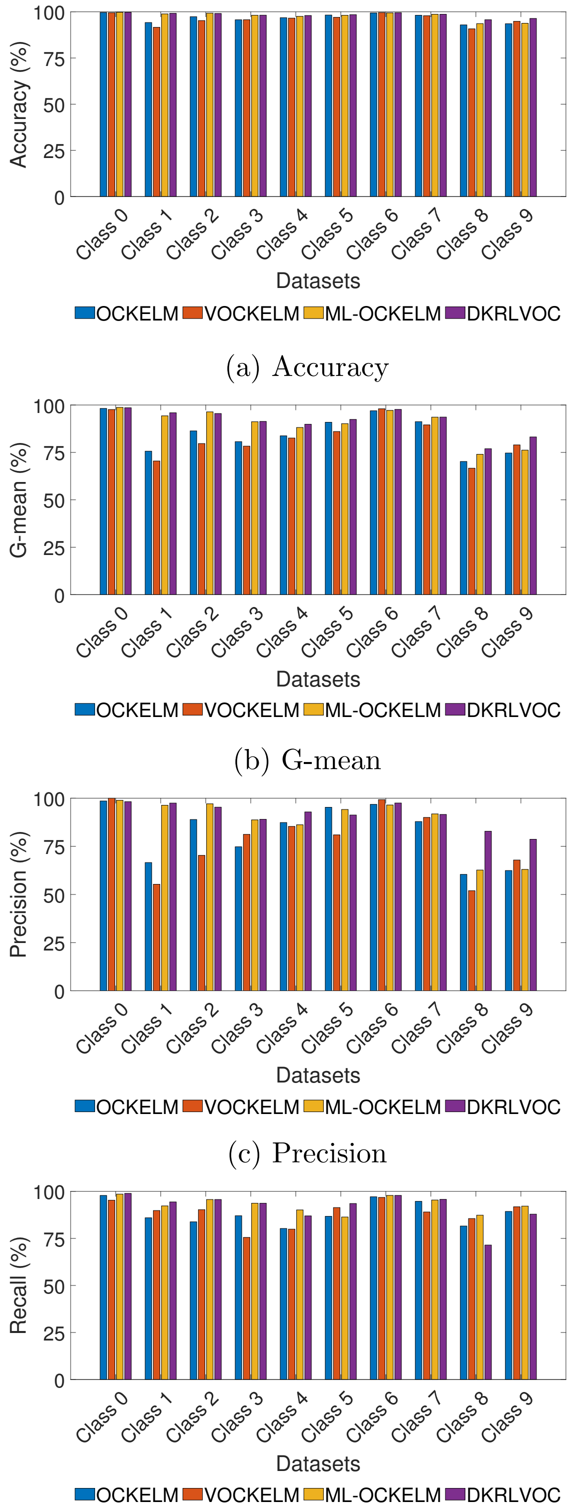
The F_1 scores for DKRLVOC and the existing KRL-based one-class classifiers for optical digit one-class datasets are provided in Table 4.5. For reasons of computational limitation for medium-size datasets, we present F_1 scores for KRL-based methods only. The last row lists the mean F_1 score (η_{F_1}) for each classifier. DKRLVOC achieves the highest η_{F_1} over all the datasets (highlighted in bold red in Table 4.5) in comparison to other KRL-based one-class classifiers, with a significant difference of 6.9% when compared to single-layer KRL-based methods. Also, it can be noted that DKRLVOC obtains the highest F_1 score for 7 out of 10 datasets (highlighted in blue in Table 4.5). For reference, we also present the experimental results based on accuracy, g-mean, precision, and recall metrics for DKRLVOC along with other KRL-based methods for optical digit datasets in Figure 4.4. DKRLVOC achieves the highest accuracy for 7 out of 10 datasets. Further, DKRLVOC scores the highest g-mean, precision, and recall for 7, 5, and 7 datasets, respectively, as compared to the other single-layer and multi-layer KRL-based classifiers. The efficiency of DKRLVOC is evident from the observation that it performs better than the other methods by scoring the highest accuracy, g-mean, and recall for 7, 7, and 7 datasets, respectively. In OCC, the decision criteria is set by taking a portion of data as outliers (δ) during training time. For reference, we present the variation of F_1 scores of the different KRL-based methods for optical digit datasets across different values of fraction of dismissal, namely $\delta = \{1\%, 5\%, 10\%\}$, in Figure 4.5. It can be observed that mostly for $\delta = 1\%$, DKRLVOC performs better than the other methods. Further, it can be observed that for 7 out of 10 cases, the curve of DKRLVOC decreases with an increase in the value of δ. The above observations suggest the improved performance of the method for low values of δ. Also, the multi-layer methods generally perform better than the single-layer methods, as the sequentially stacked reconstruction-based layers in multi-layer methods help to identify the essential information in the data.

Target Class	**#Target**	**#Outlier**	**#Features**
Class 0	554	5066	64
Class 1	571	5049	64
Class 2	557	5063	64
Class 3	572	5048	64
Class 4	568	5052	64
Class 5	558	5062	64
Class 6	558	5062	64
Class 7	566	5054	64
Class 8	554	5066	64
Class 9	562	5058	64

Table 4.4: Specification of medium-size optical digit one-class datasets.

	Single-layer methods		**Multi-layer methods**	
	OCKELM [37]	**VOCKELM [13]**	**ML-OCKELM [38]**	**DKRLVOC**
Class 0	98.19	97.6	98.73	98.56
Class 1	75.04	68.45	94.27	95.9
Class 2	86.3	79.06	96.38	95.51
Class 3	80.45	78.26	91.16	91.31
Class 4	83.67	82.55	88.12	89.82
Class 5	90.81	85.86	90.09	92.39
Class 6	96.96	98	97.15	97.67
Class 7	91.16	89.52	93.59	93.61
Class 8	69.43	64.67	73	76.74
Class 9	73.5	78.06	74.86	83.03
η_{F_1}	84.55	82.2	89.74	**91.45**

Table 4.5: Performance in terms of F_1 score over 10 medium-size optical digit one-class datasets.



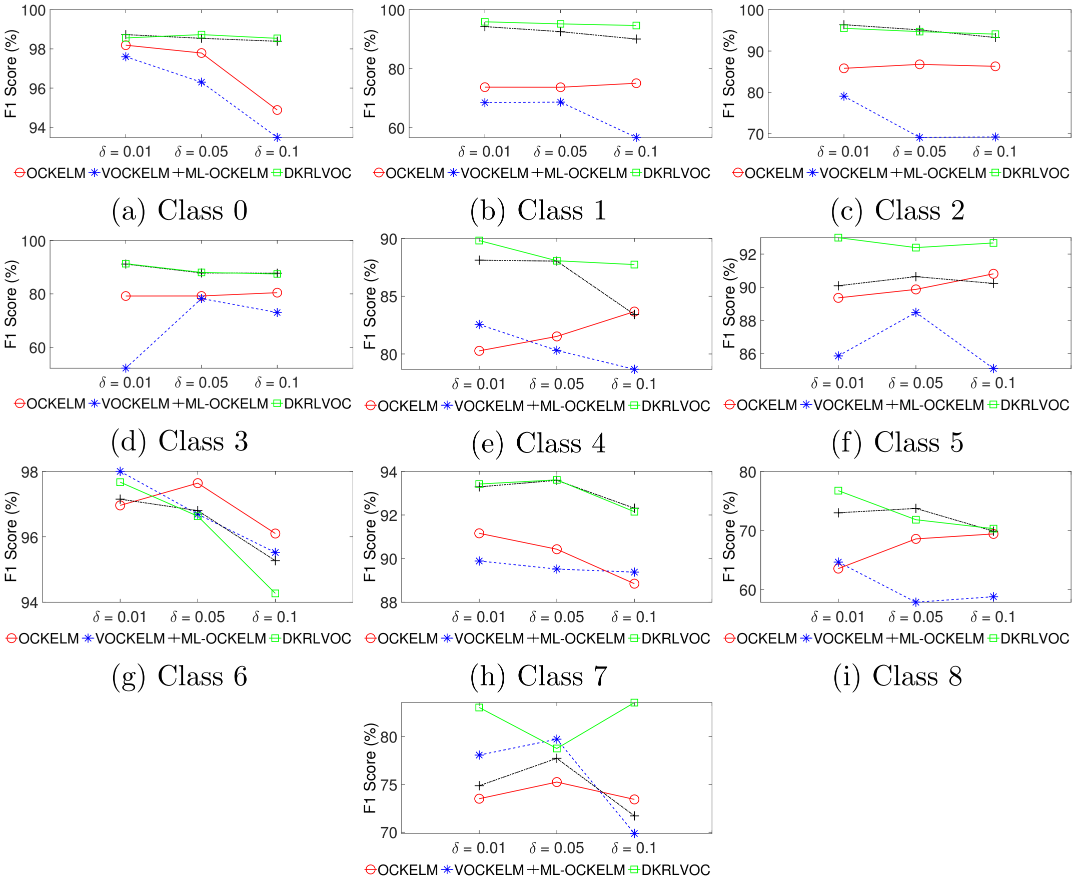
(a) Accuracy

(b) G-mean

(c) Precision

(d) Recall

Figure 4.4: Accuracy, G-mean, Precision and Recall plots for DKRLVOC and existing KRL-based one-class classifiers for optical digit datasets.



