Eidesstattliche Versicherung

(Affidavit)

Komati. Hepsiba

Name, Vorname
(surname, first name)

229983

Matrikelnummer
(student ID number)

☐ Bachelorarbeit
(Bachelor's thesis)

■ Masterarbeit
(Master's thesis)

Titel
(Title)

Enhancing Anomaly Detection with Deep Autoencoder Network using a Custom Semi-Supervised Loss

Ich versichere hiermit an Eides statt, dass ich die vorliegende Abschlussarbeit mit dem oben genannten Titel selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

I declare in lieu of oath that I have completed the present thesis with the above-mentioned title independently and without any unauthorized assistance. I have not used any other sources or aids than the ones listed and have documented quotations and paraphrases as such. The thesis in its current or similar version has not been submitted to an auditing institution before.

Dortmund - 29.02.2024

Ort, Datum
(place, date)

Unterschrift
(signature)

Belehrung:
Wer vorsätzlich gegen eine die Täuschung über Prüfungsleistungen betreffende Regelung einer Hochschulprüfungsordnung verstößt, handelt ordnungswidrig. Die Ordnungswidrigkeit kann mit einer Geldbuße von bis zu 50.000,00 € geahndet werden. Zuständige Verwaltungsbehörde für die Verfolgung und Ahndung von Ordnungswidrigkeiten ist der Kanzler/die Kanzlerin der Technischen Universität Dortmund. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuches kann der Prüfling zudem exmatrikuliert werden. (§ 63 Abs. 5 Hochschulgesetz - HG -).

Die Abgabe einer falschen Versicherung an Eides statt wird mit Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft.

Die Technische Universität Dortmund wird ggf. elektronische Vergleichswerkzeuge (wie z.B. die Software „turnitin") zur Überprüfung von Ordnungswidrigkeiten in Prüfungsverfahren nutzen.

Die oben stehende Belehrung habe ich zur Kenntnis genommen:

Official notification:
Any person who intentionally breaches any regulation of university examination regulations relating to deception in examination performance is acting improperly. This offense can be punished with a fine of up to EUR 50,000.00. The competent administrative authority for the pursuit and prosecution of offenses of this type is the Chancellor of TU Dortmund University. In the case of multiple or other serious attempts at deception, the examinee can also be unenrolled, Section 63 (5) North Rhine-Westphalia Higher Education Act (*Hochschulgesetz, HG*).

The submission of a false affidavit will be punished with a prison sentence of up to three years or a fine.

As may be necessary, TU Dortmund University will make use of electronic plagiarism-prevention tools (e.g. the "turnitin" service) in order to monitor violations during the examination procedures.

I have taken note of the above official notification:*

Dortmund - 29.02.2024

Ort, Datum
(place, date)

Unterschrift
(signature)

***Please be aware that solely the German version of the affidavit ("Eidesstattliche Versicherung") for the Bachelor's/ Master's thesis is the official and legally binding version.**

TU Dortmund

MASTER THESIS

Enhancing Anomaly Detection with Deep Autoencoder Network using a Custom Semi-Supervised Loss

Lecturers:
Prof. Dr. Emmanuel Müller
Simon Klüttermann

Author: Hepsiba Komati

February 29, 2024

Contents

List of Acronyms

SSAD	Semi Supervised Anomaly Detection
SVDD	Support Vector Data Description
ROC	Receiver Operating Characteristic
BCE	Binary Cross Entropy
GPU	Graphics Processing Unit
CPU	Central Processing Unit
RAM	Random Access Memory
KLD	Kullback–Leibler divergence
AUC-ROC	Area Under The Receiver Operator Curve
SVM	Support Vector Machine
MLP	Multi-Layer Perceptron
ANN	Artificial Neural Network
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network
LSTM	Long Short-Term Memory
ReLU	Rectified Linear Unit
MSE	Mean Squared Error
CV	Cross Validation
TPR	True Positive Rate
FPR	False Positive Rate
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative

List of Tables

List of Figures

1 Introduction

In today's data-driven world, the ability to identify and respond to unusual patterns or anomalies within datasets is of critical importance. In data analysis, anomalies are data points that differ significantly from a dataset's expected or typical behavior [11]. These deviations can occur in various forms, from unusual spikes in medical data for the cardiovascular to typical patterns in financial transactions indicative of fraudulent activities. The timely detection and appropriate response to such anomalies can have profound implications, including the early diagnosis of critical health conditions, financial fraud prevention, and system security enhancement. The importance of anomaly detection extends across numerous domains, including healthcare, finance, space science, and industrial automation, where the consequences of failing to detect anomalies can be severe. Therefore, there is a growing need for robust and effective anomaly detection methods that can operate efficiently in diverse and dynamic data environments.

Autoencoders, as a component of deep learning models, emerge as potent tools for anomaly detection in today's data-rich environment [10][12]. Autoencoders, well-known for their adaptability and efficiency in identifying complex patterns, provide an effective solution to the anomaly detection problem. Their particular interest is effectiveness in situations where a significant amount of real-world data needs clear labels. Autoencoders can significantly enhance the model's information processing power by adding a small amount of labeled input. This sophisticated method aligns with the understanding that categorizing big datasets can be time-consuming and unfeasible in some circumstances. Therefore, this study highlights the application of deep autoencoders to advance anomaly detection techniques.The suggested strategy takes advantage of the autoencoder's ability to rebuild input data to learn and express underlying patterns that characterize normal behavior.

To overcome the underlying challenges of insufficient completely labeled data, a semi-supervised learning methodology is considered, leveraging partially labeled data to improve model performance and thereby mirroring real-world scenarios. A key focus of this study is to develop novel loss functions adapted to the complexities of anomaly identification in semi-supervised deep autoencoders.

Traditional loss functions may not adequately reflect the complexities of partially labeled datasets. To overcome this limitation, innovative loss functions are designed and implemented that carefully consider the interplay between labeled and unlabeled data. This methodology improves the model's ability to detect abnormalities reliably, hence contributing to the improvement of anomaly detection strategies in the setting of semi-supervised deep autoencoders. This strategic emphasis on loss function development distinguishes our approach and adds a layer of sophistication to the field. [8] [9]

The following sections explain all concepts, experiments, and results of the thesis. The findings from relevant studies, along with research findings and contributions to deep semi-supervised anomaly detection, are thoroughly outlined in Sections 3 to 6. Within the domain of deep semi-supervised anomaly detection, prior efforts, discoveries, and implementations are in depth addressed in Section 2, providing a comprehensive overview of the present landscape. Section 3 establishes the problem statement and investigates the theoretical foundations of anomaly detection. This includes a detailed study of the different types of anomalies, practical possibilities, and a clear justification of the essential functions that autoencoders and semi-supervised learning play. The goal of this part is to provide a solid theoretical framework for suggested anomaly detection techniques. Section 4 goes into detail about methodology and explains the strategy for semi-supervised anomaly detection. The details of the approach followed in the thesis are present here, with a focus on how to improve model performance by integrating partially labeled data and using deep autoencoders. Experiments, findings, and the introduction of datasets utilized in this study become the main topics of discussion in Section 5. An extensive analysis of performance indicators is showcased, offering valuable perspectives on the effectiveness of suggested techniques. The main conclusions of thesis are clearly summarized in Section 6. Additionally, it outlines potential avenues for future work in the dynamic and evolving field of deep semi-supervised anomaly detection. This comprehensive structure ensures a thorough examination of the problem, methodology, experimental outcomes, and the broader implications of our research, contributing meaningfully to the advancement of anomaly detection methodologies.

2 Research work

Within the field of anomaly detection, this study explores a variety of methodologies, from supervised to unsupervised. However, a noticeable gap exists in the existing research, particularly concerning the utilization of deep autoencoder approaches. Motivated by this fact, this research adds contribution to this understudied subject by overcoming the challenges in anomaly recognition using semi-supervised deep auto encoders.

"Self-Supervised Anomaly Detection" paper introduce a method called self-supervised learning [17], which uses a large amount of unlabeled data to understand regular behaviors effectively. This approach is not only scalable and cost-effective but also proves to be a valuable solution for anomaly detection [17]. The paper provides a detailed overview of current methods, highlighting the strengths and weaknesses of self-supervised anomaly detection algorithms compared to other advanced models. This foundational work not only recognizes the challenges posed by anomalies, like their rarity and uneven representation but also lays the groundwork for incorporating self-supervised learning to overcome these challenges. In the context of our research, [17] insights guide and shape our exploration of advanced anomaly detection methods using deep autoencoders, aligning with the overall progress in the field.

Lukas Ruff's paper titled "Deep Semi-Supervised Anomaly Detection, [18]" stands out as a notable contribution. This paper introduces a comprehensive deep learning approach for semi-supervised anomaly detection, filling a significant gap in unsupervised Deep SVDD [9]. Ruff presents "Deep SAD," a methodology designed to handle the complexities of semi-supervised anomaly detection [18]. The work acknowledges a common scenario where, in addition to a substantial number of samples lacking labels, there is availability of a limited pool of labeled samples. Ruff strategically incorporates both normal and anomalous labeled samples into the model [18]. Moreover, the paper introduces an information-theoretic framework for deep anomaly detection, providing a theoretical foundation for the proposed methodology. In the context of our research, Ruff's investigation offers valuable insights into the landscape of deep semi-supervised anomaly detection. This helps our study to develop a novel semi-supervised loss function for deep auto encoders, addressing a crucial yet unexplored aspect in the field.

Understanding the landscape of anomaly detection algorithms and their comparative effectiveness has been substantially aided by recent benchmark initiatives Anomaly De-

tection Benchmark [19]. This research plays a crucial role in addressing key questions related to anomaly detection algorithms, covering various levels of supervision, different anomaly types, and challenges posed by noisy and corrupted data. Remarkably, it's known for its thoroughness, standing out in anomaly detection evaluations by testing 30 algorithms across a diverse set of 57 benchmark datasets. The paper sheds light on the limitations of traditional unsupervised methods in capturing nuanced anomalies while emphasizing the potential of semi-supervised methods, particularly with a modest amount of supervision[19]. Through a systematic exploration of these algorithms under diverse conditions, the paper provides valuable insights into the roles of supervision, anomaly types, and algorithm robustness. Building upon these findings, our research aims to leverage these insights to develop advanced models that surpass the existing state-of-the-art in anomaly detection.

To enhance understanding of autoencoder based anomaly detection, we turn to the research conducted by Hasan Torabi [32], titled "Practical auto encoder-based anomaly detection by using vector reconstruction error." This study introduces a novel approach tailored for anomaly detection within cloud computing networks. Their methodology presents a new paradigm in auto encoder usage, wherein they generate a vector reconstruction error for each feature, departing from conventional methods that compute a singular value. This innovation allows for a more granular assessment of anomalies, resulting in enhanced detection accuracy. Notably, the proposed autoencoder network is streamlined with just one hidden layer, thereby ensuring computational efficiency compared to existing approaches. Furthermore, [32] conducts comprehensive evaluations, leveraging the CIDDS-001 dataset, a widely accepted benchmark in the field. Through meticulous comparison with existing methods, they demonstrate notable improvements in key performance metrics such as accuracy, recall, false-positive rate, and F1-score, underscoring the efficacy of their proposed approach [32].

Our research embarks on an extensive investigation into anomaly detection methodologies, with a particular focus on leveraging semi-supervised learning and auto encoder-based techniques. The primary aim is to overcome the limitations inherent in traditional methods and existing approaches to anomaly detection. This involves development of a novel semi-supervised learning approach tailored explicitly to deep auto encoder networks. By meticulously examining and comparing established methodologies, including those proposed by Hadi Hojjati [17] and Lukas Ruff [18] , we aim to deepen the understanding of anomaly detection methodologies within the field. Additionally, the work of Torabi [32]. underscores the versatility and efficacy of auto encoder-based techniques in

capturing intricate data patterns, making them well-suited for anomaly detection tasks. Drawing inspiration from these studies, we envision the potential of integrating semi-supervised learning with auto encoder-based techniques to establish a robust anomaly detection framework, as explained in subsequent sections. Furthermore, investigation extends to the evaluation of various loss functions to determine optimal performance, with a specific emphasis on addressing the unique characteristics of anomalies and enhancing the reliability of our proposed framework.

3 Problem Statement and Theoretical Understanding

This chapter serves as a foundational analysis of anomaly detection concepts for this thesis. It looks into the fundamentals of anomaly detection, providing an overview of basic definitions, important terminology, and common deep learning and semi-supervised algorithms in this domain.

3.1 Problem Statement

Over the years, various techniques and methodologies have been developed to address the challenge of anomaly detection. Traditional methods like z-score analysis and Gaussian models face limitations in capturing intricate, non-linear patterns within high-dimensional datasets[13]. This inadequacy hampers their effectiveness in modern applications. Supervised machine learning approaches, reliant on labeled data, prove valuable but encounter challenges in obtaining costly and rare labeled anomalies, constraining their practical utility. The shortcomings of traditional and supervised methods have led to a growing interest in unsupervised anomaly detection techniques. These methods, operating without labeled data, demonstrate particular efficacy in scenarios where labeled anomalies are scarce. They leverage inherent patterns within the data to identify anomalies, offering flexibility in dynamic and evolving data environments. [18]

However, unsupervised methods introduce challenges such as potential false positives and the need for careful parameter tuning. The absence of labeled data further complicates effective validation and fine-tuning. This motivates the exploration of semi-supervised anomaly detection methods, which strike a balance by utilizing a combination of labeled and unlabeled data for training. This hybrid approach maximizes the advantages of unsupervised learning while incorporating valuable information from labeled instances

[18]. Strategic utilization of labeled data in semi-supervised methods elevates precision, mitigates false positives, and enhances overall performance.

Semi-supervised learning emerges as a possible solution to the lack of labeled anomaly data[18]. This approach makes use of both labeled normal data and unlabeled data, allowing models to learn from a large amount of normal data while detecting anomalies as deviations from the norm. Despite success in anomaly detection tasks, there is opportunity for development in semi-supervised methods. Traditional approaches frequently apply standard loss functions that treat anomalies and regular data identically, potentially resulting to unsatisfactory performance [18]. This study considers deep auto encoder networks and propose a unique strategy to improve anomaly identification by developing a custom semi-supervised loss function suited to anomalous characteristics. The aim in integrating deep auto encoder and semi-supervised approaches is to dramatically increase the accuracy and reliability of anomaly detection across a wide range of applications. This novel methodology aims to close existing gaps and contribute to the improvement of semi-supervised anomaly identification methods.

Apart from the approach detailed in this study, insights from two other established methodologies, Deep SAD (Semi-Supervised Anomaly Detection) and self-supervised learning, are also applied. This triangulation enables a thorough comparison and examination, aiming to grasp the complex benefits and drawbacks of each technique [18]. By combining these methodologies, the objective is to significantly improve accuracy, reliability, and applicability in the realm of anomaly detection across diverse applications. This innovative approach aims to bridge existing gaps and contribute to the advancement of semi-supervised anomaly detection methodologies.

