RBM-OD: A RESTRICTED BOLTZMANN MACHINE FRAMEWORK FOR OUTLIER DETECTION

BRADY HOEKSEMA
Bachelor of Science, University of Lethbridge, 2023

A thesis submitted
in partial fulfilment of the requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

Department of Mathematics and Computer Science
University of Lethbridge
LETHBRIDGE, ALBERTA, CANADA

RBM-OD: A RESTRICTED BOLTZMANN MACHINE FRAMEWORK FOR OUTLIER DETECTION

BRADY HOEKSEMA

Date of Defence: August, 2025

Dr. John Zhang Thesis Supervisor	Associate Professor	Ph.D.
Dr. Wendy Osborn Committee Member	Associate Professor	Ph.D.
Dr. Yllias Chali Committee Member	Professor	Ph.D.
Dr. Andrew Fiori Chair, Thesis Examination Committee	Associate Professor	Ph.D.

Dedication

To Mom, Dad, Taylor, and Alex.

Abstract

This thesis explores the use of Restricted Boltzmann Machines (RBMs), a class of unsupervised generative neural networks, for detecting outliers through data generation and representation-based comparison. Outlier detection (OD) is a critical task in domains where rare or anomalous patterns may indicate errors, fraud, or unexpected behaviour in data.

The primary contribution of this work is a unified framework for RBM-based outlier detection that emphasizes data generation as a detection strategy. We explore multiple model variants, including single RBMs, ensembles of RBMs, stacked RBMs, and ensembles of stacked RBMs, each offering distinct advantages in representing complex data patterns. By generating synthetic samples from trained RBMs and comparing them to input data, the approach enables unsupervised detection of unusual or unexpected instances. This generative perspective distinguishes RBM-OD from traditional methods and provides a flexible foundation for future extensions.

Acknowledgments

First and foremost, I would like to thank Dr. John Zhang for his unwavering support and the many hours he dedicated to this work. His guidance throughout this journey was instrumental in shaping and expanding this work into a full-fledged thesis. I am also grateful for our many conversations (both about research and about our dogs) which made this experience all the more meaningful.

I would also like to thank the members of my committee, Dr. Wendy Osborn and Dr. Yllias Chali, for their valuable comments, insightful advice, and the wisdom they shared throughout this process.

To Mom, Dad, and Taylor: I would not be here without your constant love and encouragement. Your support has meant everything to me.

To Alex: thank you for your patience and understanding during the long nights and countless hours this project required. Your support helped carry me through .

To my four legged family members: thank you for the much-needed distractions and the comfort you gave during this time. I owe you many treats.

Finally, heartfelt thanks to the friends I made along the way and to everyone who supported me throughout this journey. Your kindness and encouragement will not be forgotten.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Artificial Intelligence (AI) is a central area of research and innovation in computer science, with broad applications across industries. This chapter introduces the foundational concepts leading to the method explored in this thesis. It begins by outlining the general field of AI and the rise of machine learning as a dominant paradigm. The focus then shifts to neural networks, highlighting a specific family known as Boltzmann Machines. The chapter concludes with an introduction to the Restricted Boltzmann Machine (RBM), which forms the basis of the proposed outlier detection framework in our research.

1.1 Artificial Intelligence and Machine Learning

Artificial Intelligence (AI) refers to the development of computer systems capable of performing tasks that typically require human intelligence, such as perception, reasoning, and decision-making. Over recent decades, AI has evolved from rule-based systems to data-driven approaches that learn from experience. As a result, AI has become an integral part of modern technology, with applications in healthcare, finance, transportation, and personal assistants [38, 50].

AI systems are often categorized by their scope of capability. Narrow AI, also known as weak AI, is designed for specific tasks and represents the most common type in use today. Examples include virtual assistants like Siri and Alexa, recommendation engines, and fraud detection tools [6]. In contrast, general AI refers to systems that can perform any intellectual task that a human can. While full general intelligence is not yet achieved, advances in large language models, such as GPT-4, demonstrate progress toward more generalized capabilities [17]. Super AI, a hypothetical system surpassing human intelligence in all domains, remains speculative and the subject of science fiction [15].

The recent success of AI is largely attributed to the growth of Machine Learning (ML), a subfield focused on enabling computers to learn from data. Instead of relying on predefined rules, ML algorithms discover patterns and relationships directly from the data they are trained on, allowing them to make predictions or decisions on new, unseen inputs [56]. ML tasks are generally categorized into supervised, unsupervised, and reinforcement learning. In supervised learning, the algorithm is trained on labelled data to predict specific outcomes, as in email spam detection or image classification [32]. In unsupervised learning, the model learns the structure of unlabelled data, often through clustering or dimensionality reduction [26]. Reinforcement learning involves an agent learning to take actions within an environment by maximizing cumulative rewards over time [28].

This shift toward data-driven intelligence lays the foundation for neural networks, which are the primary focus of the next section.

1.2 Neural Networks

Neural networks are a class of machine learning models inspired by the structure and function of the human brain [46]. They consist of layers of interconnected units, called neurons, which process and transform data through weighted connections [3]. A typical neural network includes an input layer, one or more hidden layers, and an output layer. Each neuron in a layer receives inputs, applies a transformation (often nonlinear), and passes the result to the next layer. An example of a simple neural network architecture can be found in Figure 1.1.