(a) Class 0

(b) Class 1

(c) Class 2

(d) Class 3

(e) Class 4

(f) Class 5

(g) Class 6

(h) Class 7

(i) Class 8

(j) Class 9

Figure 4.5: Variation of F_1 score with fraction of dismissal (δ) for DKRLVOC and existing KRL-based one-class classifiers for optical digit datasets.

4.3 Summary

In this chapter, we proposed the minimum variance embedded deep KRL-based method for OCC. We proposed to embed minimum variance information in the initial reconstruction-based layer in a multi-layer architecture and used KRL as a base classifier to perform OCC. The minimum variance embedding helped to minimize the data dispersion and forced the network output weights to focus in regions of low variance. The proposed method follows a deep architecture and comprises initial Q reconstruction-based layers and a final boundary-based OCC layer. As such, we were able to leverage both reconstruction and boundary-based approaches to achieve better performance than the existing one-class classifiers. The reconstruction-based layers reconstruct the essential information of the input data at the output layer, and utilize both representation learning and kernel learning to learn an effective representation of the input data. The final boundary-based layer was used to learn a boundary around the target class and perform OCC. We performed experiments on 24 benchmark datasets (14 small-size and 10 medium-size) and compared the performance with 14 existing state-of-the-art one-class classifiers. For small-size datasets, it was observed that DKRLVOC achieved the highest η_{F_1} in comparison to other one-class classifiers, with a significant difference of 5.94% when compared with non-KRL-based methods. When compared to single-layer KRL-based methods, there was an improvement of 4.97% in terms of η_{F_1}. For medium-size datasets as well, DKRLVOC scored the highest η_{F_1} with a significant difference of 6.9% compared to single-layer KRL-based methods. It can be concluded that DKRLVOC was able to outperform other existing one-class classifiers over a range of datasets.

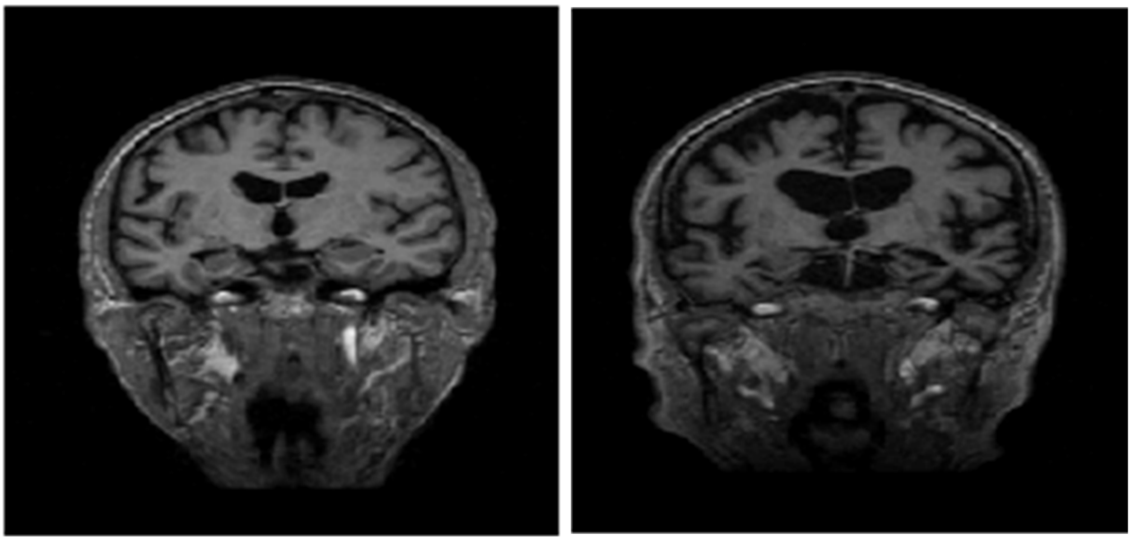
Further, in the next chapter, we utilize DKRLVOC for the identification of Alzheimer's and Breast cancer disease.

Chapter 5

Application of DKRLVOC: Identification of Alzheimer's and Breast Cancer Disease

In this chapter, we present the applicability of the proposed method, DKRLVOC, for the identification of Alzheimer's and Breast Cancer diseases using structural magnetic resonance image (sMRI) and histopathological image, respectively. Alzheimer's disease results in the degeneration of brain cells in a person. It is the most usual cause of dementia, where a person experiences a steady decline in his reasoning, behavioral and social skills leading to a disruption of his ability to function independently. It begins with forgetting recent events and conversations, further leading to severe impairment of memory and loss of ability to carry out everyday tasks. Breast cancer occurs due to the abnormal growth of some breast cells. A lump is formed due to the rapid division and continuous accumulation of these cells. The cells can spread to the lymph nodes or to the other parts of the body. In this chapter, we present the experiments and outcomes of our proposed method DKRLVOC on real-world biomedical datasets and compare the performance with existing one-class classifiers.

Further, we present the experimental discussion and results of DKRLVOC for the identification of Alzheimer's and Breast Cancer in Sections 5.1 and 5.2, respectively.



(a) Control normal (CN)

(b) Alzheimer's disease (AD)

Figure 5.1: sMRI images of Control Normal (CN) and Alzheimer's disease (AD) subjects from ADNI database.

5.1 Alzheimer's Disease

We have conducted experiments to identify Alzheimer's disease using sMRI. The sMRI data was procured from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database[1]. In particular, we have used 100 T1-weighted sMRIs comprising of 50 control normal (CN) and 50 Alzheimer's disease (AD) subjects. We present the sMRI for CN and AD subjects in Figure 5.1. The degeneration of neurons can be observed in the AD subject in Figure 5.1(b). In the ADNI dataset that we have used for our experiments, the subjects have their age varying in the range 60-90, with a mean age of 75.83 and a standard deviation of 6.07. We have used the Freesurfer software v6.0.1 recon-all pipeline [94, 95] to process the images and obtain volume and thickness measures of the brain. The processing yielded 34 cortical thickness measures, 23 subcortical tissue volumes, and 34 white matter tissue volumes for each image. We have normalized the volumetric data for variation in head size with division by total intracranial volume. In our experiments, we have used an 80%-20% train-test split ratio, meaning 80% of target and outlier samples are used for 5-fold cross-validation, and 20% is used for testing. 5-fold cross-validation is used to select optimal value for the parameters. The

[1] `adni.loni.usc.edu`

	Target Class	**#Target**	**#Outlier**	**#Features**
CN vs. AD	CN	50	50	91
AD vs. CN	AD	50	50	91

Table 5.1: Dataset specifications for the two cases of Alzheimer's disease identification.

optimal value for the regularization parameters $C^{(q)}$ and $C^{(f)}$ is selected from the range $\{2^{-3}, 2^{-2}, ..., 2^{3}\}$. The number of clusters for k-means clustering is selected from the range $\{1, 2, ..., 10\}$. The value of the fraction of dismissal (δ) is selected from the range $\{0.01, 0.05, 0.1\}$. All the methods employ the RBF kernel.