3.2 Theoretical Understanding

3.2.1 About Anomalies

Anomalies are data patterns that do not correspond to the concept of normal behavior. [3] In practice, anomalies appear as data points that contradict established standards and deviate significantly from what is considered typical or standard behavior. These anomalies often arise from distinct underlying processes, introducing unique functions or distributions that differentiate them from the rest of the data.

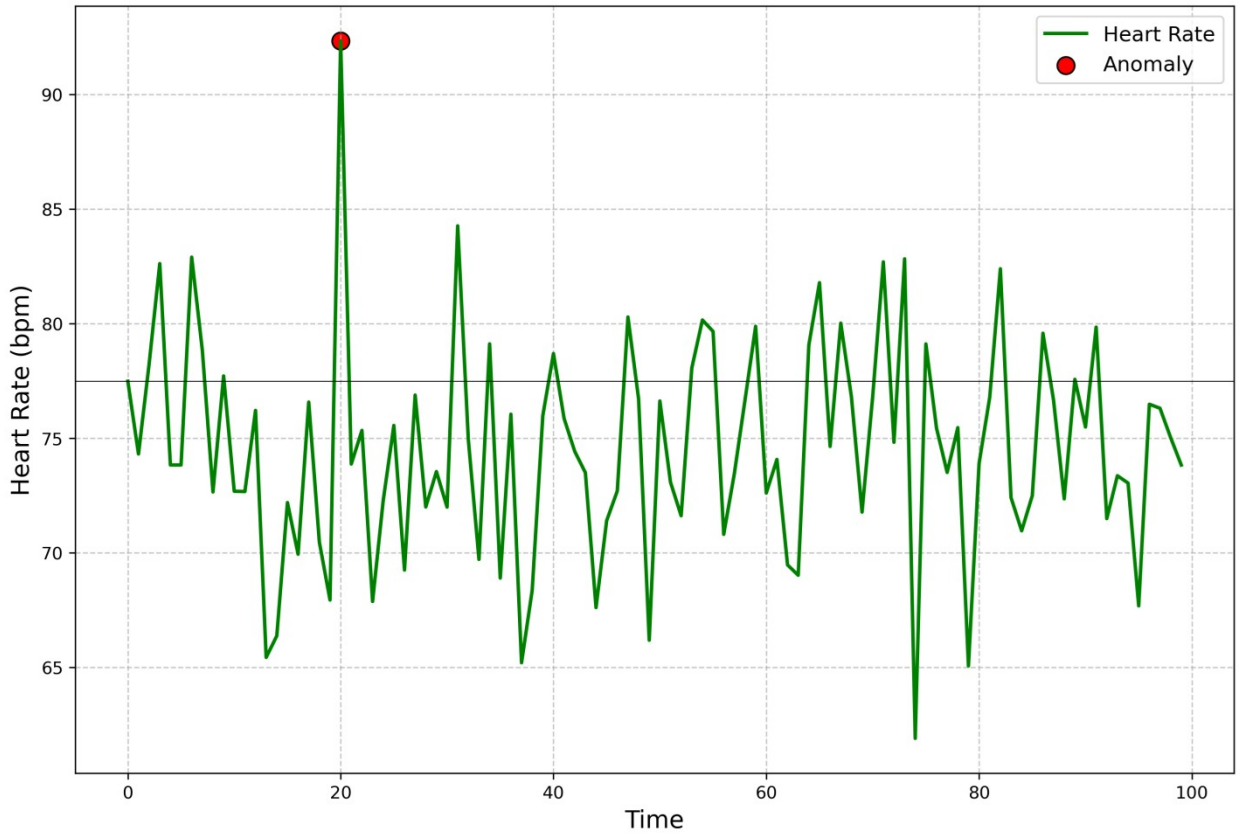

Figure 1: Anomaly Detection in Heart Rate Monitoring

Let $X = \{x_i\}, i = 1, \ldots, N, x \in \mathbb{R}^k$, represent the set of input data points. The goal is to learn a scoring function $h(x)$ such that, data points can be classified as a normal ($y_i = 0$) or a anomaly ($y_i = 1$) based on a threshold λ.

$$y_i = \begin{cases} 0, & \text{if } h(x_i) < \lambda \\ 1, & \text{if } h(x_i) \geq \lambda \end{cases}$$

To illustrate, consider a scenario in healthcare where heart rate data is monitored over time. In Figure 1, the green line represents the heart rate pattern, displaying expected fluctuations over time. However, at a specific point (highlighted in red), an anomaly is introduced a deviation from the anticipated pattern.
This anomaly, simulated by an offset in heart rate, symbolizes irregularities that could have critical implications in real-world scenarios.

3.2.2 Types of Anomalies

In the subsequent discussion, three distinct types of anomalies are explained: Point, Contextual, and Collective, each shedding light on varied deviations.

1. **Point Anomaly:** This occurs when a solitary data point markedly strays from the overall heart rate pattern. In Figure 1, the point anomaly is vividly highlighted in red, representing a moment where the heart rate experiences a sudden and unexpected increase, significantly deviating from the neighboring data points.

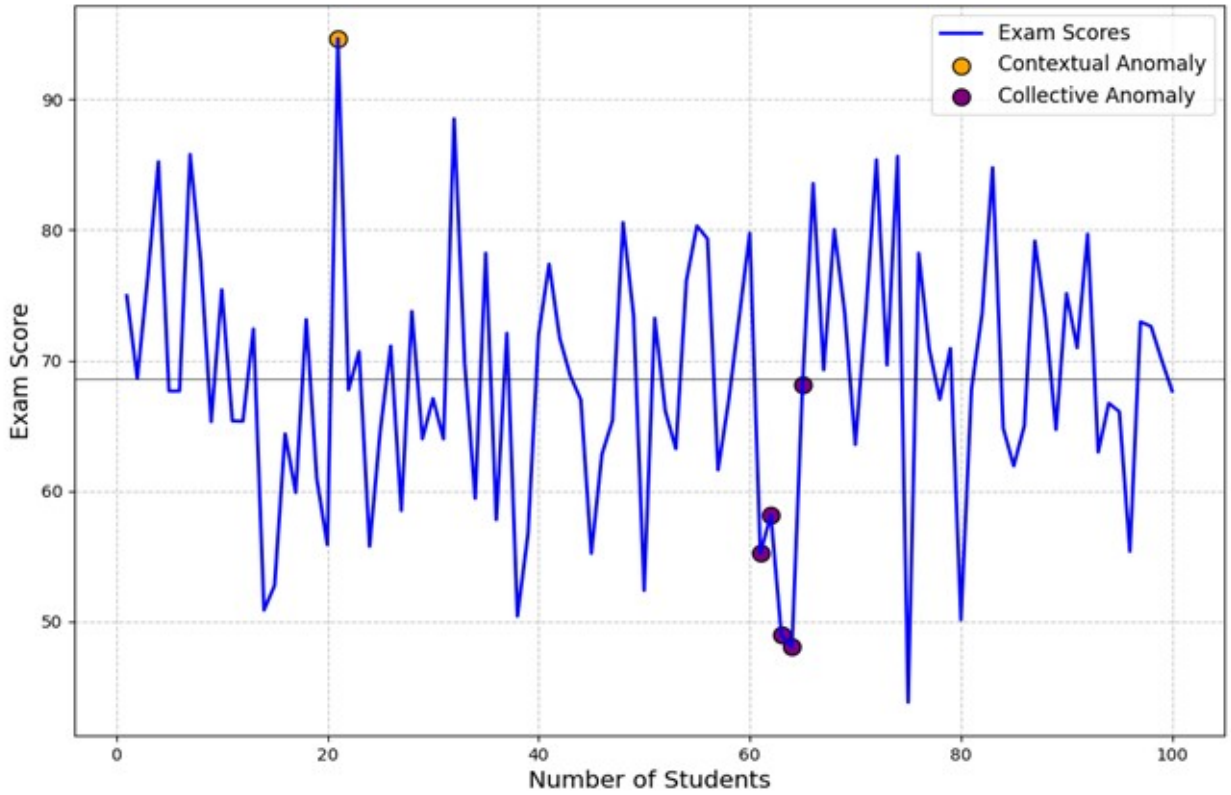

Figure 2: Different types of Anomalies

2. **Contextual Anomaly:** These anomalies arise when a data point exhibits abnormal behavior within a specific context or subset. As illustrated in Figure 2 and marked by the orange indicator, simulate a sudden increase in an individual's exam score. This showcases how anomalies can be context-dependent, manifesting uniquely in different scenarios. The orange marker identifies a contextual anomaly, emphasizing instances where individual student scores deviate significantly from the norm.

3. **Collective Anomaly:** Also referred to as group anomalies, these anomalies occur when a collection of data points collectively displays abnormal behavior, even if individual points may seem normal. In Figure 2, this could be represented by a series of consecutive anomalies indicating a coordinated deviation in exam scores. The purple markers in the plot signify such collective anomalies, suggesting a potential shared characteristic among a group of students. This scenario could indicate systemic issues or external influences affecting a specific subset of students.

3.2.3 Methods to Detect Anomalies

The anomaly detection method used is determined on the properties of the data being considered. In univariate datasets with a single feature, anomalies are identified by examining deviations from the norm in that single characteristic. Multivariate datasets, such as those used in this thesis, require more complex approaches to detect anomalies across multiple dimensions at the same time. The nature of the features in the dataset is critical; some algorithms can handle both categorical and continuous characteristics, while others specialize in one. The features in the dataset strongly influence the choice of a suitable anomaly detection technique. Alternatively, adjusting the data to align with a certain anomaly detection tool is a possible option for assuring compatibility between the data format and the method used. This involves modifying the dataset to fit the assumptions and needs of the chosen anomaly detection method. For example, it might involve changing features, encoding categorical variables, scaling numerical values, or making other changes to ensure that the dataset meets the expectations of the chosen anomaly detection tool. This proactive strategy not only broadens the scope of relevant anomaly detection methods, but it also improves the overall performance of the chosen algorithm by adapting the data to its specific requirements.[1]

As illustrated in Figure 3, provide a detailed view on the many techniques in this subject. The parts that follow go over particular anomaly detection approaches, each customized to different data circumstances and requirements.

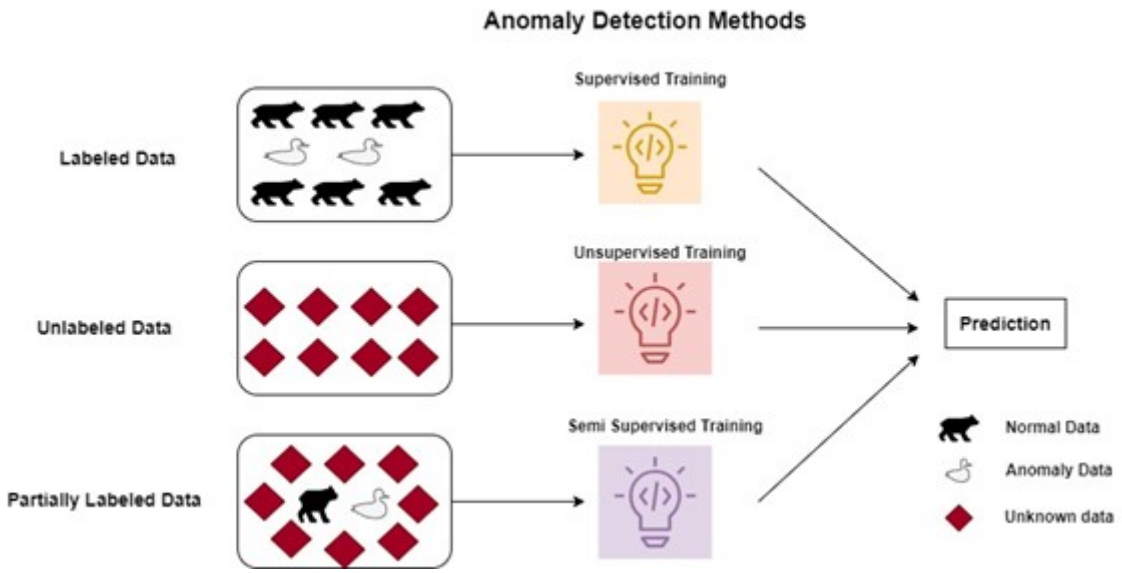

Figure 3: Different anomaly detection modes based on data availability

1. **Supervised Anomaly Detection:** This method necessitates labeled datasets wherein anomalies are explicitly identified during the training phase. These algorithms learn to differentiate between normal and anomalous instances based on the provided labels. In the case of Support Vector Machines (SVM), the objective is to establish a classification hyperplane as a decision surface to maximize the margin between positive and negative instances [4]. While other supervised approaches, such as decision trees, perform exceptionally well when clear anomaly labels are available for training, they may exhibit diminished effectiveness in scenarios where obtaining labeled anomaly data is either scarce or impractical.

2. **Unsupervised Anomaly Detection:** Operating without the need for labeled data, this approach is well-suited for scenarios in which obtaining instances with explicitly marked anomalies poses challenges. Algorithms such as K-Means clustering or isolation forests identify anomalies by discerning patterns that deviate from the majority. Unsupervised methods prove particularly beneficial when anomalies are not clearly defined or are anticipated to evolve over time. The underlying concept is that an unsupervised anomaly detection algorithm assigns scores to data solely based on intrinsic properties of the dataset [1]. However, a notable disadvantage is that known anomalies may be overlooked, resulting in performance lower than that of supervised anomaly detection methods.

3. **Semi-Supervised Anomaly Detection:** It combines aspects of both supervised and unsupervised techniques, leveraging a small amount of labeled data along with a larger pool of unlabeled instances. This hybrid approach enhances precision by incorporating the valuable information provided by labeled instances while maintaining adaptability to scenarios where obtaining comprehensive labeled data is impractical [18]. Semi-supervised methods are effective when a limited amount of labeled anomaly data is available, striking a balance between the advantages of both supervised and unsupervised approaches.

3.2.4 Deep Learning in Anomaly Detection

Deep Learning has emerged as a powerful paradigm for understanding complex data, with the ability to identify hidden temporal connections and automatically derive high-level representations [10]. Specifically customized designs, such as Artificial Neural Networks (ANNs), Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs), including Long Short-Term Memory (LSTM) networks, have demonstrated significant success in many deep learning tasks [16]. This major development lays the path for further advances in deep learning approaches, resulting in an ever-expanding toolbox for anomaly identification methodologies.

The Multi-Layer Perceptron (MLP) is a basic but powerful deep learning architecture. Figure 4 illustrates the MLP as a fully-connected feed-forward network comprising an input layer, an output layer, and potentially multiple hidden layers, each containing neurons. Each neuron computes a weighted sum of the input and often employs an activation function to enable non-linear computations. Incorporating bias terms produces a desirable shift in this result by adding parameters that are independent of the present input. The connection weights between neurons are optimized by minimizing a loss function that measures the variance between input and output, finally learning an appropriate transformation. Moving on to the next section, Auto encoder architecture, a distinguished member of deep learning methodologies are detailed. [15]

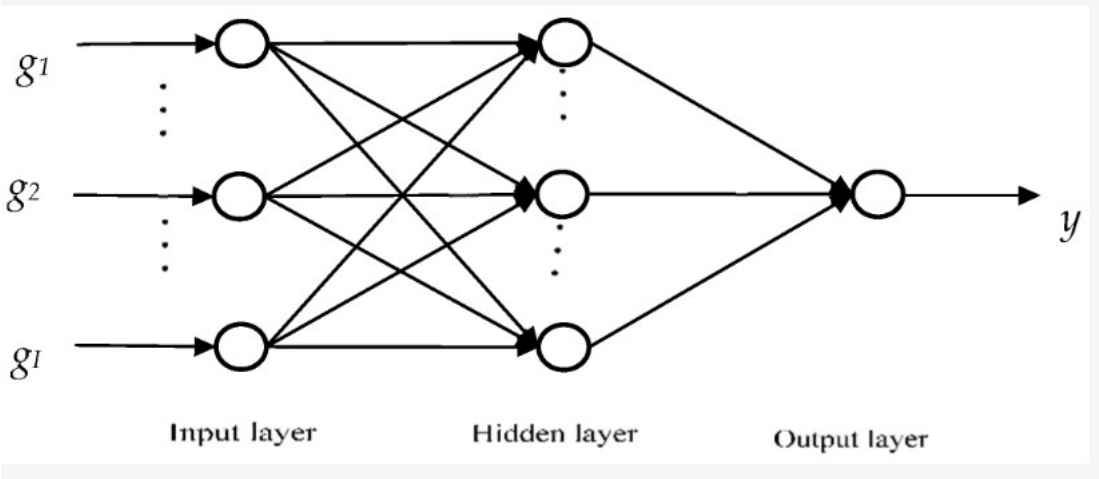

Figure 4: Multi-Layer Perceptron (MLP) Architecture from [15]

3.2.5 Autoencoder: Bridging Traditional Gaps

In the field of anomaly detection, classical machine learning (ML) and statistical techniques have been widely used. These methods rely on basic statistical measures like mean, standard deviation, and percentiles. However, they often struggle to handle the complexities found in high-dimensional datasets, especially when dealing with non-linear relationships and intricate patterns [13]. Recognizing these limitations, there's been a growing interest in using deep learning techniques for anomaly detection. Among these techniques, autoencoders have shown promise in overcoming the drawbacks of traditional methods. By utilizing neural networks, autoencoders can uncover latent representations and extract meaningful features from data, leading to more accurate anomaly detection

3.2.6 What are Autoencoders

Autoencoders, a type of neural network, operate on the principle of unsupervised learning and have proven instrumental in anomaly detection. The primary aim of an autoencoder is to grasp a concise, informative representation of input data, referred to as the latent space[9]. This is accomplished through two key components: the encoder, tasked with mapping input data to a lower-dimensional space, and the decoder, responsible for rebuilding the input data from this compressed representation.

Autoencoder's inherent capability to capture essential data features renders them invaluable for identifying anomalies. This architecture, crucial in deep learning, operates on the principle of efficient data compression and reconstruction. To understand the mechanics of this process, let's construct a mathematical basis and go into the structural components indicated in Figure 5.

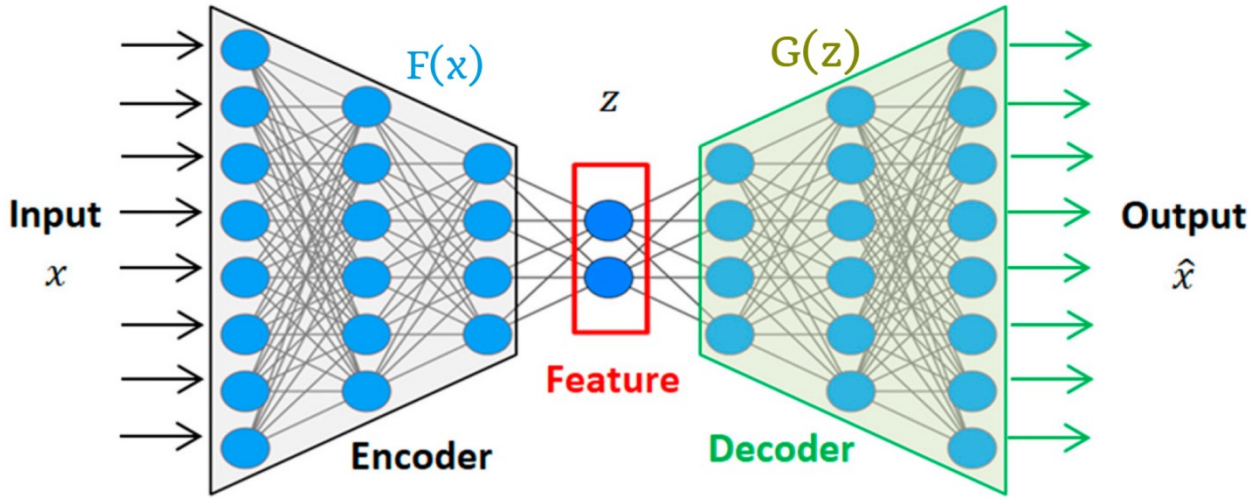

Figure 5: General structure of auto encoder from [2]

In mathematical terms, the encoding process can be written as $Z = f(X)$, where f is the encoder function, and the decoding process as $X = g(Z)$, where g is the decoder function. Furthermore, the reconstruction error (L) is an important statistic for quantifying the difference between the input and the reconstructed output. It is determined using a suitable loss function, such as Mean Squared Error (MSE). The MSE of the input X and the output X_i' is defined as [2]

$$L\left(X, X'\right) = \frac{1}{n}\sum_{i=1}^{n}\left(X_i - X_i'\right)^2,$$

The Autoencoder comprises three main components:

1. **Encoder:** Positioned in black on the left side of Figure 5, the encoder operates as a fully connected feed-forward network. It takes the input X_i and transforms it into a lower-dimensional representation, denoted as the latent space Z. This transformation occurs through a sequence of fully connected layers, providing flexibility in determining the layers and neurons in this crucial component.

2. **Latent Space:** Positioned centrally within Figure 5, the latent space Z (depicted as $h(X_i)$)) act as a condensed representation of the input data. The number of neurons in this layer can be specified, and it encapsulates the most essential features necessary for reconstruction. This single layer efficiently encapsulates the essence of the data.

3. **Decoder:** The decoder, highlighted in green on the right side of Figure 5, undertakes the crucial task of reconstructing the original input (X) from the encoded representation (Z). Employing the output of the encoder, the decoder attempts to reverse the encoding process. Like the encoder, the decoder comprises fully connected layers, and the structure mirrors that of the encoder in regards to layers and neurons.

   In essence, the Autoencoder is trained to minimize the reconstruction error by adjusting its weights during the learning process. This optimization process compels the network to leverage the inherent structure within the dataset, creating an efficient lower-dimensional latent space. The interplay between the encoder, latent space, and decoder forms the backbone of the Autoencoder's ability to compress and reconstruct data effectively. In the upcoming section, specific autoencoder architecture adopted for this thesis is explained.

4 Methodology

In this section, an investigation of key tools and approaches that shape anomaly detection methodology is discussed. In the present study, the autoencoder architecture, and its working is explained. This review includes a look at the various loss functions utilized to evaluate anomaly detection models, as well as the evaluation metrics that guide performance assessment. Furthermore, the outline of the preprocessing techniques used to achieve robust and relevant results are discussed.

4.1 Architectural Insight

Within this anomaly detection study, the autoencoder architecture is designed for optimal performance. The encoder consists of three dense layers, each comprising 64, 32, and 16 neurons, respectively, utilizing rectified linear unit (ReLU) activation functions. Acting as a bottleneck, a layer with 12 neurons and ReLU activation effectively compresses input features. In a symmetrical fashion, the decoder reconstructs the encoded information using two dense layers with 16 and 32 neurons, and an output layer with a variable number of neurons tailored to the input data dimension. Both encoder and decoder layers utilize ReLU activation, while the output layer employs a sigmoid activation. To prevent overfitting and enhance generalization during training, a dropout layer with a 30% dropout rate is strategically positioned between the latent space and decoder layers. However, despite its incorporation, this dropout layer did not yield noticeable improvements to the model's performance. Figure 6 visually captures the overarching structure of the autoencoder, providing a clear representation of its design and functionality in our work.

Layer Type	Activation Function	Number of neurons
Input	-	(input dimension)
Encoder	Relu	$(64, 32, 16)$
Latent Space	Relu	(12)
Decoder	Relu	$(16, 32, 64)$
Output	Sigmoid	(output dimension)

Table 2: Autoencoder parameter details

In addition to the baseline model, Deep SAD (Semi-Supervised Anomaly Detection) approach is implemented, aligning its structure closely with the original auto encoder. Simultaneously, a self-supervised learning variant is introduced, augmenting the architecture with unique transformation functions. These functions, namely Gaussian noise and Random mask (from a uniform distribution), serve as pivotal components in augmenting the auto encoder's robustness to diverse anomaly detection paradigms. For further details on the parameters utilized in the Self-Supervised Auto encoder, refer to Table 3

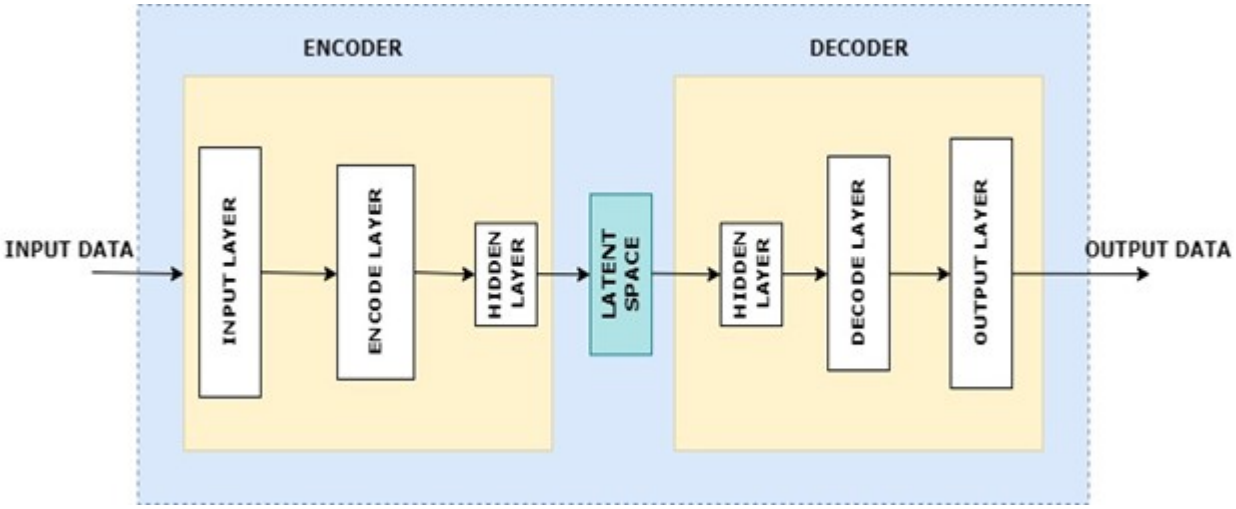

Figure 6: Autoencoder Architecture Used in this Thesis

4.1.1 Gaussian Noise Transformation

The Gaussian noise transformation function injects random noise into the data, simulating noisy real-world conditions. Mathematically, this transformation can be represented as:

$$x_{\text{noisy}} = x_{\text{clean}} + \mathcal{N}\left(0, \sigma^2\right),$$

Where:

- x_{noisy} represents the noisy version of the input data.
- x_{clean} represents the clean input data.
- $\mathcal{N}(0, \sigma^2)$ represents random noise drawn from a Gaussian distribution with mean 0 and standard deviation σ.

4.1.2 Random Mask Transformation

The random mask transformation introduces a random mask to cultivate resilient representations within the model. Mathematically, this transformation can be represented as:

$$x_{\text{masked}} = x_{\text{clean}} \odot M,$$

Where:

- x_{masked} represents the masked version of the input data.
- x_{clean} represents the clean input data.
- M represents a random mask matrix drawn from a uniform distribution.
- $\odot$ represents element-wise multiplication.

Layer Type	Activation Function	Number of neurons
Input	-	(input dimension)
Transformation	-	-
Hidden Layer	Relu	$(64, 32, 16)$
Output	Sigmoid	(output dimension)

Table 3: Self Supervised model parameter details

As depicted in Table 3, noise component is incorporated into the self supervised model architecture, a pivotal enhancement that elevates its performance beyond conventional anomaly detection methods. This augmentation enables the model framework to adapt more easily to diverse and dynamic data environments, resulting in superior anomaly detection capabilities. By integrating this noise component, the model not only identifies anomalies more effectively but also generates robust data representations that capture the underlying structure of the data. Furthermore, this integration facilitates direct comparisons within the same autoencoder architecture, thereby facilitating comparative studies. Figure 7 provides a visual representation of the enhanced model, offering a comprehensive overview of its design and operational functions. Notably, a unique transformation after the input layer is introduced, a distinctive aspect of our method. The encoder comprises one dense layer with 64, 32, and 16 neurons, respectively, utilizing rectified linear unit (ReLU) activation functions. The output layer employs a sigmoid activation function. This architecture consists of three main components: the input layer, transformation layer, and hidden layer, complemented by a custom loss function.

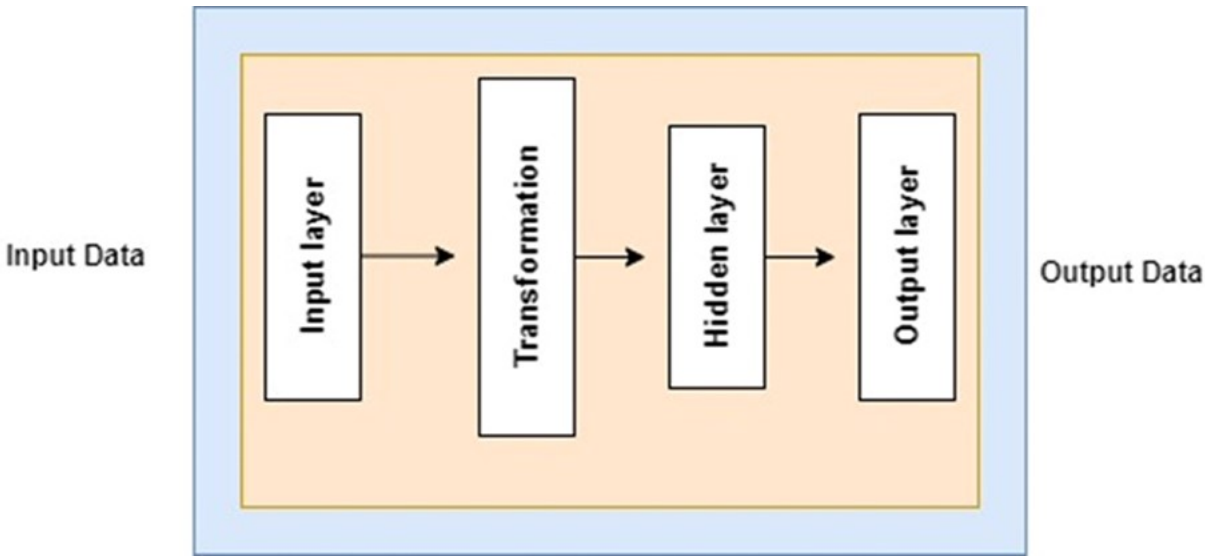

Figure 7: Self Supervised model Architecture

4.1.3 Autoencoder Parameters and Optimization

The configuration of autoencoders demands careful consideration of several key parameters, each playing a pivotal role in determining the model's performance. In this section, parameters crucial to this thesis are outlined.