The datasets used for the experiments comprise of samples from two classes, that is, CN and AD. We perform experiments for the identification of Alzheimer's disease by considering two cases. The specifications for both cases are presented in Table 5.1. In the first case (i.e., CN vs. AD), we train the DKRLVOC model on the CN data, while in the second case (i.e., AD vs. CN), we train the model on AD data. Further, for each case, we use four different measures, namely, All features, Cortical thickness, Subcortical volume, and White matter volume, to train the model. We present and compare the results for both the cases with the existing kernel-based one-class classifiers in Table 5.2. In the table, the upper rows list both the cases along with all four measures for each case. The first column lists the existing kernel-based one-class classifiers and DKRLVOC, while the last column lists the η_{F_1} values for each classifier

	CN vs. AD				**AD vs. CN**				η_{F_1}
	All features	**Cortical thickness**	**Subcortical volume**	**White matter volume**	**All features**	**Cortical thickness**	**Subcortical volume**	**White matter volume**	$\begin{pmatrix}\textbf{CN}\\\textbf{vs.}\\\textbf{AD}\end{pmatrix}$
OCSVM [1]	70.59	81.82	74.07	66.67	53.85	64.29	80	40	73.29
SVDD [35]	62.5	77.78	69.57	66.67	58.33	61.54	75	43.48	69.13
OCKELM [37]	80	81.82	69.23	66.67	66.67	68.97	66.67	66.67	74.43
VOCKELM [13]	76.19	81.82	69.23	66.67	62.07	66.67	75	66.67	73.48
ML-OCKELM [38]	80	86.96	69.23	76.92	62.07	66.67	64.29	66.67	78.28
DKRLVOC	81.82	86.96	75	76.92	64	68.97	76.92	66.67	**80.18**

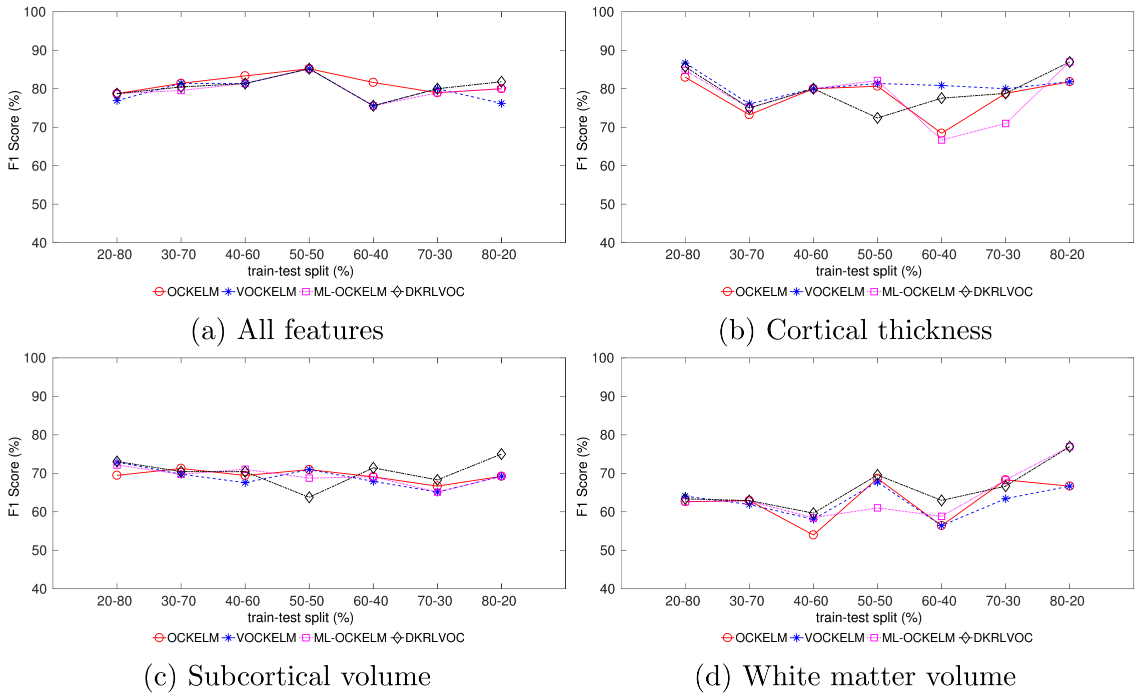
Table 5.2: Performance in terms of F_1 score over different measures for the two cases of Alzheimer's disease identification.

for the CN vs. AD case. Note that we have used η_{F_1} as the final metric for the performance evaluation of DKRLVOC against the existing one-class classifiers. In Table 5.2, it can be observed that the scores for AD vs. CN are generally less than the corresponding scores for CN vs. AD cases for the same measure. The reason for the same can be attributed to the variation in neurodegeneration in AD patients. Hence, we conclude that training the one-class models on CN data is more efficient than training on AD data. Also, for the CN vs. AD case, the measures all features and cortical thickness report better F_1 scores than the other measures, leading to the conclusion that all features and cortical thickness are prominent measures for the identification of Alzheimer's from sMRIs. Since training on CN vs. AD case was observed to be more efficient than training on AD vs. CN case, we have reported η_{F_1} values for CN vs. AD case only in Table 5.2. DKRLVOC scored the highest η_{F_1} (highlighted in bold red) in comparison to other one-class classifiers, with a significant increase of 6.89% against non-KRL-based methods, i.e., OCSVM and SVDD. Additionally, DKRLVOC showed an improvement of 5.75% against single-layer KRL-based methods, i.e., OCKELM and VOCKELM. Also, DKRLVOC scored the highest F_1 score in comparison to most of the other one-class classifiers for all four measures in the CN vs. AD case (highlighted in blue). The comparatively better performance of DKRLVOC can be attributed to the fact that DKRLVOC uses minimum variance embedding, which helps to minimize the data dispersion. Further, the multiple sequentially stacked KRL-based autoencoders help to learn the essential features from the input data.

We present the variation in the performance of the one-class classifiers for different train-test split ratios in Figure 5.2. The following observations can be made from the figure,

(1) As can be seen in Figures 5.2(a), 5.2(c), and 5.2(d), DKRLVOC usually achieves a better F_1 score than other one-class classifiers in most train-test splits. This signifies that it generally does a better job of identifying Alzheimer's than the other classifiers.

(2) Figures 5.2(a) and 5.2(b) show a better F_1 score for the one-class classifiers over



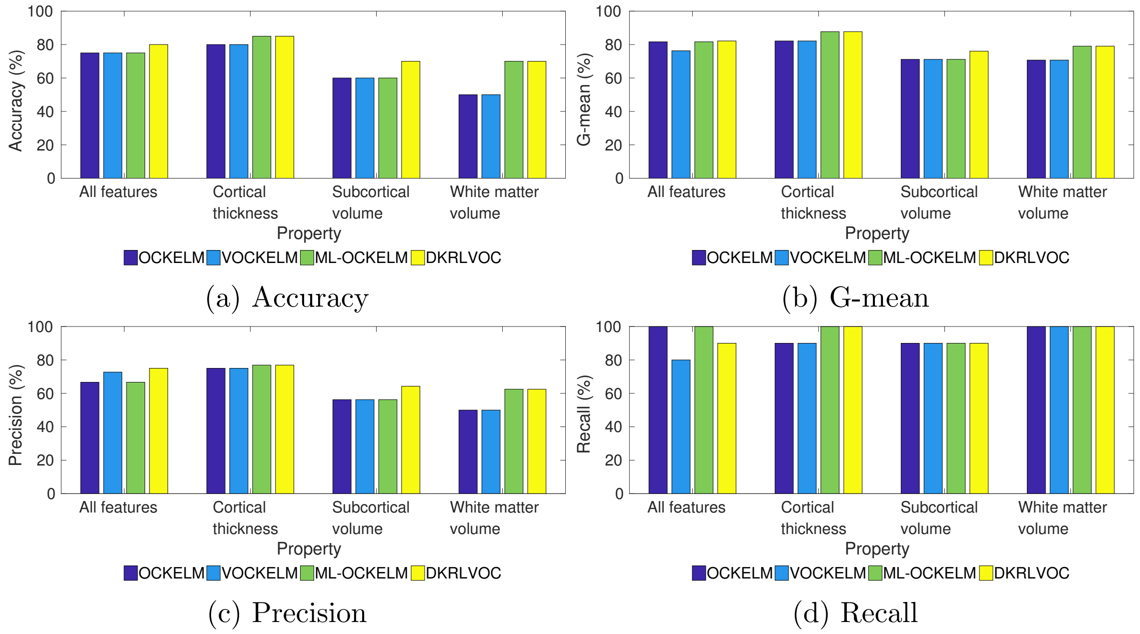
(a) All features

(b) Cortical thickness

(c) Subcortical volume

(d) White matter volume

Figure 5.2: Variation of F_1 score over different train-test splits for various measures of Alzheimer's disease dataset.



(a) Accuracy

(b) G-mean

(c) Precision

(d) Recall

Figure 5.3: Accuracy, G-mean, Precision and Recall plots for different measures of Alzheimer's disease dataset.

different train-test splits than the other two measures in Figure 5.2. It helps us to conclude that all features and cortical thickness are more prominent in identifying Alzheimer's from sMRI images.

(3) The maximum F_1 score is reported for DKRLVOC at 80-20 train-test split for 3 out of 4 measures, as can be seen in Figures 5.2(b), 5.2(c), and 5.2(d). The reason can be attributed to the fact that more training data is available in the 80-20 train-test split.