1. **Activation Function:** A neural network's architecture is strongly reliant on its activation functions, which influence how well the model learns from the training dataset. The activation function used for the hidden layer has a considerable impact on how weighted input is transformed into output inside a network layer. In this study after careful consideration, rectified linear unit (ReLU) activation functions for both encoders and decoders is used. [6]
2. **Reconstruction Loss:** The type of input and output desired for the autoencoder significantly influences the choice of the loss function used during training. After experimenting with various alternatives, Mean Squared Error (MSE) Loss is considered. This choice proved effective in guiding the autoencoder to adapt to the desired input-output relationships.
3. **Loss Functions:** The selection of a suitable loss function is a critical hyperparameter when tuning autoencoders. In this study, given the application of semi-supervised learning, conventional loss functions were not applicable. Therefore, custom loss functions were created, a detailed explanation of which will be provided in the upcoming sections. These custom loss functions were designed to

align with the unique characteristics of semi-supervised autoencoders employed in this research [7].

4. **Learning Rate:** The learning rate is of utmost significance in optimizing the training process [5]. In this experiments, optimal performance with an Adam optimizer set at a learning rate of 0.01.

5. **Regularization:** To mitigate the risk of over fitting, a regularization strategy is employed. In this case, early stopping is utilized to prevent the model from excessively learning the training data, thereby enhancing generalization capabilities.

4.2 Loss Functions

The loss function is a crucial metric that measures the disparity between the original input data and the output reconstructed by the autoencoder. Throughout the training phase, adjustments are made to the autoencoder to minimize this difference. The choice of a loss function has significant implications for the autoencoder's ability to detect anomalies and reconstruct normal data. These functions are critical in training the autoencoder to efficiently encode and reconstruct normal instances while accurately detecting anomalies or deviations from the norm.

In the context of semi-supervised learning, the challenge becomes more complicated. Tailored techniques are required because to the dataset's peculiarities, which include both labeled (normal and anomalous) and unlabeled occurrences. Traditional loss functions might not adequately address the complexity of these datasets. Therefore, the specialized loss functions presented here are specifically tailored to accommodate semi-supervised environments. Each function is a purposeful adaptation that improves the autoencoder's ability to encode and reconstruct typical instances with precision while detecting anomalies.

The choice of an appropriate loss function becomes critical, determining the autoencoder's capacity to navigate the complexities of semi-supervised learning scenarios. These customized loss functions aim to provide the model with enhanced insight into differentiating between typical and abnormal cases, especially when dealing with unlabeled data. Additionally, incorporating techniques such as sample weighting, Rényi entropy, and boosted learning enhances the model's adaptability, rendering it suitable for real-world settings with varying degrees of supervision. In essence, the selection and design of these unique loss functions emerge as a strategic reaction to the problems of

semi-supervised anomaly detection, demonstrating a smart and adaptive approach to anomaly identification across varied datasets.

4.2.1 Semi-Supervised Hinge Loss

The Semi-Supervised Hinge Loss introduces a hinge loss, commonly used in support vector machines, to penalize deviations from the correct classification boundary. By separating the loss calculations for normal, anomaly, and unlabeled instances, the model can adapt differently to each category, enhancing its discrimination capabilities.

$$\text{Loss} = \frac{1}{N_{\text{unlabeled}}} \sum_{i=1}^{N_{\text{unlabeled}}} \left(\max\left(1 - y_{\text{true},i} \cdot y_{\text{pred},i}, 0\right)\right)^2 + \frac{1}{N_{\text{normal}}} \sum_{i=1}^{N_{\text{normal}}} \left(\max\left(1 - y_{\text{true},i} \cdot y_{\text{pred},i}, 0\right)\right)^2 - \frac{1}{N_{\text{anomaly}}} \sum_{i=1}^{N_{\text{anomaly}}} \left(\max(1 - y_{\text{true},i} \cdot y_{\text{pred},i}, 0)\right)^2,$$

Where:

- $N_{\text{unlabeled}}$ is the number of unlabeled instances,
- N_{normal} is the number of normal labeled instances,
- N_{anomaly} is the number of anomaly labeled instances,
- $y_{\text{true},i}$ is the true label for instance i,
- $y_{\text{pred},i}$ is the predicted label for instance i.

The following pseudo-code outlines the procedural steps involved in computing the Semi-Supervised Hinge Loss.

Input:

- Y_{true} : True labels
- Y_{pred} : Predicted labels

Output:

Total loss

Algorithm 1 Calculate Hinge Loss

1: Initialize hinge loss for anomalies, unlabeled, and normal points to zero
2: for each data point in the batch do
    2.1: Calculate hinge loss based on true labels and predictions
    2.2: if data point is labeled as anomaly then
        2.2.1: Accumulate hinge loss for anomaly
    2.3: else if data point is labeled as unlabeled then
        2.3.1: Accumulate hinge loss for unlabeled
    2.4: else if data point is labeled as normal then
        2.4.1: Accumulate hinge loss for normal
3: end for
4: Calculate mean squared loss for each category
5: return a weighted combination of losses.

4.2.2 Generalized Rényi Entropy Loss

The Generalized Rényi Entropy Loss incorporates a non-linear entropy-based measure to capture the uncertainty of predicted probabilities. This loss is particularly adept at handling the challenges posed by anomalies and unlabeled instances, providing a holistic evaluation of the model's predictive capabilities. After thorough deliberation on the most effective approach to introduce this loss function, we discovered that leveraging anomaly and unlabeled data autonomously reduces the overall loss, hence prompting our adoption of this strategy.

$$\text{Loss} = -\frac{1}{N_{\text{anomaly}}} \sum_{i=1}^{N_{\text{anomaly}}} \frac{1}{1-Q} \log \left(\sum_{c=1}^{C} \left(p_{\text{anomaly },i,c} \right)^{Q} \right) + \frac{1}{N_{\text{unlabeled}}} \sum_{i=1}^{N_{\text{unlabeled}}} \frac{1}{1-Q} \log \left(\sum_{c=1}^{C} \left(p_{\text{unlabeled },i,c} \right)^{Q} \right),$$

Where:

- N_{anomaly} is the number of anomaly-labeled instances,
- $N_{\text{unlabeled}}$ is the number of unlabeled instances,
- C is the number of classes,
- Q is the order of the Rényi entropy,
- $p_{\text{anomaly },i,c}$ is the predicted probability of class c for instance i in the anomaly-labeled category,

- $p_{\text{unlabeled },i,c}$ is the predicted probability of class c for instance i in the unlabeled category.

The following pseudo-code outlines the procedural steps involved in computing the Rényi Entropy Loss.

Input:

- Y_{true} : True labels
- Y_{pred} : Predicted labels
- Q : Rényi Entropy parameter

Output:

Total loss

Algorithm 2 Custom Generalized Rényi Entropy Loss

1: Initialize Rényi Entropy Loss for anomaly and unlabeled points to zero
2: for each data point in the batch do
  2.1: Separate true labels and softmax predictions for the current data point
  2.2: Categorize data points into anomaly and unlabeled based on their labels
  2.3: Calculate Rényi entropy for anomaly points
  2.4: Calculate Rényi entropy for unlabeled points
3: end for
4: Sum up the Rényi entropy losses for anomaly and unlabeled points
5: return the total generalized Rényi entropy loss

4.2.3 Custom Boosted Semi-Supervised Loss

The Custom Boosted Semi-Supervised Loss incorporates sample weighting into the hinge loss, enabling a focus on challenging instances within the model. This approach replicates a boosting-like mechanism by giving various weights to samples based on their difficulty, improving the model's performance in complex settings.

$$\begin{aligned}\text{Loss } = & \frac{1}{N_{\text{normal}}} \sum_{i=1}^{N_{\text{normal}}} \exp\left(-\alpha \cdot \text{ hinge }_{\text{normal },i}\right) \cdot \text{ hinge }_{\text{normal },i} + \\ & \frac{1}{N_{\text{anomaly}}} \sum_{i=1}^{N_{\text{anomaly}}} \exp\left(-\alpha \cdot \text{ hinge }_{\text{anomaly },i}\right) \cdot \text{ hinge }_{\text{anomaly },i} + \\ & \frac{1}{N_{\text{unlabeled}}} \sum_{i=1}^{N_{\text{unlabeled}}} \exp\left(-\alpha \cdot \text{ hinge }_{\text{unlabeled },i}\right) \cdot \text{ hinge }_{\text{unlabeled },i}\end{aligned}$$

Where:

- N_{normal} is the number of normal-labeled instances,
- N_{anomaly} is the number of anomaly-labeled instances,
- $N_{\text{unlabeled}}$ is the number of unlabeled instances,
- α is the weight parameter for unlabeled samples,
- hinge $_{\text{normal},i}$ is the hinge loss for instance i in the normal-labeled category,
- hinge $_{\text{anomaly},i}$ is the hinge loss for instance i in the anomaly-labeled category,
- hinge $_{\text{unlabeled},i}$ is the hinge loss for instance i in the unlabeled category.

The following pseudo-code outlines the procedural steps involved in computing the custom boosted semi-supervised loss

Input:

- Y_{true} : True labels
- Y_{pred} : Predicted labels

Output:

Total loss

Algorithm 3 Custom Boosted Semi Supervised Loss

1: Initialize hinge loss for anomaly and unlabeled points to zero.
2: for each data point in the batch do
    2.1: Calculate hinge loss based on true labels and predictions
    2.2: Categorize data points into normal, anomaly, and unlabeled based on labels
    2.3: Assign weights to samples based on difficulty (boosting-like)
    2.4: Adjust the loss calculation with weights for each category
        2.4.1: Calculate loss for normal points with boosted weights
        2.4.2: Calculate loss for anomaly points with boosted weights
        2.4.3: Calculate loss for unlabeled points with boosted weights
3: end for
4: Combine the losses for normal, anomaly, and unlabeled points
5: return the boosted semi-supervised loss

4.2.4 Semi-Supervised Cross-Entropy Loss

The Semi-Supervised Cross-Entropy Loss extends the traditional cross-entropy loss to accommodate unlabeled instances. By computing losses separately for normal, anomaly,

and unlabeled instances, this loss function ensures that the model is adept at handling all categories, striking a balance between supervised and unsupervised learning.

$$LOSS = -\sum_{i=1}^{n_{\text{normal}}} y_{\text{true},i} \log\left(y_{\text{pred},i}\right) - \sum_{j=1}^{n_{\text{anomaly}}} y_{\text{true},j} \log\left(y_{\text{pred},j}\right) - \sum_{k=1}^{n_{\text{unlabled}}} y_{\text{true},k} \log\left(y_{\text{pred},k}\right),$$

Where:

- y_{true} represents the true labels,
- y_{pred} represents the predicted probabilities,
- n_{normal} is the number of normal instances,
- n_{anomaly} is the number of anomaly instances,
- $n_{\text{unlabeled}}$ is the number of unlabeled instances.

The following pseudo-code outlines the procedural steps involved in computing the semi-supervised cross-entropy loss.

Input:

- Y_{true} : True labels
- Y_{pred} : Predicted labels

Output:

Total loss

Algorithm 4 Semi Super Cross Entropy Loss

1: Initialize cross entropy loss for normal, anomaly, and unlabeled points to zero
2: for each data point in the batch do
    2.1: Separate true and predicted labels for the current data point
    2.2: Categorize data points into normal, anomaly, and unlabeled based on their labels
    2.3: Calculate cross entropy loss for normal points
    2.4: Calculate cross entropy loss for anomaly points
    2.5: Calculate cross entropy loss for unlabeled points
3: end for
4: Sum up the cross entropy losses for normal, anomaly, and unlabeled points
5: return the total semi-supervised cross entropy loss

4.2.5 Additional Loss functions for comparison

In addition to the previously stated bespoke loss functions, two novel loss functions intended for comparison with existing approaches from the "Deep SSAD" and "Self-Supervised" articles are presented. The new loss functions aim to improve the adaptability and performance of deep semi-supervised autoencoders in anomaly detection contexts. By incorporating these loss functions, we aim to benchmark our approach against established methodologies, allowing for a thorough comparison of performance and adaptability across different anomaly detection tasks.

4.2.6 Custom SAD Loss (combination of Hinge and SVDD loss)

The Custom SAD is inspired by the "Deep SSAD" paper's methodology and is designed to address both supervised and unsupervised aspects of anomaly detection. Its unique aspect lies in its treatment of labeled data, where instances are categorized into two classes: normal and anomaly, while unlabeled records are deemed anomalous. This distinction allows for a more nuanced understanding of the data and enhances the model's ability to discern anomalies effectively. Specifically, it utilizes a combination of hinge loss for supervised instances and Deep SVDD (Support Vector Data Description) loss for unsupervised instances. The formulation of the total loss is as follows:

$$\text{Loss} = \text{Supervised Weight} \times \text{Hinge Loss}\left(y_{\text{true supervised},0:}, y_{\text{pred supervised},0:}\right) + \text{Unsupervised Weight} \times \frac{1}{N_{\text{unsupervised}}} \sum_{i=1}^{N_{\text{unsupervised}}} \sum_{j=1}^{D} \left(y_{\text{true unsupervised},i,j} - y_{\text{pred unsupervised},i,j}\right)^2$$

Where:

- $N_{\text{supervised}}$ is the number of instances labeled as supervised,
- $N_{\text{unsupervised}}$ is the number of instances labeled as unsupervised,
- D is the dimensionality of the data,
- Supervised Weight is the weight assigned to the supervised loss term,
- Unsupervised Weight is the weight assigned to the unsupervised loss term,
- $y_{\text{true supervised}}$ is the true label for supervised instances,
- $y_{\text{pred supervised}}$ is the predicted label for supervised instances,
- $y_{\text{true unsupervised},i,j}$ is the true value for feature j in unsupervised instance i,
- $y_{\text{pred}_{\text{unsupervised},i,j}}$ is the predicted value for feature j in unsupervised instance i.

The following pseudo-code outlines the procedural steps involved in computing the Custom SAD Loss.

Input:

- Y_{true} : True labels
- Y_{pred} : Predicted labels
- supervised weight: Weight for supervised loss
- unsupervised weight: Weight for unsupervised loss

Output:

Total loss

Algorithm 5 Custom SAD Loss

1: Initialize supervised and unsupervised loss terms to zero
2: for each data point in the batch do
    2.1: Split data based on label into supervised and unsupervised portions
    2.2: Extract relevant portions of y_true and y_pred for both supervised and unsupervised losses
    2.3: Calculate supervised loss using the Hinge loss function
    2.4: Define the Deep SVDD loss function for unsupervised loss
    2.5: Calculate unsupervised loss (Deep SVDD loss)
3: end for
4: Combine the supervised and unsupervised losses with specified weights
5: return the total Custom SAD Loss

4.2.7 Custom SAD Loss (combination of Binary Cross Entropy and KLD loss)

This Custom SAD loss draws inspiration from the "Deep SSAD",introduces a distinct approach by integrating binary cross-entropy for supervised instances and KLD for unsupervised instances. This amalgamation allows for a comprehensive treatment of labeled and unlabeled data, enhancing the model's capability to detect anomalies across diverse scenarios. The total loss is computed as a weighted combination of both binary cross-entropy and KLD, with careful consideration given to the weights assigned to each component.

$$\text{Loss} = \text{Supervised Weight} \times \text{Binary Cross Entropy Loss} \left(y_{\text{true supervised},0:}, y_{\text{pred supervised},0:}\right) + \text{Unsupervised Weight} \times \frac{1}{N_{\text{unsupervised}}} \sum_{i=1}^{N_{\text{unsupervised}}} \sum_{j=1}^{D} y_{\text{true unsupervised},i,j}.$$

Where:

- $N_{\text{supervised}}$ is the number of instances labeled as supervised,
- $N_{\text{unsupervised}}$ is the number of instances labeled as unsupervised,
- D is the dimensionality of the data,
- Supervised Weight is the weight assigned to the supervised loss term,
- $y_{\text{true supervised}}$ is the true label for supervised instances,
- $y_{\text{pred supervised}}$ is the predicted label for supervised instances,
- $y_{\text{truc unsupervised},i,j}$ is the true value for feature j in unsupervised instance i,
- $y_{\text{pred unsupervised},ij}$ is the predicted value for feature j in unsupervised instance i.

The following pseudo-code outlines the procedural steps involved in computing the Custom SAD Loss..

Input:

- Y_{true} : True labels
- Y_{pred} : Predicted labels
- supervised weight: Weight for supervised loss
- unsupervised weight: Weight for unsupervised loss

Output:

Total loss

Algorithm 6 Custom SAD Loss.

1: Initialize supervised and unsupervised loss terms to zero
2: for each data point in the batch do
    2.1: Split data based on label into supervised and unsupervised portions
    2.2: Extract relevant portions of y_true and y_pred for both supervised and unsupervised losses
    2.3: Calculate supervised loss using Binary Cross-Entropy
    2.4: Calculate unsupervised loss using KL Divergence
3: end for
4: Combine the supervised and unsupervised losses with specified weights
5: return the total Custom SAD Loss.

4.3 Performance Evaluation

Evaluating the performance of algorithms is crucial for understanding their effectiveness in distinguishing between normal and anomalous instances. One widely utilized method for gauging performance is through the examination of a confusion matrix, which provides a detailed breakdown of predictions and their correspondence to ground truth labels. The confusion matrix serves as the foundational tool for assessing the predictive accuracy of anomaly detection methods. Each prediction is classified into one of four categories: true positive (TP), false positive (FP), true negative (TN), and false negative (FN). These categories represent the instance-by-instance match between predictions and the ground truth of whether a sample is an anomaly or not. [14]

	Actual Values			
		Normal	Anomaly	Total
Predicted Values	Normal	True Positive (TP)	False Positive (FP)	Precision = TP/TP+FP
	Anomaly	False Negative (FN)	True Negative (TN)	Negative predicted rate = FN/FN+TN
	Total	TPR = TP/TP+FN	FPR = FP/FP+TN	

Table 4: Confusion Matrix illustrating the essential components for anomaly detection performance quantification

In scenarios where anomaly detection predictions are more nuanced than binary decisions, the introduction of thresholds becomes paramount. This intricate process of mapping ranks or scores to binary labels sets the stage for a nuanced evaluation. Here, we introduce a comprehensive metric ROC-AUC score, a stalwart in anomaly detection studies derived from the insights of the confusion matrix. The ROC (Receiver Operating Characteristic) score is a visual representation that demonstrates how the true positive rate (TPR) and false positive rate (FPR) change at different decision thresholds. Specifically, the TPR is plotted against the FPR, with the latter on the x-axis and the former on the y-axis. The evaluation of our algorithm's performance is quantified using the area under the receiver operator curve (AUC-ROC). AUC is a valuable metric as it condenses the ROC curve's information into a single numerical value, allowing for a quantitative and comparative assessment of anomaly detection algorithms [14]. This

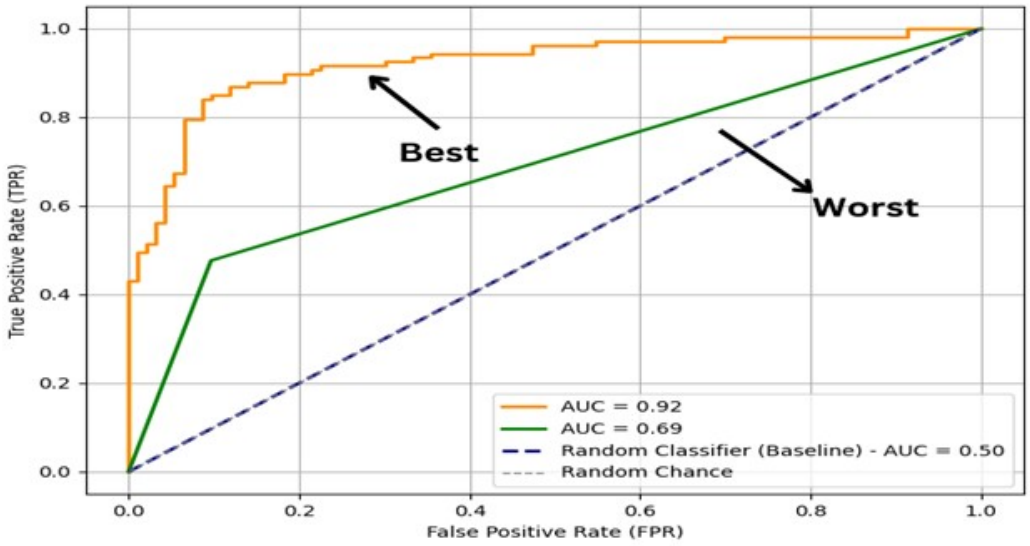

Figure 8: Performance evaluation using ROC-AUC curve

facilitates the simultaneous display of multiple outcomes in a single graph, aiding researchers and practitioners in making informed decisions about the efficacy of different algorithms.

To elucidate, Figure 8 depicts the ROC curve, demonstrating the relationship between TPR and FPR. An ideal anomaly detection algorithm achieves a TPR of 1.0 and an FPR of 0.0, resulting in an AUC value of 1.0. In contrast, a random classifier, reflecting no discriminatory ability, yields an AUC value of 0.5. Our evaluation encompasses two distinct scenarios: the best-case performance (AUC = 0.92, highlighted in orange) and the worst-case performance (AUC = 0.69, denoted in green). The baseline, symbolizing a classifier with no discriminatory power, is visually represented by a diagonal line traversing the plot.

4.4 Data Preparation

At this phase, a systematic approach is used to ensure the robustness of model's performance. Figure 9 illustrates the process of preparing data.

1. **Data cleaning and prepossessing:** Data cleaning involves an initial assessment, and as the data is already cleaned, no further cleaning processes are conducted.

Feature scaling is implemented using Min-Max scaling, a crucial step to normalize numerical features with varying scales. This technique is represented by the equation:

$$X_{\text{scaled}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}}$$

Here, X_{scaled} is the normalized feature, X is the original feature, X_{min} and X_{min} are the minimum and maximum values of X across the data set, respectively.

2. **Data Splitting:** The standard practice involves partitioning the dataset into a training set and a testing set with a ratio of 0.8:0.2. This ensures a well-shuffled and representative distribution for both sets, mitigating bias.

3. **Parameter Tuning:** The hyper parameter tuning phase employs a random search parameter strategy, a vital technique for enhancing model performance. This involves specifying a range or list of values for each hyper parameter and randomly selecting values within these ranges. The iterative nature of this process is crucial for optimizing hyper parameters and refining the model's architecture.

4. **Training Strategy:** For the training strategy, we adopt a robust approach by iteratively training the model ten times. This iterative process ensures the stability of the training procedure and reduces the impact of randomness in the results. By calculating the mean AUC score and standard deviation across these iterations, we obtain a more reliable estimate of the model's performance. This method not only enhances the consistency of the training process but also provides insights into the variability of the model's performance, enabling better interpretation of the results.

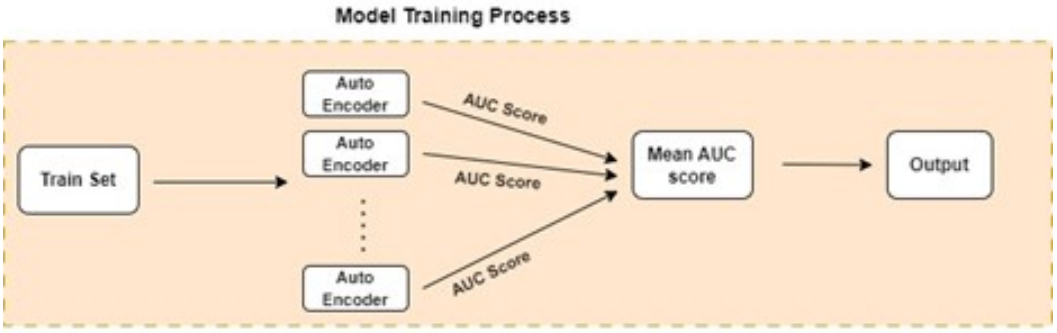

Figure 9: Flow diagram of training process

4.5 Semi-Supervised Anomaly Detection Setup

In our semi-supervised anomaly detection setup, we establish a framework with two distinct classes, namely normal and anomaly, forming the foundation of our anomaly detection configuration. Notably, the training set adopts an unconventional approach, deliberately lacking labels, and is designated as unlabeled. To infuse a degree of supervision into the training process, a small fraction (5 percent) of the test set is amalgamated with the training set, creating a partially labeled dataset. The labeling convention encompasses $\tilde{y} = 0$ for the normal class, $\tilde{y} = 1$ for the anomaly class, and $\tilde{y} = 2$ for unlabeled instances. Figure 10 visually portraits the experimental setup, providing a clear representation of our data configuration and labeling strategy. To quantify and substantiate the efficacy of the approach, the widely recognized AUC measure on the original respective test sets are computed, utilizing ground truth labels for accurate comparison. This ensures a robust and quantitative evaluation of the semi-supervised anomaly detection methodology, shedding light on its performance against conventional benchmarks.

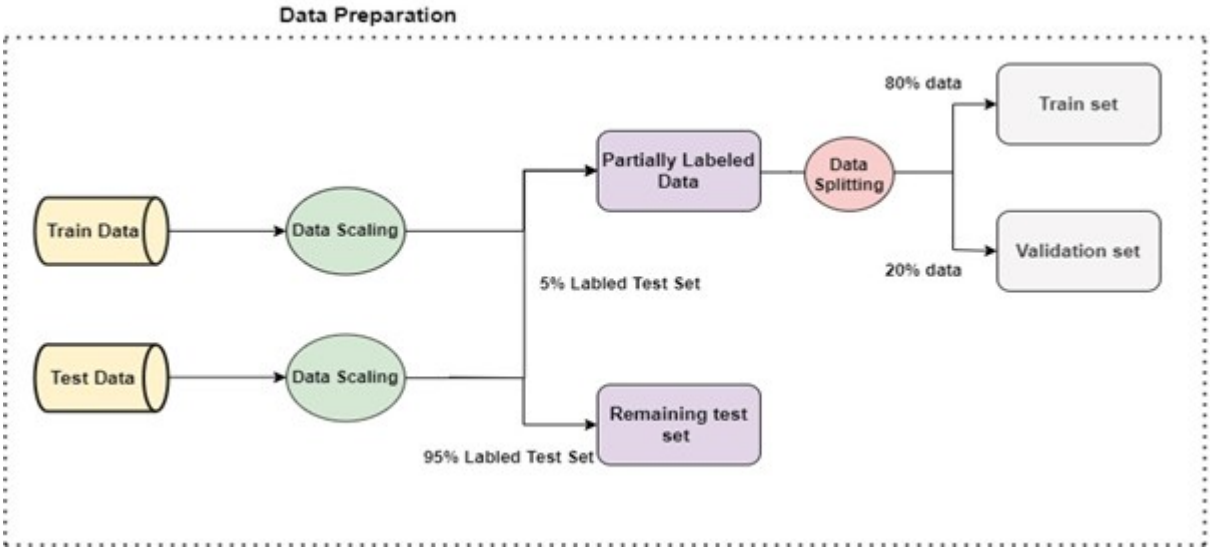

Figure 10: Flow diagram of Data preparation

5 Experiment and Result

In this section, through examination of the datasets and outcomes that serve as the foundation for anomaly detection study. This research looks into the properties of the datasets used, determining their significance and relevance within the experimental framework. The results are given with a strong emphasis on transparency and interpretability, followed with tables summarizing the quantitative findings. A detailed overview, bringing together the key components of our experimentation approach to reveal the insights gained from our anomaly detection models are discussed.

5.1 Data Description and Exploration

To train the model, the experiment uses a comprehensive data set of 40 sets. The Appendix contains thorough information about each data set, including its significance and the outliers it contains. The table below provides a quick description of the attributes found in each data collection.

Dataset	**Train Data Count**	**Columns**	**Test Data Count**	**Anomalies**	**Range (Min-Max)**
satellite	3080	36	2638	1319	$[31.00 - 139.00]$
mnist	6265	664	2056	1028	$[0.00 - 255.00]$
Ecoli	37	7	30	15	$[0.17 - 1.00]$
har	1081	561	926	463	$[-1.00 - 1.00]$
waveform	1185	40	1014	507	$[-3.54 - 8.82]$
thyroid	3586	6	186	93	$[0.00 - 1.00]$
elevators	8029	18	6880	3440	$[-993.00 - 973.00]$
Glass_building.afloat	49	9	42	21	$[0.00 - 73.70]$
spambase	1952	57	1672	836	$[0.00 - 5902.00]$
fashion	6000	784	2000	1000	$[0.00 - 255.00]$
cardio	1479	21	352	176	$[-3.00 - 14.03]$
breast	311	9	266	133	$[1.00 - 10.00]$
speech	3564	400	122	61	$[-5.32 - 4.84]$
Bioresponse	1424	1776	1220	610	$[0.00 - 1.00]$

Table 5: Summary of select datasets utilized in this thesis

To begin the study, firstly exploration of numerous datasets and show visuals. Figures 11, 12, and 13 illustrate samples from the cardiac, Delft pump, and fashion datasets, displaying both normal and anomalous data. The visual evaluation reveals a clear separation between regular and anomalous images, laying the framework for our following analytical work.