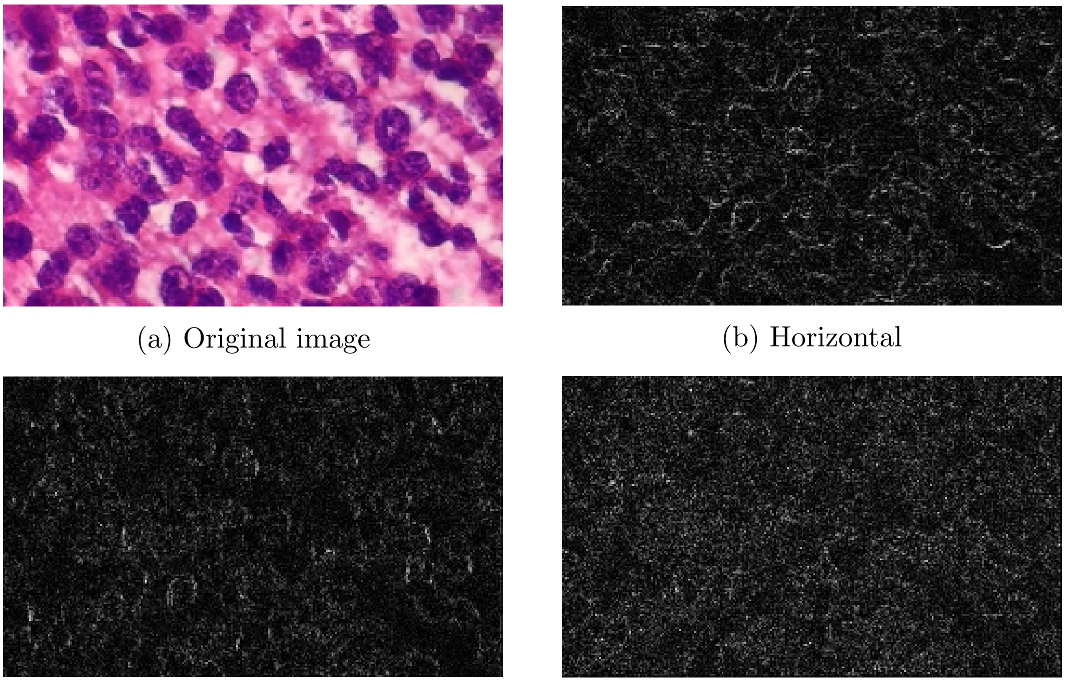
(4) It can be observed that for the lower train-test splits, the F_1 score for all the one-class classifiers is reported to be in close proximity. This signifies that for less training data, all the classifiers have similar performance.

For reference, we also present the experimental results based on accuracy, g-mean, precision, and recall metrics for DKRLVOC and other KRL-based methods in Figure 5.3. It can be observed that out of 4 cases, DKRLVOC achieves the highest accuracy, g-mean, precision, and recall for 4, 4, 4, and 3 cases, respectively.

Further, in Section 5.2, we apply DKRLVOC for the identification of breast cancer disease and compare the results with other one-class classifiers.

5.2 Breast Cancer Disease

We have conducted experiments to identify breast cancer by training the DKRLVOC model on BreakHis [39] histopathological image dataset and compare the results with the existing kernel-based one-class classifiers. For this purpose, we use 1240 images with 400X magnification from the dataset. The selected images belong to either of the two categories, benign or malignant. The benign category consists of four subclasses, namely, adenosis (AN), fibroadenoma (FA), phyllodes tumor (PT), and tubular adenoma (TA) comprising of 106, 237, 115, and 130 images, respectively. Further, the malignant category consists of four subclasses, namely, ductal carcinoma (DC), lobular carcinoma (LC), mucinous carcinoma (MC), papillary carcinoma (PC) comprising of 208, 137, 169, and 138 images, respectively. We have transformed the


(a) Original image

(b) Horizontal

(c) Vertical

(d) Diagonal

Figure 5.4: Histopathological image of (a) ductal carcinoma. Image of different detail coefficients obtained after wavelet transform on image (a) are shown in subfigures (b)-(d).

original images into grayscale images to extract the essential features and performed a wavelet transform using Daubechies-4 wavelet up to 3 levels of decomposition [96, 97], as shown in Figure 5.4. We have obtained the feature vectors by concatenating the approximation and detail coefficients. The feature vectors are not normalized. The rest of the experimental setup is kept the same as the setup in Section 5.1. Each of the benign and malignant categories consists of 4 subclasses. We have considered all possible pairs between two categories and obtained 16 one-class datasets, such that each one-class dataset contains samples from a subclass of both the categories. The specifications of these one-class datasets are provided in Table 5.3. The subclass from the benign category is always kept as the target class. The datasets are prepared in this manner to show the ability of DKRLVOC in identifying non-cancerous tumor in all possible pairs of benign and malignant categories.

We present the results for the experiments on the one-class breast cancer datasets

	Target Class	**Outlier Class**	**#Target**	**#Outlier**	**#Features**
AN Vs. DC	Adenosis	Ductalcarcinoma	106	208	768
AN Vs. LC	Adenosis	Lobularcarcinoma	106	137	768
AN Vs. MC	Adenosis	Mucinouscarcinoma	106	169	768
AN Vs. PC	Adenosis	Papillarycarcinoma	106	138	768
FA Vs. DC	Fibroadenoma	Ductalcarcinoma	237	208	768
FA Vs. LC	Fibroadenoma	Lobularcarcinoma	237	137	768
FA Vs. MC	Fibroadenoma	Mucinouscarcinoma	237	169	768
FA Vs. PC	Fibroadenoma	Papillarycarcinoma	237	138	768
PT Vs. DC	Phyllodes tumor	Ductalcarcinoma	115	208	768
PT Vs. LC	Phyllodes tumor	Lobularcarcinoma	115	137	768
PT Vs. MC	Phyllodes tumor	Mucinouscarcinoma	115	169	768
PT Vs. PC	Phyllodes tumor	Papillarycarcinoma	115	138	768
TA Vs. DC	Tubular adenoma	Ductalcarcinoma	130	208	768
TA Vs. LC	Tubular adenoma	Lobularcarcinoma	130	137	768
TA Vs. MC	Tubular adenoma	Mucinouscarcinoma	130	169	768
TA Vs. PC	Tubular adenoma	Papillarycarcinoma	130	138	768

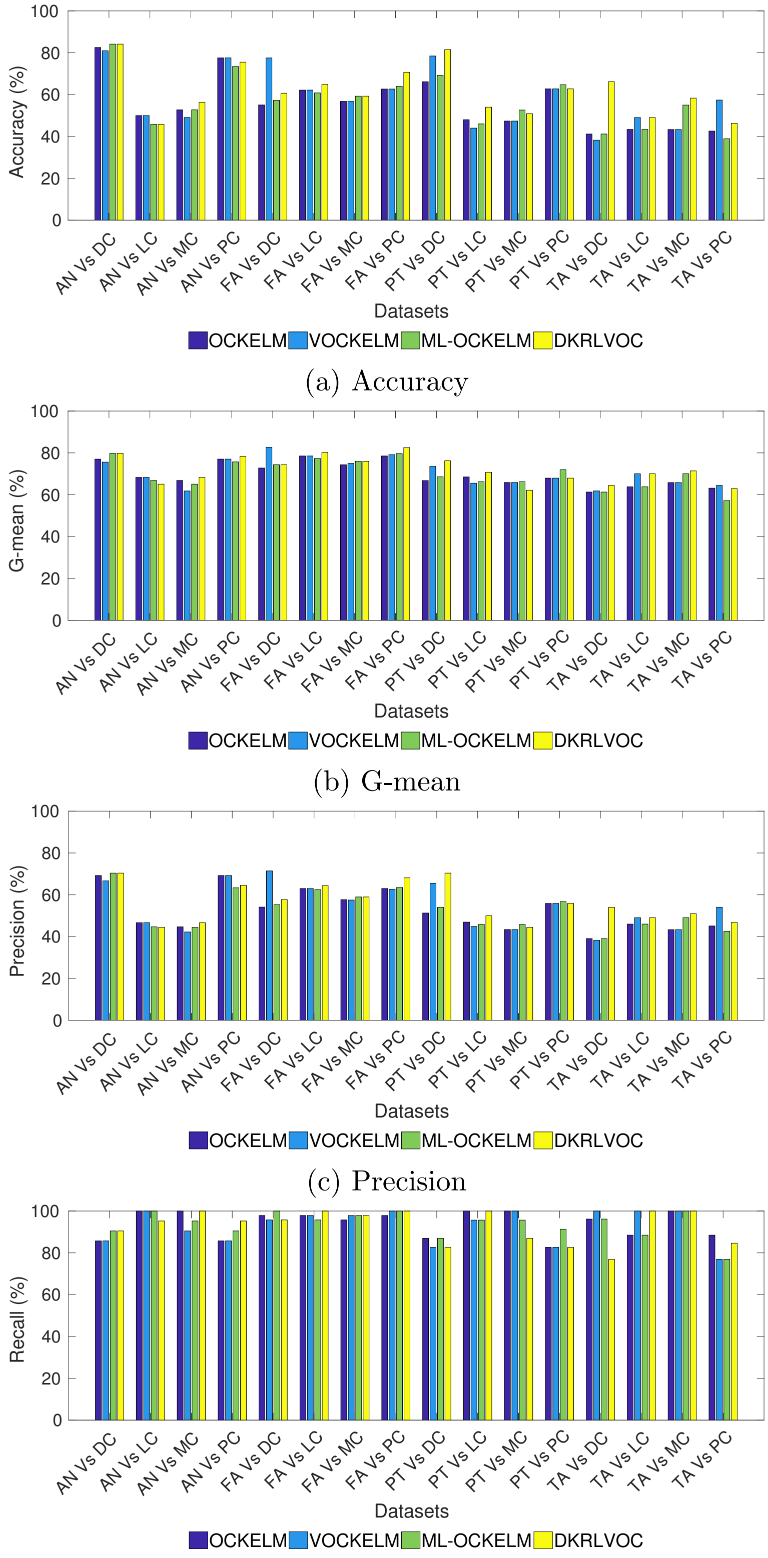
Table 5.3: Specification of Breast Cancer one-class datasets. Here, AN, FA, PT, TA, DC, LC, MC, PC refer to Adenosis, Fibroadenoma, Phyllodes tumor, Tubular adenoma, Ductalcarcinoma, Lobularcarcinoma, Mucinouscarcinoma, and Papillarycarcinoma, respectively.

	OCSVM [1]	**SVDD [35]**	**OCKELM [37]**	**VOCKELM [13]**	**ML-OCKELM [38]**	**DKRLVOC**
AN Vs. DC	79.07	71.43	76.6	75	79.17	79.17
AN Vs. LC	58.06	49.12	63.64	63.64	61.76	60.61
AN Vs. MC	64.41	50.98	61.76	57.58	60.61	63.64
AN Vs. PC	72.73	66.67	76.6	76.6	74.51	76.92
FA Vs. DC	67.2	66.12	69.7	81.82	71.21	72
FA Vs. LC	74.58	72.07	76.67	76.67	75.63	78.33
FA Vs. MC	71.19	67.86	72	72.44	73.6	73.6
FA Vs. PC	75.68	72.9	76.67	77.05	77.69	81.03
PT Vs. DC	59.7	65.57	64.52	73.08	66.67	76
PT Vs. LC	57.58	56.25	63.89	61.11	61.97	66.67
PT Vs. MC	52.94	49.23	60.53	60.53	61.97	58.82
PT Vs. PC	66.67	61.82	66.67	66.67	70	66.67
TA Vs. DC	52.87	54.76	55.56	55.32	55.56	63.49
TA Vs. LC	53.52	52.17	60.53	65.82	60.53	65.82
TA Vs. MC	51.85	51.85	60.47	60.47	65.82	67.53
TA Vs. PC	59.46	62.86	59.74	63.49	54.79	60.27
η_{F_1}	63.59	60.73	66.6	67.96	66.97	**69.41**

Table 5.4: Performance in terms of F_1 score over different Breast Cancer datasets.

for DKRLVOC and the other existing kernel-based one-class classifiers in Table 5.4. As evident from the table, DKRLVOC obtained the highest η_{F_1} score (highlighted in bold red) in comparison to other one-class classifiers, with a significant increase of 5.82% against non-KRL-based methods, i.e., OCSVM and SVDD. Also, DKRLVOC performed better than the other methods for 10 out of 16 datasets (highlighted in blue), obtaining the highest score of 81.03 for the case FA Vs. PC. The reduction in intra-class variance at first layer helps in better separation of target class from outliers leading to improved performance of DKRLVOC. Further, the presence of multiple reconstruction-based layers helps to learn the essential features from input data. The above observations show that DKRLVOC can be successfully applied in the biomedical field.

For reference, we also provide the experimental results based on accuracy, g-mean, precision, and recall metrics for DKRLVOC along with other KRL-based methods in Figure 5.5. As evident, DKRLVOC obtains the best accuracy, g-mean, precision, and recall against other one-class classifiers for 10, 11, 10, and 9 datasets, respectively. Further, we present the variation in the performance of the one-class classifiers for different train-test split ratios in Figure 5.6. Note that only the target class samples are used for training the one-class classifiers. It can be observed from the figure that for the datasets which have less number of target class samples, the score obtained by the classifiers over different train-test splits is relatively low. For the datasets having relatively more number of target class samples, i.e., datasets with target class fibroadenoma, the scores are consistently better. This can be seen in Figures 5.6(e), 5.6(f), 5.6(g), and 5.6(h).



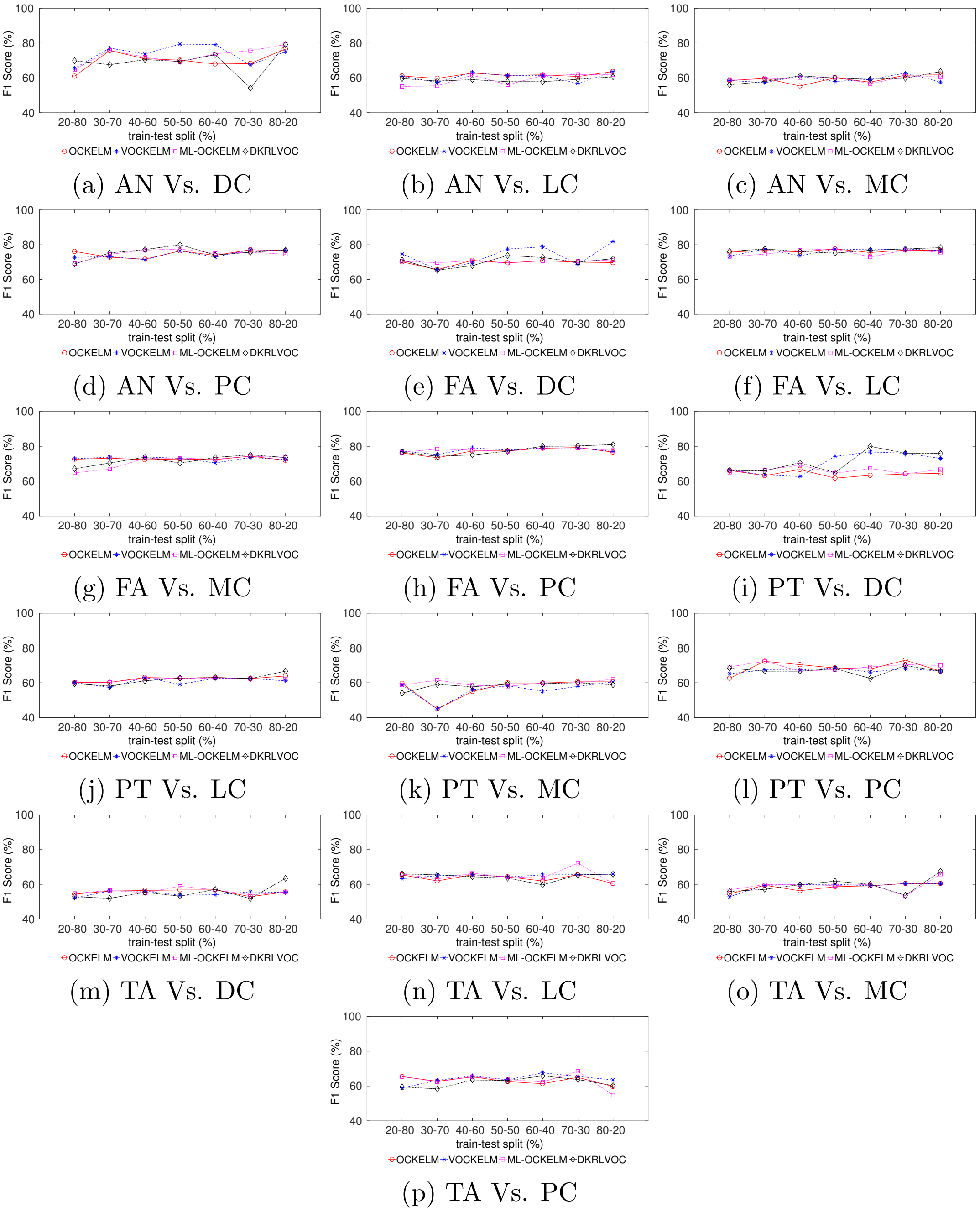
(a) Accuracy

(b) G-mean

(c) Precision

(d) Recall

Figure 5.5: Accuracy, G-mean, Precision and Recall plots for Breast Cancer disease one-class datasets.