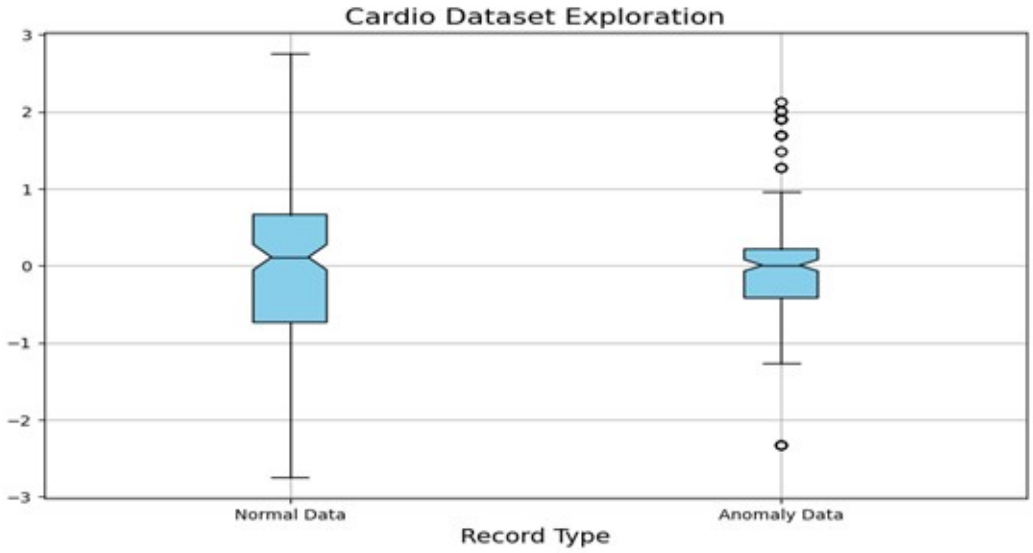

Figure 11: Boxplot representing Normal and Anomaly record from Cardio Data set

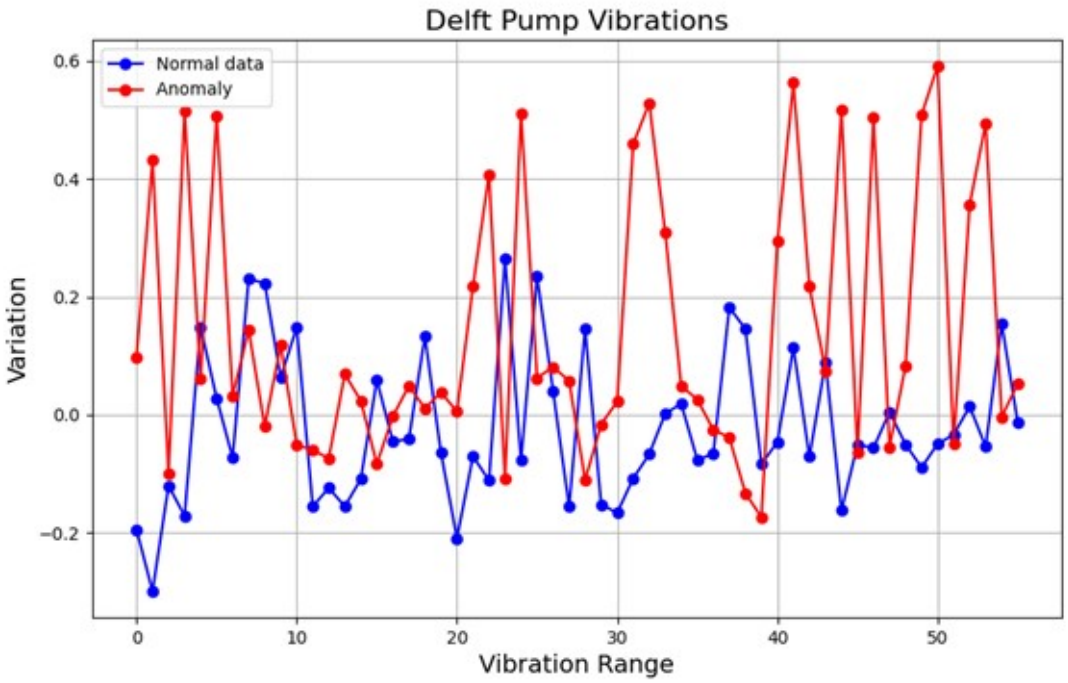

Figure 12: Line chart showing the Normal and abnormal behavior in Delft Pump vibration

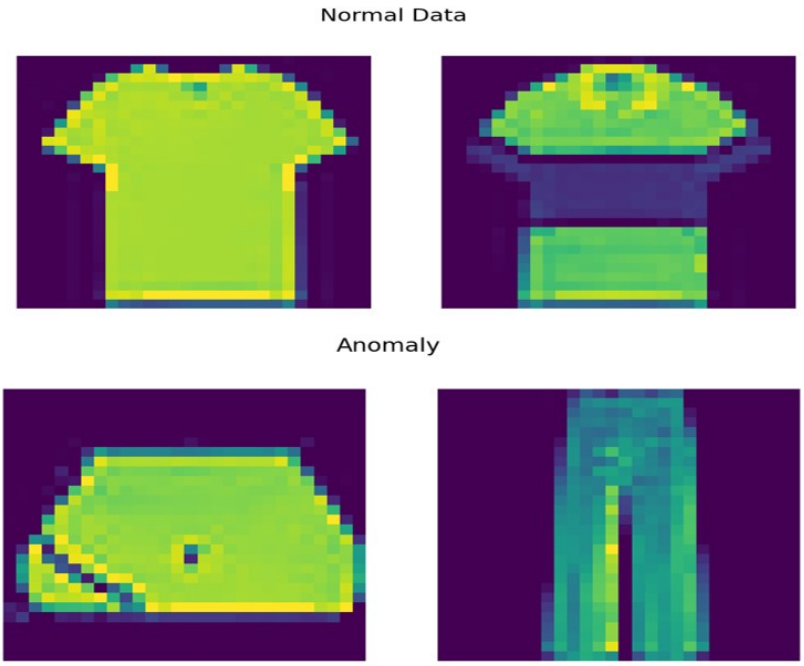

Figure 13: Normal and Anomaly images from Fashion Dataset

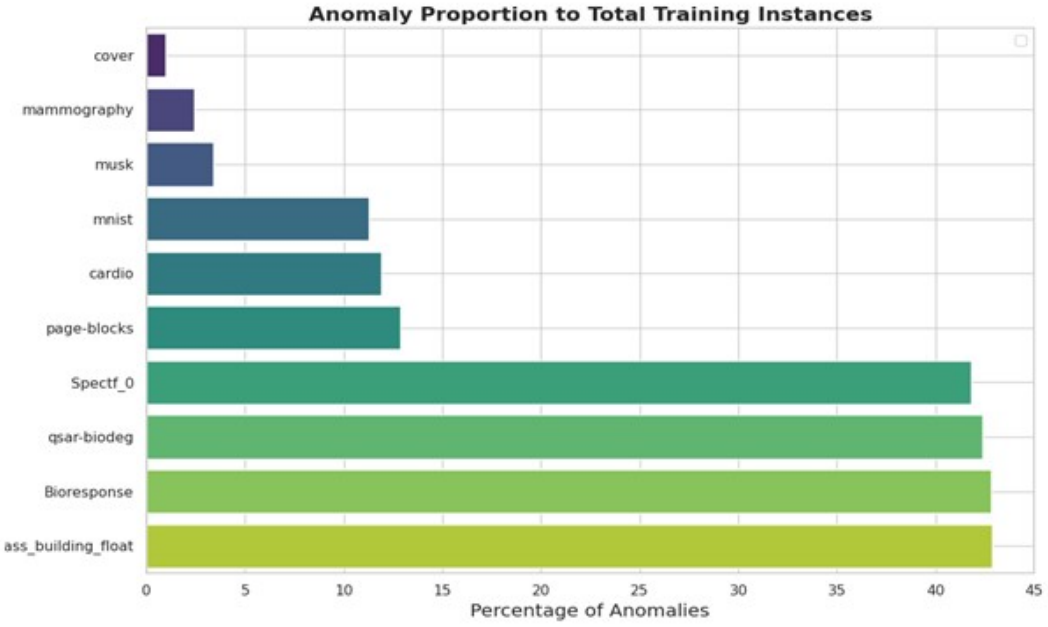

Figure 14: Anomaly proportion to total training instances

Additionally, we delve into the structure of the training and test sets. Figures 14 represent the anomaly proportion to total training instances. These analyses serve to deepen our comprehension of the data set dynamics, providing a solid foundation for a more thorough exploration of our anomaly detection model. Among the 40 datasets scrutinized, it is noteworthy that 25 datasets exhibit a proportion of anomalies in the test set greater than 40% of the training set. This observation suggests a high prevalence of anomalies within these datasets, which can significantly aid the model in effectively distinguishing between normal and anomalous instances. Furthermore, our investigation extends to examining the training and test counts across the datasets. The analysis reveals that the majority of the datasets possess sufficient data for training purposes. However, approximately 12 datasets have fewer than 250 counts, indicating potential challenges associated with limited data availability in these cases. This insight underscores the importance of meticulous attention during model training and tuning, especially in cases where data scarcity may pose challenges.

5.2 Environment setup

All experiments were conducted utilizing the online open-source platform Kaggle. Kaggle provides a robust environment for data science projects, offering access to a wide range of datasets and computational resources. For training the anomaly detection models, employed Nvidia T4 GPUs with 15GB of memory. These GPUs offer powerful parallel processing capabilities, enabling efficient training of deep learning models. Additionally, preprocessing tasks were performed using CPUs with 30GB of RAM. This combination of GPU and CPU resources facilitated the efficient handling and processing of our datasets. All datasets used in the experiments were stored in .npz compressed format. This format allows for efficient storage and retrieval of large datasets while minimizing disk space usage. By compressing the datasets, optimal utilization of storage resources was ensured, streamlining data handling processes during both training and evaluation phases.

5.3 Results

The model described in Section 4.1 was subjected to extensive evaluation through semi-supervised anomaly detection experiments, which were executed 10 times. Evaluation metrics were primarily centered around the Area Under the Curve (AUC) score, providing a comprehensive measure of model performance across multiple runs. Averaging the AUC scores across these runs yielded a representative performance metric, indicative of the model's overall effectiveness in anomaly detection. Additionally, the Standard Deviation (SD) of the AUC scores was calculated and presented in the results table. The standard deviation serves as a measure of variability or deviation from the average performance, with lower values indicating greater consistency and reliability in the model's performance.

Furthermore, to visually illustrate the model's performance, Receiver Operating Characteristic (ROC) curves were plotted, showcasing the trade-off between true positive rate and false positive rate across different thresholds. Results were analyzed based on the utilization of different loss functions, allowing for a comparative assessment of their impact on model performance. This analysis sheds light on the effectiveness of different loss functions in guiding the model towards optimal anomaly detection outcomes.

5.3.1 Deep Semi-Supervised Anomaly Detection

A comprehensive comparison of various loss functions are conducted, yielding three distinct sets of results. These comparisons were based on mean AUC scores, standard deviation, and the percentage of matching anomalies. The table below provides a detailed overview of the mean AUC scores obtained for different loss functions.

In analyzing the results, let's consider the Cardio dataset, where the mean AUC score ranged from 0.9436 for the Cross Entropy loss to 0.9334 for the RenyiEntropy Loss. Despite its smaller size, the Cardio dataset didn't significantly impact the model's performance, as indicated in Table 5. Another example is the Delft_pump dataset, where the mean AUC score ranged from 0.9161 for the Semi-Supervised loss to 0.8992 for the RenyiEntropy loss. Here, despite having ample data and 36 columns, the performance was satisfactory but not exceptional. This suggests that a higher number of columns might make it more challenging to predict anomalies.

Our analysis consistently showed that both the Semi-Supervised loss and Cross Entropy loss outperformed other variants. Notably, the ROC curve in Figure 16 highlights the

Dataset	Mean AUC-ROC Score			
	Semi Supervised Loss	Renyi Entropy Loss	Custom Boosted Loss	Cross Entropy Loss
Cardio	0.934608	0.933426	0.93496	**0.943637**
Delft_pump	**0.916136**	0.899204	0.914385	0.91563
shuttle	**0.9865**	0.986490	0.986426	0.98641
Satellite	0.735051	**0.753493**	0.746137	0.741947
ionosphere	0.845283	0.851286	0.857118	**0.8580**

Table 6: Performance Evaluation of Deep semi supervised Loss Function (Mean AUC Score)

superior performance of the Semi-Supervised Loss Function. Although the RenyiEntropy loss exhibited commendable performance, it encountered inefficiencies across certain datasets, indicating areas for improvement. Conversely, the Custom Boosted loss fell short of expectations, likely due to its nature of handling class weights. These findings prompt further investigation into the performance of the remaining two variants across various metrics.

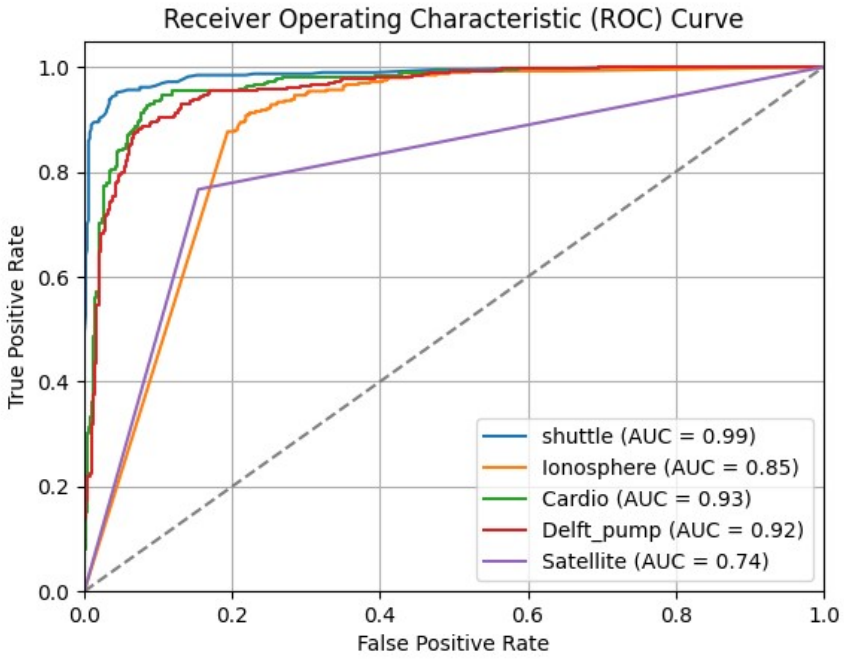

Figure 15: ROC curve of selected datasets using Semi-Supervised Hinge Loss Function

The experiments, each run was conducted 10 times to ensure result consistency, and we used the standard deviation of the AUC scores as a metric. A lower standard deviation indicates better performance consistency across multiple runs, providing valuable insight into result reliability. The table 7 represents a detailed breakdown of the standard deviation scores obtained for different loss functions across five selected datasets. For instance, in the analysis of the Cardio dataset, the Custom Boosted Loss exhibited a standard deviation of 0.00178, whereas the Semi-Supervised Loss demonstrated a notably lower standard deviation of 0.00032, underscoring its contribution to the reliability of AUC scores. Overall, these figures indicate minimal deviation, suggesting that all loss functions provided good consistency in performance. Across the board, it is evident that the cross-entropy loss consistently outperformed others, showcasing lower standard deviation scores.

Dataset	Standard deviation of AUC-ROC Score			
	Semi Supervised Loss	Renyi Entropy Loss	Custom Boosted Loss	Cross Entropy Loss
Cardio	0.00032	0.00031	**0.00178**	0.00023
Delft_pump	0.00017	0.00024	**0.0007**	0.00003
shuttle	0.00018	0.00014	**0.00385**	0.00004
Satellite	0.00031	**0.00035**	0.00003	0.00006
ionosphere	**0.00049**	0.00036	0.00002	0.00006

Table 7: Performance Evaluation of Deep Semi-Supervised Hinge Loss Function (Standard Deviation)

To further evaluate the model's anomaly detection capabilities, we utilize another metric: the percentage of matching anomalies. This metric provides valuable insight into our primary objective of anomaly detection, where a higher percentage indicates better results. The table 8 presents the percentage of anomalies detected by different loss functions across five selected datasets. For instance, in the analysis of the Delft_pump dataset, we observe that the cross-entropy loss achieved a matching anomaly rate of 90%, while the RenyiEntropy Loss yielded an 81% matching anomaly rate. This observation highlights the effectiveness of the cross-entropy loss in matching anomalies. Upon analysis, we once again observe that the cross-entropy loss exhibits superior performance, reaffirming its supremacy over other loss functions within our approach.

Dataset	Matching anomalies (in Percentage)			
	Semi Supervised Loss	Renyi Entropy Loss	Custom Boosted Loss	Cross Entropy Loss
Cardio	92	90	92	92
Delft_pump	89	81	84	90
shuttle	98	99	99	99
Satellite	75	73	74	75
ionosphere	85	84	83	84

Table 8: Performance Evaluation of Deep Semi-Supervised Loss Function (Matching Anomalies)

5.3.2 Deep SAD Method

To compare the results of our method with the Deep SAD approach, we employed two different combinations of loss functions, as detailed in Sections 4.2.6 and 4.2.7, within the model. Each combination was tailored with specific parameters outlined for every dataset in Table 2. These methods were utilized to evaluate the efficacy of our approach. For this comparison, we introduced two variants of SAD loss: one is, combination of binary cross-entropy and KLD loss, and the other is combination of Hinge and SVDD loss.

In the analysis of the Cardio dataset, SAD loss1 (combination of Hinge and SVDD loss) achieved an AUC score of 0.9250, while SAD loss2 (combination of BCE and KLD loss) achieved an AUC score of 0.9357. This indicates that SAD loss performed better in this scenario. Interestingly, while cross-entropy loss has shown dominance in our previous analyses, its combination with other loss functions did not yield significant improvements. Overall, SAD loss2 consistently outperformed SAD loss1 across various datasets. Encouragingly, our approach demonstrated superior performance across the majority of evaluated datasets. However, it is worth highlighting that there were instances where the Deep SAD method exhibited better performance compared to our approach for select datasets. This suggests that the combination of distance-based loss functions worked well in certain scenarios. Below are the results obtained for five select datasets.

Dataset	Deep SAD loss (combination of Hinge and SVDD loss)		Deep SAD loss (combination of BCE and KLD loss)	
	Mean AUC-ROC Score	Standard Deviation	Mean AUC-ROC Score	Standard Deviation
Cardio	0.925	0.000271	0.9357	0.000409
Satellite	0.6061	0.00225	0.7765	0.005525
shuttle	0.9863	0.000012	0.9865	0.000016
Concordia	0.8769	0.000825	0.8742	0.003244
ionosphere	0.7717	0.000438	0.8437	0.001511

Table 9: Performance Evaluation of Deep SAD method

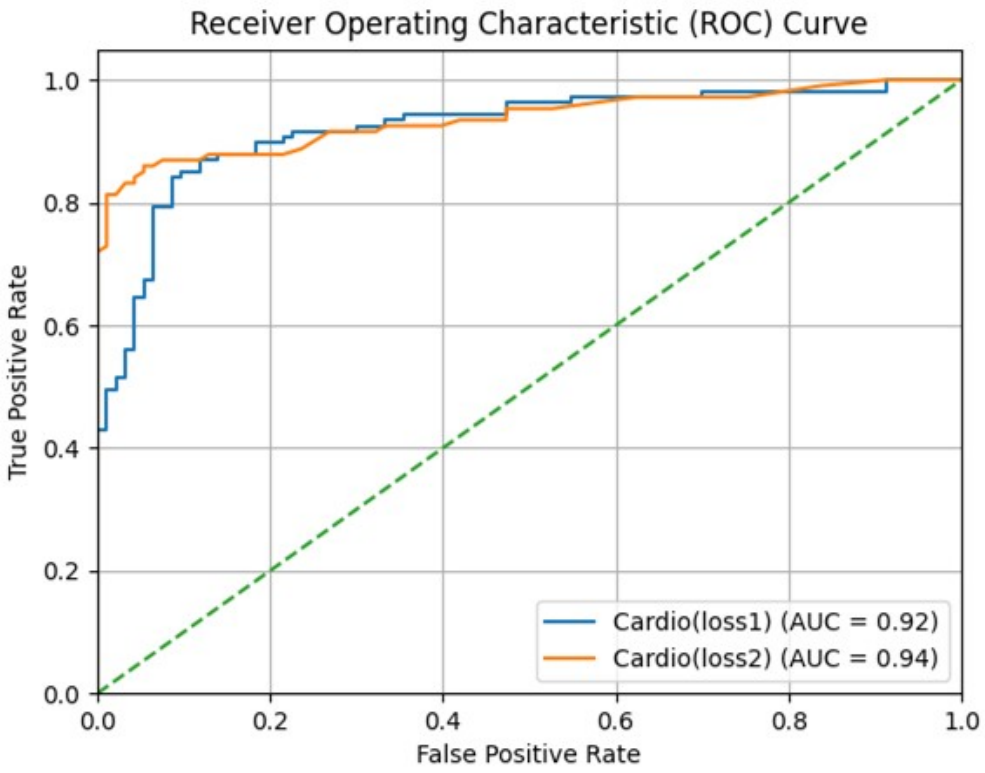

Figure 16: ROC curve of Cardio data set using Deep SAD Loss Functions

5.3.3 Self Supervised Learning Method

For another comparison, we employed a model inspired by the self-supervised learning model described in Sections 4.1.2 and 4.1.3, with distinct loss functions specified for each dataset, as outlined in Table 3. These models were then evaluated for their performance. In the analysis of the Cardio dataset, we observed an AUC score of 0.9651 for the model using Gaussian noise and an AUC score of 0.9688 for the model using random mask. Interestingly, both of these results outperformed those obtained using the Deep SAD method and our approach. Additionally, the standard deviation of these results was relatively low, indicating the reliability of the model. However, upon considering other datasets, this model exhibited lower performance compared to both Deep SAD and our method.

While Gaussian noise and random mask transformations demonstrated promising performance for certain datasets, they were less effective for the majority of datasets. We found that the additional noise introduced by the model did not significantly contribute to its performance improvement. Moreover, the architecture of this model more closely resembles a neural network rather than a deep autoencoder, potentially explaining its varied performance. Further analysis of the observed patterns in results and potential reasons for this behavior will be elucidated in the forthcoming section. Below are the results obtained for five select datasets.

Dataset	Self supervised method (Gaussian Noise)		Self supervised method (Random Mask)	
	Mean AUC-ROC Score	Standard Deviation	Mean AUC-ROC Score	Standard Deviation
Cardio	0.9651	0.0058411	0.9688	0.002813
satellite	0.57004	0.000447	0.5922	0.001136
shuttle	0.5510	0.026184	0.5128	0.019702
Concordia	0.756817	0.000025	0.739970	0.002758
mnist	0.6726	0.001651	0.710532	0.004967

Table 10: Performance Evaluation of Self supervised learning method

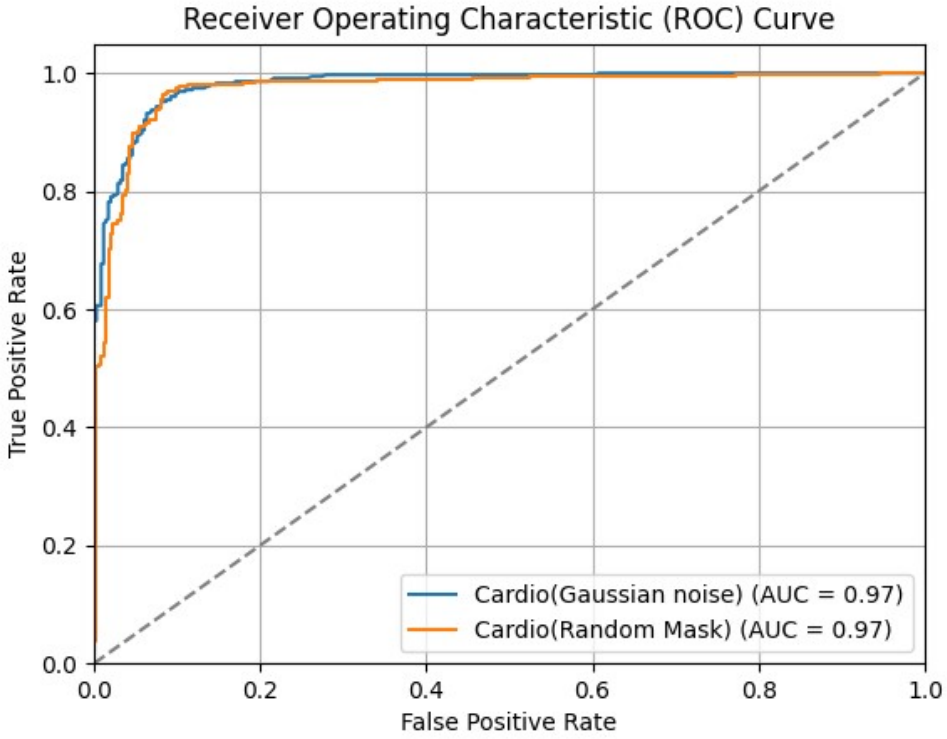

Figure 17: ROC curve of Shuttle data set using Self Supervised method

5.3.4 Overall Comparison

Based on our comprehensive evaluation, which included analyses of various loss functions and methodologies, our custom semi-supervised approach using deep autoencoders consistently outperformed both Deep SAD and self-supervised learning methods, as demonstrated in Table 11. For example, in the Fashion dataset, our method yielded a mean AUC score of 0.8338, while Deep SAD achieved 0.8094, and the self-supervised method reached 0.6876. Notably, the Fashion dataset comprises 784 dimensions, and despite its complexity, our model performed significantly better, showcasing a larger difference in results compared to others.

The success of our method can be attributed to its adaptability and the diverse range of loss functions we experimented with, tailored specifically for fully connected autoencoder networks. This approach enabled us to identify the most effective strategies for achieving optimal results across diverse datasets. In contrast, Deep SAD exhibited moderate results due to its reliance on a combination of two loss functions, where under performance in one component could impact overall performance. Moreover, considering unlabeled

data as anomalies introduced bias, further contributing to its moderate performance. Similarly, the self-supervised learning method, when implemented via semi-supervised learning, may not be as effective as anticipated. The structure of the self-supervised learning model, which diverges from a fully connected autoencoder network, exhibited mixed performance across datasets. Additionally, the addition of noise to the autoencoder structure did not consistently improve performance.

Although the results are promising, it is important to acknowledge that out of the 40 datasets evaluated, there were instances where none of the methods performed satisfactorily, particularly with datasets featuring higher dimensions or more columns. Additionally, in some scenarios, none of the methods were able to provide acceptable results. Thus, to ensure a fair comparison, we focused on a subset of nine datasets where all methods exhibited comparatively better performance.

Dataset	Mean AUC-ROC Score		
	Custom Semi Supervised Method	Deep SAD Method	Self Supervised Method
Cardio	0.943637	0.9357	**0.9688**
Delft_pump	**0.9170**	0.9148	0.6539
shuttle	**0.9865**	0.9864	0.5510
Satellite	0.7534	**0.7765**	0.5922
ionosphere	**0.8580**	0.8437	0.710532
Concordia	0.8744	**0.8769**	0.7568
mnist	0.7715	**0.8453**	0.6844
Mammography	**0.7941**	0.7430	0.6228
Fashion	**0.8338**	0.8094	0.6876

Table 11: Overall Performance Evaluation

6 Conclusion and Future Scope

Anomaly detection, crucial across numerous domains, revolves around pinpointing data points that exhibit significant deviations from the norm within a given data set. The approach to this task spans across supervised, semi-supervised, or unsupervised methodologies, largely dependent on the availability of labeled data. With labeled datasets often difficult to obtain, semi-supervised methods emerge as promising alternatives, leveraging both labeled and unlabeled data. This gives a distinct benefit by enabling the combination of supervised and unsupervised procedures. Auto encoders have gained prominence in this context, presenting a powerful tool for semi-supervised anomaly detection. In our research, we delved deeply into deep semi-supervised anomaly detection utilizing auto encoders, thereby pushing the boundaries of anomaly detection techniques.

Furthermore, our study included a comprehensive comparative analysis, against two established methodologies: self-supervised anomaly detection and Deep SAD (SemiSupervised Anomaly Detection) techniques. Through rigorous experimentation and evaluation, we aimed to showcase the effectiveness and robustness of our proposed method. By elucidating the strengths and weaknesses of each approach, our research contributes to the broader understanding of anomaly detection methodologies, paving the way for advancements in anomaly detection research and applications across diverse domains.