(a) AN Vs. DC (b) AN Vs. LC (c) AN Vs. MC

(d) AN Vs. PC (e) FA Vs. DC (f) FA Vs. LC

(g) FA Vs. MC (h) FA Vs. PC (i) PT Vs. DC

(j) PT Vs. LC (k) PT Vs. MC (l) PT Vs. PC

(m) TA Vs. DC (n) TA Vs. LC (o) TA Vs. MC

(p) TA Vs. PC

Figure 5.6: Variation of F_1 score over different train-test splits for Breast Cancer disease one-class datasets.

5.3 Summary

In this chapter, we applied the proposed method, DKRLVOC, to identify Alzheimer's disease using sMRI data and Breast Cancer using histopathological image datasets. For the detection of Alzheimer's disease, we utilized four measures, namely, All features, Cortical thickness, Subcortical volume, and White matter volume, to train DKRLVOC and the other one-class classifiers. We considered two approaches to training the DKRLVOC model. In the first approach, we trained the model using the data of normal (CN) subjects, while in the second approach, we trained the model using the data from diseased (AD) subjects. It was observed that it was more efficient to train the model using CN data, rather than AD data. This was attributed to the variation in neurodegeneration in AD patients. Further, it was found that all features and cortical thickness are prominent measures for the identification of Alzheimer's from sMRI images. DKRLVOC scored the highest η_{F_1} in comparison to other one-class classifiers. DKRLVOC reported an improvement of 6.89% against non-KRL-based methods, i.e., OCSVM and SVDD, and 5.75% against single-layer KRL-based methods, i.e., OCKELM and VOCKELM. For the detection of Breast Cancer, we prepared 16 one-class datasets, where each dataset was comprised of samples from one of the four subclasses of both benign and malignant categories. The aim was to explore how efficiently DKRLVOC can identify the non-cancerous tumor in each case. It was observed that DKRLVOC obtained the highest η_{F_1} score in comparison to other one-class classifiers, with a significant increase of 5.82% against non-KRL-based methods, i.e., OCSVM and SVDD. The better performance of DKRLVOC against other one-class classifiers is attributed to the utilization of minimum variance information that helps to minimize the data dispersion, leading to better separation of the target class samples from the outliers. Further, the presence of multiple reconstruction-based layers helps to learn the essential information from the input data. From the above observations, it can be concluded that DKRLVOC is a better alternative to the existing kernel-based one-class classifiers for the identification of Alzheimer's and Breast Cancer diseases.

Chapter 6

Conclusions and Future Work

This thesis primarily explored the concept of variance minimization for the reconstruction-framework KRL-based approach for OCC. We leveraged minimum variance embedding to minimize the data dispersion and combined it with representation learning and kernel learning to learn an effective representation of the input data. First, we developed the single-layer minimum variance embedded auto-associative KRL-based one-class classifier. The proposed method follows a reconstruction-based approach to OCC and uses KRL-based autoencoders to learn an efficient representation of the input data. Second, we developed the minimum variance embedded deep KRL-based method for OCC. The proposed method follows a multi-layer architecture and utilizes minimum variance information at the first layer. Further, it is composed of multiple KRL-based reconstruction-based layers stacked sequentially and a final boundary-based OCC layer. The proposed methods were tested on various benchmark datasets, and the results were compared with different existing state-of-the-art one-class classifiers. Further, we applied the DKRLVOC method for the identification of Alzheimer's and Breast Cancer disease. The results obtained exhibited that the proposed methods outperformed existing one-class classifiers in terms of η_{F_1} and F_1 score due to the minimum variance embedding that was responsible for a reduction in the dispersion of data.

Further, we discuss a summary of the contributions achieved in this thesis in Section 6.1, followed by the possible future directions in Section 6.2.

6.1 Summary of Contributions

We have achieved the objectives specified in Section 1.3 in this thesis by making the following main contributions:

(1) **Minimum Variance Embedded Auto-associative KRL-based Method for OCC:** We proposed the minimum variance embedded KRL-based autoencoder (VAAKRL) for OCC. VAAKRL leverages the minimum variance information to minimize the data dispersion of the target class and force the network output weights to focus in regions of low variance. VAAKRL follows a single-layer architecture and uses a reconstruction-based approach to perform OCC. It follows the idea that, since the model is trained solely on target class samples, the outliers should have a high reconstruction error in comparison to target class samples. Hence, the deviation in reconstruction error is used to define a threshold criterion that decides the membership for the new samples. We conducted experiments on 14 datasets and compared the results with 14 existing one-class classifiers. VAAKRL achieved the highest η_{F_1} in comparison to the existing classifiers, with a significant improvement of 6.9% over the existing non-kernel-based one-class classifiers. However, VAAKRL showed a slightly better η_{F_1} score in comparison to the existing boundary-based and reconstruction-based KRL classifiers. To further improve the KRL-based one-class classifier, we proposed a model by embedding the minimum variance information in a multi-layer architecture.

(2) **Minimum Variance Embedded Deep KRL-based Method for OCC:** We explored the effectiveness of variance minimization in a multi-layer architecture by combining both the reconstruction-based and boundary-based frameworks in a single architecture for OCC (DKRLVOC). The deep architecture of DKRLVOC comprises of multiple sequentially stacked reconstruction-based layers and a final boundary-based OCC layer. The reconstruction-based KRL layers are responsible for reconstructing the essential information of the input data at the output layer and learning an effective representation of the data. The final boundary-based layer is responsible for performing OCC by learning a discrimination boundary

around the target class data. The distance between the samples is used to define a threshold criteria, which further decides the membership of a new sample. We conducted experiments on 14 small-size and 10 medium-size benchmark datasets and compared the results with 14 existing one-class classifiers. DKRLVOC achieved the highest η_{F_1} in comparison to the existing classifiers for both the small-size and medium-size datasets, with a significant improvement of 4.97% for small-size datasets and 6.9% for medium-size datasets against single-layer KRL-based classifiers. Further for the small-size datasets, there was an improvement of 5.94% when compared to the existing non-KRL-based classifiers. The success of DKRLVOC can be attributed to the use of minimum variance embedding and the multiple sequentially stacked KRL-based autoencoders. The minimum variance embedding helped to minimize the data dispersion, and the KRL-based autoencoders helped to learn the essential features from the input data.

(3) **Application of DKRLVOC for the identification of Alzheimer's and Breast Cancer Diseases:** We applied DKRLVOC for the identification of Alzheimer's and Breast Cancer diseases. For Alzheimer's, it was observed that it was more efficient to train the model using CN data, rather than AD data. Further, it was observed that all features and cortical thickness are prominent measures for the identification of Alzheimer's from sMRI data. For both Alzheimer's and Breast Cancer, DKRLVOC performed better than the existing one-class classifiers, with a significant improvement of 6.89% for Alzheimer's and 5.82% for Breast Cancer against non-KRL-based classifiers. It can be concluded that DKRLVOC is a better alternative to the existing kernel-based one-class classifiers for the identification of Alzheimer's and Breast Cancer diseases.

6.2 Future Research Directions

We understand that our current work can be explored in the following future directions:

(1) **Minimum Variance Embedded KRL-based OCC using machines-teaching-machines paradigm:**

Privileged information [98] and distillation [99] are two techniques that enable machines to learn from other machines. Privileged information is additional information about the data which is available only during the time of training and is absent at the test time. In distillation, a simple machine learns a complex task by imitating the solution of a flexible machine. One future direction is to utilize the paradigm to improve the proposed minimum variance embedded KRL-based one-class classifiers.

(2) **Minimum Variance Embedded KRL-based OCC to handle streaming data in an online setting:**

Online learning [100, 101] has attracted researchers in recent years due to its capability to handle a high volume of streaming data with less computational and storage costs. In online learning, a model is built based on the currently available data, and then it is continuously updated as the next samples arrive for training. The KRL-based one-class classifiers developed as part of my thesis work can handle only stationary data. As most of the real-world problems deal with streaming data, further research can be done to enable the proposed classifiers in this thesis to handle streaming data in an online setting. This will enable the classifiers to learn in a real-time environment where the data characteristics keep changing over time.

(3) **BatchEnsemble KRL-based one-class classifiers:**

BatchEnsemble [102] is an ensemble method whose computational and memory costs are significantly lower than the typical ensembles. Unlike traditional ensembles, BatchEnsemble is mini-batch friendly, where it is parallelizable across devices like typical ensembles but also parallelizable within a device. A possible future research direction is to utilize the BatchEnsemble mechanism in KRL-based one-class classifiers for handling large-scale data.

Bibliography

[1] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt. Support vector method for novelty detection. In *Advances in Neural Information Processing Systems*, volume 12, pages 582–588, Denver, CO, 1999. MIT Press.

[2] Y. Guerbai, Y. Chibani, and B. Hadjadji. The effective use of the one-class svm classifier for handwritten signature verification based on writer-independent parameters. *Pattern Recognition*, 48(1):103–113, 2015.

[3] S. Luca, D. A. Clifton, and B. Vanrumste. One-class classification of point patterns of extremes. *Journal of Machine Learning Research*, 17(1):6581–6601, 2016.

[4] S. S. Khan and M. G. Madden. A survey of recent trends in one class classification. In *Proceedings of the 20th Irish Conference on Artificial Intelligence and Cognitive Science*, pages 188–197, Dublin, Ireland, 2009. Springer.

[5] D.M.J. Tax. *One-class classification: Concept learning in the absence of counter-examples.* PhD thesis, Technische Universiteit Delft, Netherlands, 2002.

[6] Shehroz S Khan and Michael G Madden. One-class classification: taxonomy of study and review of techniques. *The Knowledge Engineering Review*, 29(3):345–374, 2014.

[7] Xiaoming Wang, Fu-lai Chung, and Shitong Wang. On minimum class locality preserving variance support vector machine. *Pattern Recognition*, 43(8):2753–2762, 2010.

[8] Xiaobo Chen, Jian Yang, Qiaolin Ye, and Jun Liang. Recursive projection twin support vector machine via within-class variance minimization. *Pattern Recognition*, 44(10-11):2643–2655, 2011.

[9] Qiaolin Ye, Chunxia Zhao, and Ning Ye. Least squares twin support vector machine classification via maximum one-class within class variance. *Optimization methods and software*, 27(1):53–69, 2012.

[10] Ming Ji and Jiawei Han. A variance minimization criterion to active learning on graphs. In *Artificial Intelligence and Statistics*, pages 556–564, 2012.

[11] Alexandros Iosifidis, Anastasios Tefas, and Ioannis Pitas. Minimum class variance extreme learning machine for human action recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(11):1968–1979, 2013.

[12] Alexandros Iosifidis, Anastasios Tefas, and Ioannis Pitas. Minimum variance extreme learning machine for human action recognition. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5427–5431. IEEE, 2014.

[13] V. Mygdalis, A. Iosifidis, A. Tefas, and I. Pitas. One class classification applied in facial image analysis. In *Proceedings of the IEEE International Conference on Image Processing*, pages 1644–1648, Phoenix, AZ, USA, 2016. IEEE.

[14] Guillermo L Grinblat, Lucas C Uzal, and Pablo M Granitto. Abrupt change detection with one-class time-adaptive support vector machines. *Expert Systems with Applications*, 40(18):7242–7249, 2013.

[15] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.

[16] Colin Bellinger, Shiven Sharma, and Nathalie Japkowicz. One-class classification–from theory to practice: A case-study in radioactive threat detection. *Expert Systems with Applications*, 108:223–232, 2018.

[17] Fahad Sohrab, Jenni Raitoharju, Moncef Gabbouj, and Alexandros Iosifidis. Subspace support vector data description. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 722–727. IEEE, 2018.

[18] C. Gautam, A. Tiwari, S. Suresh, and K. Ahuja. Adaptive online learning with regularized kernel for one-class classification. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pages 1–16, 2019.

[19] H.J. Shin, D.H. Eom, and S.S. Kim. One-class support vector machines-an application in machine fault detection and classification. *Computers & Industrial Engineering*, 48(2):395–408, 2005.

[20] Jianguo Zhang, Kai-Kuang Ma, Meng-Hwa Er, and Vincent Chong. Tumor segmentation from magnetic resonance imaging by learning via one-class support vector machine. In *International Workshop on Advanced Image Technology*, pages 207–211, Singapore, January 2004.

[21] W. Fan, M. Miller, S. Stolfo, W. Lee, and P. Chan. Using artificial anomalies to detect unknown and known network intrusions. *Knowledge and Information Systems*, 6(5):507–527, Sep 2004.

[22] Moshe Koppel and Jonathan Schler. Authorship verification as a one-class classification problem. In *Proceedings of the Twenty-First International Conference on Machine Learning*, page 62, New York, NY, USA, 2004. Association for Computing Machinery.

[23] L. Manevitz and M. Yousef. One-class document classification via neural networks. *Neurocomputing*, 70(7-9):1466–1481, 2007.

[24] C.P. Diehl and J.B. Hampshire. Real-time object classification and novelty detection for collaborative video surveillance. In *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290)*, volume 3, pages 2620–2625. IEEE, 2002.

[25] M. Markou and S. Singh. A neural network-based novelty detector for image sequence analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1664–1677, 2006.

[26] H. G. Bu, J. Wang, and X. B. Huang. Fabric defect detection based on multiple fractal features and support vector data description. *Engineering Applications of Artificial Intelligence*, 22(2):224–235, 2009.

[27] J. Mourão-Miranda, D. R. Hardoon, T. Hahn, A. F. Marquand, S. C. R. Williams, J. Shawe-Taylor, and M. Brammer. Patient classification as an outlier detection problem: an application of the one-class support vector machine. *Neuroimage*, 58(3):793–804, 2011.

[28] T. C. Minter. Single-class classification. In *Symposium on Machine Processing of Remotely Sensed Data*, pages 2A12–2A15. IEEE, 1975.

[29] M. M. Moya, M. W. Koch, and L. D. Hostetler. One-class classifier networks for target recognition applications. Technical report, Sandia National Labs., Albuquerque, United States, 1993.

[30] Christopher M Bishop. Novelty detection and neural network validation. *IEEE Proceedings-Vision, Image and Signal processing*, 141(4):217–222, 1994.

[31] G. Ritter and M. Gallegos. Outliers in statistical pattern recognition and an application to automatic chromosome classification. *Pattern Recognition Letters*, 18(6):525–539, 1997.

[32] N. Japkowicz. *Concept-learning in the absence of counter-examples: An autoassociation-based approach to classification.* PhD thesis, Rutgers, The State University of New Jersey, 1999.

[33] D. M. J. Tax and R. P. W. Duin. Support vector domain description. *Pattern Recognition Letters*, 20(11-13):1191–1199, 1999.

[34] H. Hoffmann. Kernel PCA for novelty detection. *Pattern Recognition*, 40(3):863–874, 2007. Software available at `http://www.heikohoffmann.de/kpca.html`.

[35] D. M. J. Tax and R. P. W. Duin. Support vector data description. *Machine Learning*, 54(1):45–66, 2004.

[36] Y. S. Choi. Least squares one-class support vector machine. *Pattern Recognition Letters*, 30(13):1236–1240, 2009.

[37] Q. Leng, H. Qi, J. Miao, W. Zhu, and G. Su. One-class classification with extreme learning machine. *Mathematical Problems in Engineering*, 2015:1–11, 2015.

[38] H. Dai, J. Cao, T. Wang, M. Deng, and Z. Yang. Multilayer one-class extreme learning machine. *Neural Networks*, 115:11–22, 2019.

[39] F. A. Spanhol, L. S. Oliveira, C. Petitjean, and L. Heutte. A dataset for breast cancer histopathological image classification. *IEEE Transactions on Biomedical Engineering*, 63(7):1455–1462, July 2016.

[40] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[41] Kevin P Murphy. *Machine learning: a probabilistic perspective.* MIT press, 2012.

[42] Manfred K Warmuth and Dima Kuzmin. Online variance minimization. In *International Conference on Computational Learning Theory*, pages 514–528. Springer, 2006.

[43] Thomas Hofmann, Aurelien Lucchi, Simon Lacoste-Julien, and Brian McWilliams. Variance reduced stochastic gradient descent with neighbors. In *Advances in Neural Information Processing Systems*, pages 2305–2313, 2015.

[44] Róbert Ormándi. Variance minimization least squares support vector machines for time series analysis. In *2008 Eighth IEEE International Conference on Data Mining*, pages 965–970. IEEE, 2008.

[45] Xiaofei He, Ming Ji, Chiyuan Zhang, and Hujun Bao. A variance minimization criterion to feature selection using laplacian regularization. *IEEE transactions on pattern analysis and machine intelligence*, 33(10):2013–2025, 2011.

[46] Neal Jean, Sang Michael Xie, and Stefano Ermon. Semi-supervised deep kernel learning: Regression with unlabeled data by minimizing predictive variance. In *Advances in Neural Information Processing Systems*, pages 5322–5333, 2018.