Our novel approach, Deep Semi-Supervised Anomaly Detection, effectively addresses the labeling issue and demonstrates the capability to handle complex datasets, yielding promising results. Furthermore, to mitigate the issue of prolonged training times associated with numerous datasets, we implemented regularization techniques such as early stopping, ensuring efficient model convergence and training efficiency. In addition to methodological advancements, we significantly contributed to this thesis by considering four distinct custom loss functions tailored specifically for auto encoders: Semi-Supervised Hinge Loss, Generalized Rényi Entropy Loss, Custom Boosted Semi-Supervised Loss, and Semi-Supervised Cross-Entropy Loss. Each of these loss functions offers unique advantages, enhancing the optimization process and contributing to the overall robustness of our anomaly detection framework. Our exploration of the Semi-Supervised Hinge Loss, as elaborated in Section 4.2.1, showcased its efficacy in penalizing deviations from the correct classification boundary, leading to promising results across various datasets. This loss function consistently outperformed both Deep SAD and self-supervised learning approaches in the majority of cases, reaffirming its superiority in anomaly detection tasks.

The Generalized Rényi Entropy Loss, a non-linear entropy-based method detailed in Section 4.2.2, demonstrated competency across the datasets. However, our analysis did not reveal superior performance compared to existing loss functions or the Deep SAD approach. Notably, our observations suggest that its capacity for anomaly detection is comparatively lower than that of other loss functions. The discrepancy could be attributed to the fundamental character of non-linear methods, which may fail to define decision limits as effectively as linear approaches.

Custom Boosted Semi-Supervised Loss, a boosting-like mechanism with sample weights explained in Section 4.2.3, showcased remarkable performance. This approach demonstrated superiority over both the Deep SAD and self-supervised approaches. Particularly noteworthy is its ability to achieve a high percentage of anomaly detection, attributed to the sample weight mechanism which prioritizes challenging instances. This highlights the effectiveness of leveraging sample weights to enhance the focus on intricate anomalies, ultimately improving the overall performance of the anomaly detection system. The Semi-Supervised Cross Entropy Loss, a traditional cross-entropy loss as detailed in Section 4.2.4, demonstrated exceptional performance, boasting superior AUC scores compared to other proposed custom loss functions, as well as the Deep SAD and self-supervised approaches. Additionally, we observed a gradual decrease in standard deviation, indicating model stabilization over time. Our exploration of the Deep SAD method, where combinations of two loss functions were utilized as explained in Section 4.2.6 and 4.2.7, yielded decent performance but fell short of surpassing our proposed approach consistently. Nonetheless, in certain instances, this method exhibited notable improvements over others.

In contrast, the self-supervised learning method, which utilizes Gaussian noise and random mask transformations as described in Sections 4.1.2 and 4.1.3, showcased promising performance for select datasets. However, it proved less effective across the majority of datasets. This disparity in effectiveness may be attributed to differences in architecture compared to autoencoders, coupled with the possibility that additional noise did not contribute significantly to performance enhancement. Notably, we observed that datasets with fewer dimensions (less than 25) tended to outperform more complex datasets, possibly indicating the presence of the curse of dimensionality.

Our approaches have demonstrated superiority over existing methods across a diverse range of datasets, as evidenced by the consistently higher AUC scores achieved. This success underscores the effectiveness and real-world applicability of our methodology in anomaly detection scenarios. While our method demonstrated consistent superiority over both Deep SAD and self-supervised learning methods across various datasets, it is essential to acknowledge areas where our approach fell short of expectations. Specifically, our analysis revealed that the Generalized Rényi Entropy Loss did not yield the expected results and did not outperform existing loss functions or the Deep SAD approach. This discrepancy highlights the inherent challenges associated with non-linear methods in delineating decision boundaries effectively. Moreover, while our approaches surpassed those presented in other research papers in many instances, they did not consistently outperform benchmark results across all datasets, presenting a noteworthy drawback.

Moving forward, there are several avenues for extending the research presented in this thesis. The innovative auto encoding approach utilized opens doors for further enhancement of the model. Exploring the optimization of the proposed loss functions could lead to substantial improvements in performance. Additionally, ensemble methods offer promise by combining multiple models to yield more robust outcomes. Transitioning from random hyper parameter tuning to more advanced techniques such as grid or Bayesian optimization could further refine the model's effectiveness. Furthermore, the implementation of new custom loss functions presents an opportunity for innovation and refinement. Finally, expanding the scope of the research to encompass multi-dimensional or unused datasets could provide valuable insights into the model's efficiency across diverse data landscapes. These future directions hold the potential to advance anomaly detection methodologies and contribute to the broader field of machine learning research.

7 Bibliography

[1] Markus Goldstein and Seiichi Uchida. *Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data.* https://doi.org/10.1371/journal.pone.0152173

[2] Khaled A. Alaghbari, Heng-Siong Lim, Mohamad Hanif Md Saad, and Yik Seng Yong. *Deep Autoencoder-Based Integrated Model for Anomaly Detection and Efficient Feature Extraction in IoT Networks.* https://doi.org/10.3390/iot4030016

[3] Varun Chandola, Arindam Banerjee, and Vipin Kumar. *Outlier Detection: A Survey.* http://cucis.ece.northwestern.edu/projects/DMS/publications/AnomalyDetection.pdf

[4] Jie Xu, Suri Guga, Guangzhi Rong, Dao Riao, Xingpeng Liu, Kaiwei Li, and Jiquan Zhang. *Estimation of Frost Hazard for Tea Tree in Zhejiang Province Based on Machine Learning.* https://www.mdpi.com/2077-0472/11/7/607

[5] Sebastian Ruder. *An Overview of Gradient Descent Optimization Algorithms.* https://arxiv.org/pdf/1609.04747.pdf

[6] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. *Deep Sparse Rectifier Neural Networks.* https://proceedings.mlr.press/v15/glorot11a/glorot11a.pdf

[7] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization.* https://arxiv.org/pdf/1711.05101.pdf

[8] D. P. Kingma and M. Welling. *Auto-Encoding Variational Bayes.* https://arxiv.org/pdf/1312.6114.pdf

[9] Lukas Ruff, Robert A. Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. *Deep One-Class Classification.* https://proceedings.mlr.press/v80/ruff18a/ruff18a.pdf

[10] Y. LeCun, Y. Bengio, and G. Hinton. *Deep Learning.* https://www.nature.com/articles/nature14539

[11] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko. *A Review of Novelty Detection.* https://www.sciencedirect.com/science/article/abs/pii/S016516841300515X

[12] G. E. Hinton and R. R. Salakhutdinov. *Reducing the Dimensionality of Data with Neural Networks*. https://www.cs.toronto.edu/~hinton/absps/science.pdf

[13] V. J. Hodge and J. Austin. *A Survey of Outlier Detection Methodologies*. https://link.springer.com/article/10.1007/s10462-004-4304-y

[14] J. Davis and M. Goadrich. *The Relationship Between Precision-Recall and ROC Curves*. https://dl.acm.org/doi/10.1145/1143844.1143874

[15] Joseph Isabona, Agbotiname Lucky Imoize, Stephen Ojo, Olukayode Karunwi, Yongsung Kim, Cheng-Chi Lee, and Chun-Ta Li. *Development of a Multilayer Perceptron Neural Network for Optimal Predictive Modeling in Urban Microcellular Radio Environments*. https://www.mdpi.com/2076-3417/12/11/5713

[16] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio. *Deep Learning*. https://www.deeplearningbook.org/front_matter.pdf

[17] Hadi Hojjati, Thi Kieu Khanh Ho, and Naregs Armanfard. *Self-Supervised Anomaly Detection: A Survey and Outlook*. https://arxiv.org/pdf/2205.05173.pdf

[18] Lukas Ruff, Robert A. Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. *DEEP SEMI-SUPERVISED ANOMALY DETECTION*. https://arxiv.org/pdf/1906.02694.pdf

[19] Songqiao Han, Xiyang Hu, Hailiang Huang, Minqi Jianga, and Yue Zhao. itAD-Bench: Anomaly Detection Benchmark. https://arxiv.org/pdf/2206.09426.pdf

[20] Jerone T. A. Andrews, Morton, Edward J. Morton, and Lewis D. Griffin. *Detecting Anomalous Data Using Auto-Encoders*. https://www.ijmlc.org/vol6/565-L009.pdf

[21] L. Ruff, Y. Zemlyanskiy, R. Vandermeulen, T. Schnake, and M. Kloft. *Self-attentive, multi-context one-class classification for unsupervised anomaly detection on text*. https://ml.cs.uni-kl.de/publications/2019/acl2019.pdf

[22] Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. *Anomaly detection using one-class neural networks*. https://arxiv.org/pdf/1802.06360.pdf

[23] Tolga Ergen, Ali Hassan Mirza, and Suleyman Serdar Kozat. *Unsupervised and semi-supervised anomaly detection with LSTM neural networks*. https://arxiv.org/abs/1710.09207

[24] Dan Hendrycks, Mantas Mazeika, and Thomas G Dietterich. *Deep anomaly detection with outlier exposure*. https://arxiv.org/abs/1812.04606

[25] Simon Hawkins, Hongxing He, Graham Williams, and Rohan Baxter. *Outlier Detection Using Replicator Neural Networks*. https://link.springer.com/chapter/10.1007/3-540-46145-0_17

[26] Yassine Ouali, Céline Hudelot, and Myriam TamAn. *Overview of Deep Semi-Supervised Learning*. https://arxiv.org/pdf/2006.05278.pdf

[27] Marco Schreyer, Timur Sattarov, and Damian Borth. *Multi-view Contrastive Self-Supervised Learning of Accounting Data Representations for Downstream Audit Tasks*. https://arxiv.org/pdf/2109.11201.pdf

[28] Marco Rudolph, Bastian Wandt, and Bodo Rosenhahn. *Same same but differnet: Semi-supervised defect detection with normalizing flows*. https://arxiv.org/pdf/2008.12577.pdf

[29] Chris Kuo/Dr. Dataman. *Handbook of Anomaly Detection with Python Outlier Detection — (12) Autoencoders*. https://towardsdatascience.com/anomaly-detection-with-autoencoder-b4cdce4866a6 (visited on 11/21/2023).

[30] Renu Khandelwal. *Anomaly Detection using Autoencoders*. https://towardsdatascience.com/anomaly-detection-using-autoencoders-5b032178a1ea (visited on 11/23/2023)

[31] Kunal Chowdhury.
Understanding loss functions: Hinge loss.
https://medium.com/analytics-vidhya/understanding-loss-functions-hinge-loss-a0ff
(visited on 18/10/2023)

[32] Hasan Torabi, Seyedeh Leili Mirtaheri, and Sergio Greco. *Practical autoencoder based anomaly detection by using vector reconstruction error*. https://cybersecurity.springeropen.com/articles/10.1186/s42400-022-00134-9

[33] Sultan Zavrak and Murat İskefiyeli. *Anomaly-Based Intrusion Detection From Network Flow Features Using Variational Autoencoder.* https://acikerisim.sakarya.edu.tr/bitstream/handle/20.500.12619/95381/10.1109%20ACCESS.2020.3001350.pdf

]Appendix

1. **Satellite Dataset:** This dataset comprises multi-spectral pixel values of 3x3 satellite image neighborhoods, accompanied by classifications associated with the central pixel of each neighborhood. It involves binary classification of satellite image features into normal and anomaly categories.
2. **Cardio Dataset:** The Cardio dataset contains medical attributes related to heart disease diagnosis, sourced from the UCI. It involves binary classification of heart disease presence as normal and absence as anomaly.
3. **MNIST Dataset:** MNIST Dataset: The Modified National Institute of Standards and Technology (MNIST) database focuses on handwritten digits, specifically selecting instances representing the digit 7. It involves binary classification between normal (digit 7) and anomaly (non-7 digits) categories.
4. **Ecoli Dataset:**This dataset contains features derived from protein sequences for classifying subcellular localization sites in Escherichia coli. It entails binary classification between normal (non-anomalous) and anomalous subcellular localization sites.
5. **Letter Dataset:** The Letter dataset consists of rectangular black and white pixel displays of 26 different capital letters of the English alphabet. It involves binary classification for letter recognition into normal and anomaly categories.
6. **Vowel Dataset:** The Vowel dataset is an undocumented dataset from UCI, focusing on recognizing eleven British English vowels. It involves binary classification of vowel characteristics into normal and anomaly categories.
7. **Cover Type Dataset:** The Cover Type dataset comprises cartographic features predicting forest cover types. It involves binary classification of forest cover types into normal and anomaly categories.
8. **Concordia Dataset:** The Concordia3_32 dataset consists of black-and-white images of the digit 3 stored in 32x32 dimensions. It involves binary classification of digit 3 images as normal and other digits as anomaly
9. **Speech Dataset:** The Speech dataset includes English language recordings with anomalies represented by additional speakers with varied accents. It involves binary classification of speech segments into normal and anomaly categories.

10. **Gas Drift Dataset:** The Gas Drift dataset contains data from 16 chemical sensors used to discriminate between six gases at different concentrations. It involves binary classification of gas types into normal and anomaly categories.
11. **Vehicle Van Dataset:** The Vehicle Van dataset consists of silhouette images of different types of vehicles, particularly focusing on vans. It involves binary classification of vehicle types into van and other categories.
12. **Liver Dataset:** The Liver dataset contains medical attributes related to liver diseases, with the target indicating the presence or absence of liver disease. It entails binary classification of liver disease presence as normal and absence as anomaly.
13. **Pima Dataset:** The Pima dataset comprises medical attributes for predicting the onset of diabetes among Pima Indian women. It involves binary classification of diabetes onset as normal and absence as anomaly.
14. **HeartC Dataset:** The HeartC dataset includes attributes related to heart disease indicators from the Cleveland database. It involves binary classification of heart disease presence as normal and absence as anomaly.
15. **Shuttle Dataset:** The Shuttle dataset contains attributes from sensor readings for classifying anomalies in shuttle operations. It involves binary classification of shuttle anomalies into normal and anomaly categories.
16. **Musk Dataset:** The Musk dataset aims to discriminate musk molecules from non-musk molecules based on various molecular features. It involves binary classification of musk molecules as normal and non-musk molecules as anomaly.
17. **Fashion Dataset:** The Fashion dataset contains grayscale images of fashion items, particularly focusing on classifying clothing types. It involves binary classification of fashion items into normal and anomaly categories.
18. **Abalone Dataset:** The Abalone dataset includes physical measurements of abalone specimens for predicting their age. It involves binary classification of abalone age as normal and anomaly.
19. **Mammography Dataset:** The Mammography dataset contains attributes from mammogram images used for breast cancer diagnosis. It involves binary classification of mammogram findings into normal and anomaly categories.

20. **Cover Datasett:** The Cover dataset contains attributes derived from satellite imagery for predicting forest cover types. It involves binary classification of forest cover types into normal and anomaly categories.
21. **Thyroid Dataset:** The Thyroid dataset contains medical attributes for diagnosing thyroid disorders. It involves binary classification of thyroid health into normal and anomaly categories.
22. **Ionosphere Dataset:** The Ionosphere dataset is a binary classification dataset containing radar data collected by the Goose Bay Laboratory. It involves binary classification of radar returns into normal and anomaly categories.
23. **Sonar Mines Dataset:** The Sonar Mines dataset contains sonar signals reflected from metal cylinders (mines) and rocks collected under controlled conditions. It involves binary classification of sonar signals into normal and anomaly categories.