[47] M. E. Abbasnejad, D. Ramachandram, and R. Mandava. A survey of the state of the art in learning the kernels. *Knowledge and information systems*, 31(2):193–221, 2012.

[48] C. Gautam, A. Tiwari, and Q. Leng. On the construction of extreme learning machine for online and offline one-class classification—an expanded toolbox. *Neurocomputing*, 261:126–143, 2017.

[49] Bernhard Scholkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press, 2001.

[50] J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society, London*, 209:415–446, 1909.

[51] J. A. K. Suykens, T. V. Gestel, and J. D. Brabanter. *Least squares support vector machines.* World Scientific, 2002.

[52] T. V. Gestel, J. A. K. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B. De Moor, and J. Vandewalle. Benchmarking least squares support vector machine classifiers. *Machine Learning*, 54(1):5–32, 2004.

[53] Jens Hainmueller and Chad Hazlett. Kernel regularized least squares: Reducing misspecification bias with a flexible and interpretable machine learning approach. *Political Analysis*, 22(2):143–168, 2014.

[54] Ryan Michael Rifkin. *Everything old is new again: a fresh look at historical approaches in machine learning.* PhD thesis, Massachusetts Institute of Technology, 2002.

[55] G. B. Huang, Q. Y. Zhu, and C. K. Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1-3):489–501, 2006.

[56] G. B. Huang. An insight into extreme learning machines: random neurons, random features and kernels. *Cognitive Computation*, 6(3):376–390, 2014.

[57] G. B. Huang, H. Zhou, X. Ding, and R. Zhang. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(2):513–529, 2012.

[58] W. Yan. One-class extreme learning machines for gas turbine combustor anomaly detection. In *Proceedings of the International Joint Conference on Neural Networks*, pages 2909–2914, Vancouver, BC, Canada, 2016. IEEE.

[59] S. Wang, E. Zhu, and J. Yin. Video anomaly detection based on ulgp-of descriptor and one-class elm. In *Proceedings of the International Joint Conference on Neural Networks*, pages 2630–2637, Vancouver, BC, Canada, 2016. IEEE.

[60] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[61] D. Lituiev. Autoencoders: a bibliographic survey. `https://dslituiev.github.io/deeplearning/2017/04/19/autoencoders-bibliographic-survey.html`, 2019. [Online; accessed 30-April-2019].

[62] G. E. Hinton and R. S. Zemel. Autoencoders, minimum description length and helmholtz free energy. In *Advances in Neural Information Processing Systems*, pages 3–10, Denver, Colorado, USA, 1994. Morgan-Kaufmann.

[63] L Yann. *Connectionist models of learning.* PhD thesis, Universite Paris, 1987.

[64] Hervé Bourlard and Yves Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological cybernetics*, 59(4-5):291–294, 1988.

[65] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[66] D. Erhan, Y. Bengio, A. Courville, P. A. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(2):625–660, 2010.

[67] F. Zhuang, D. Luo, X. Jin, H. Xiong, P. Luo, and Q. He. Representation learning via semi-supervised autoencoder for multi-task learning. In *Proceedings of the IEEE International Conference on Data Mining*, pages 1141–1146, Atlantic City, NJ, USA, 2015. IEEE.

[68] Y. Pu, Z. Gan, R. Henao, X. Yuan, C. Li, A. Stevens, and L. Carin. Variational autoencoder for deep learning of images, labels and captions. In *Advances in Neural Information Processing Systems*, pages 2352–2360, Barcelona, Spain, 2016. Curran Associates Inc.

[69] P. Vincent, H. Larochelle, Y. Bengio, and P. A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1096–1103, Helsinki, Finland, 2008. ACM.

[70] S. Zhai, Y. Cheng, W. Lu, and Z. Zhang. Deep structured energy based models for anomaly detection. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48, pages 1100–1109, New York, NY, USA, 2016.

[71] H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, D. Pei, Y. Feng, J. Chen, Z. Wang, and H. Qiao. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the World Wide Web Conference*, pages 187–196, Lyon, France, 2018. IW3C2.

[72] C. Zhou and R. C. Paffenroth. Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 665–674, Halifax, NS, Canada, 2017. ACM.

[73] R. Salakhutdinov and G. Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, 2009.

[74] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 2, Anchorage, Alaska, USA, 2008. IEEE.

[75] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Advances in Neural Information Processing Systems*, pages 1753–1760, Vancouver, Canada, 2009. Curran Associates Inc.

[76] F. Zhuang, X. Cheng, P. Luo, S. J. Pan, and Q. He. Supervised representation learning: Transfer learning with deep autoencoders. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 4119–4125, Buenos Aires, Argentina, 2015. AAAI Press.

[77] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):295–307, 2016.

[78] G. Cohen, M. Hilario, H. Sax, S. Hugonnet, C. Pellegrini, and A. Geissbuhler. An application of one-class support vector machine to nosocomial infection detection. *Studies in health technology and informatics*, 107(1):716, 2004.

[79] Michael Kemmler, Erik Rodner, Petra Rösch, Jürgen Popp, and Joachim Denzler. Automatic identification of novel bacteria using raman spectroscopy and gaussian processes. *Analytica Chimica Acta*, 794:29–37, 2013.

[80] A. Iosifidis, V. Mygdalis, A. Tefas, and I. Pitas. One-class classification based on extreme learning and geometric class information. *Neural Processing Letters*, 45(2):577–592, 2017.

[81] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim. Do we need hundreds of classifiers to solve real world classification problems. *Journal of Machine Learning Research*, 15(1):3133–3181, 2014.

[82] D. Dua and C. Graff. UCI Machine Learning Repository. `http://archive.ics.uci.edu/ml`, 2017.

[83] TU Delft one-class dataset repository. `http://homepage.tudelft.nl/n9d04/occ/`, Last Accessed by 21 October 2019.

[84] Chesner Désir, Simon Bernard, Caroline Petitjean, and Laurent Heutte. One class random forests. *Pattern Recognition*, 46(12):3490–3506, 2013.

[85] C.M. Bishop. *Neural networks for pattern recognition.* Oxford university press, 1995.

[86] R.P.W. Duin. On the choice of smoothing parameters for parzen estimators of probability density functions. *IEEE Transactions on Computers*, (11):1175–1179, 1976.

[87] M.F. Jiang, S.S. Tseng, and C.M. Su. Two-phase clustering process for outliers detection. *Pattern recognition letters*, 22(6-7):691–700, 2001.

[88] D.M. Tax and R.P. Duin. Data description in subspaces. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, volume 2, pages 672–675. IEEE, 2000.

[89] E.M. Knorr, R.T. Ng, and V. Tucakov. Distance-based outliers: algorithms and applications. *The VLDB Journal—The International Journal on Very Large Data Bases*, 8(3-4):237–253, 2000.

[90] D.S. Hochbaum and D.B. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of operations research*, 10(2):180–184, 1985.

[91] P. Juszczak, D.M. Tax, E. Pe kalska, and R.P. Duin. Minimum spanning tree based one-class classifier. *Neurocomputing*, 72(7-9):1859–1869, 2009.

[92] C. C. Chang and C. J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:1–27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[93] D. M. J. Tax. Ddtools, the data description toolbox for matlab, Jan 2018. version 2.1.3.

[94] M. Reuter, N.J. Schmansky, H.D. Rosas, and B. Fischl. Within-subject template estimation for unbiased longitudinal image analysis. *NeuroImage*, 61(4):1402–1418, 2012.

[95] E. Westman, J.S. Muehlboeck, and A. Simmons. Combining MRI and CSF measures for classification of alzheimer's disease and prediction of mild cognitive impairment conversion. *NeuroImage*, 62(1):229–238, 2012.

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

[97] H-G Hwang, H-J Choi, B-I Lee, H-K Yoon, S-H Nam, and H-K Choi. Multiresolution wavelet-transformed image analysis of histological sections of breast carcinomas. *Analytical Cellular Pathology*, 27(4):237–244, 2005.

[98] V. Vapnik and A. Vashist. A new learning paradigm: Learning using privileged information. *Neural Networks*, 22(5-6):544–557, 2009.

[99] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *stat*, 1050:9, 2015.

[100] J. Kivinen, A. J. Smola, and R. C. Williamson. Online learning with kernels. *IEEE Transactions on Signal Processing*, 52(8):2165–2176, 2004.

[101] N. Liang, G. Huang, P. Saratchandran, and N. Sundararajan. A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Transactions on Neural Networks*, 17(6):1411–1423, 2006.

[102] Yeming Wen, Dustin Tran, and Jimmy Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. In *International Conference on Learning Representations*, 2020.