Early work on neural networks begins with the McCulloch-Pitts neuron model in the 1940s [48], followed by the perceptron developed by Rosenblatt in the 1950s [53]. After a period of limited progress, interest in neural networks resurges in the 1980s with the

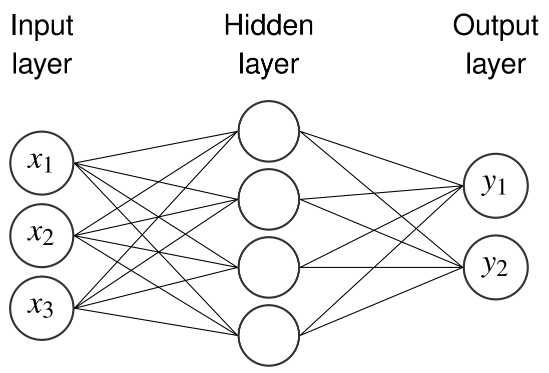


Figure 1.1: A simple neural network architecture.

introduction of the backpropagation algorithm [41], which enables efficient training of multilayer networks.

Neural networks are particularly effective at modeling complex, non-linear relationships in data, making them suitable for tasks such as speech recognition, image classification, and natural language processing [55]. Recent advances in computing power and data availability further drive the adoption of deep learning, an approach that extends traditional neural networks by increasing their depth (i.e., number of layers) to enable the learning of hierarchical representations [10].

Various architectures exist to support different applications. Feedforward neural networks (FNNs) pass information in one direction, from input to output, and are used in many basic classification tasks [21]. Convolutional neural networks (CNNs), designed to exploit spatial hierarchies in data, are especially effective for image analysis [20].

Among the more specialized types of neural networks are those designed for unsupervised learning and generative modelling. One such model is the Boltzmann Machine, which forms the theoretical foundation for the Restricted Boltzmann Machine studied in this thesis.

1.3 The Boltzmann Machines

The Boltzmann Machine (BM) model is a type of stochastic neural network designed for unsupervised learning and generative modeling. It differs from traditional feedforward architectures in that it models the joint probability distribution of its inputs and can generate new data samples by drawing from this distribution [3].

A Boltzmann Machine consists of two layers of binary-valued units: a visible layer that represents the input data and a hidden layer that captures dependencies and patterns. Unlike standard neural networks, connections exist not only between the visible and hidden units but also within each layer. Generally speaking, these undirected connections allow the network to reach an equilibrium state by iteratively updating the states of its neurons based on probabilistic rules.

Training a BM involves adjusting its weights to minimize an energy function, which specifies the probability of a given configuration of visible and hidden units. Configurations with lower energy are interpreted as more probable under the model. While powerful in theory, full Boltzmann Machines are difficult to train and scale due to their fully connected structure and computational complexity.

To address these limitations, a simplified version known as the Restricted Boltzmann Machine (RBM) is developed. The Restricted Boltzmann Machine (RBM) is a variant of the Boltzmann Machine that enables more efficient training by imposing a key architectural constraint: connections between units within the same layer are removed. This restriction eliminates cycles in the network and allows for more tractable learning algorithms, such as contrastive divergence, to be used in practice [3].

An RBM consists of a visible layer that receives input data and a hidden layer that learns to represent dependencies among the input features. The model trains to capture the underlying structure of the data in an unsupervised manner by adjusting its weights to favour low-energy configurations. Once trained, the RBM can generate new data samples or extract features from the original inputs.

RBMs are applied to various tasks, including dimensionality reduction, feature learning, collaborative filtering, and anomaly detection. Their generative capabilities make them particularly well-suited for identifying unusual or rare patterns in data—an idea central to this thesis.

More complex variants of the RBM include the Stacked RBM, where multiple RBMs are trained sequentially to form a deep architecture, and the Deep Boltzmann Machine (DBM), which trains these layers jointly with bidirectional connections. These extensions increase the model's ability to learn hierarchical representations, though at the cost of increased training complexity [11].

Further theoretical details on RBMs and their training mechanisms are provided in Chapter 2.

1.4 Outlier Detection

Outlier detection is a fundamental problem in data analysis, concerned with identifying observations that deviate significantly from the general pattern of a dataset. Such deviations may correspond to critical and informative events, such as fraudulent transactions, medical abnormalities, or network intrusions, or they may represent errors and noise introduced during data collection [2, 37]. Detecting these unusual instances is essential, as they often provide valuable insights and directly impact decision-making in domains such as finance, healthcare, system monitoring, and cybersecurity [2].

Formally, an outlier can be defined as an observation that is inconsistent with the distribution assumed to have generated the majority of the data [37]. While some outliers result from random error, others may signal rare but meaningful phenomena that require attention. This duality makes outlier detection both practically significant and methodologically challenging.

Traditional methods of outlier detection include statistical tests, distance-based techniques, and clustering approaches [2]. Although effective in certain settings, these methods often struggle with high-dimensional data, complex attribute interactions, and categorical datasets. As modern applications increasingly involve such complexities, there is growing demand for models capable of capturing intricate data distributions in an unsupervised manner.

This thesis explores the Restricted Boltzmann Machine (RBM) as a foundation for outlier detection. By leveraging the generative capabilities of RBMs to learn normal data distributions, it becomes possible to identify points that deviate significantly from expected patterns. This perspective motivates the framework developed in this research, which is introduced in the following section.

1.5 Our Contribution

This thesis presents a novel application of the Restricted Boltzmann Machine (RBM) for unsupervised outlier detection. The primary contribution is the design of a detection framework in which the generative abilities of the RBMs are used to create a synthetic dataset based on learned features and patterns from the RBM. This synthetic dataset allows us to test outliers not only on the original dataset used for training, but any hidden relationships the model identified.

A second contribution is the application of this approach to both binary and categorical datasets, demonstrating its adaptability across different data types. For binary data, the RBM operates directly on the input. For categorical data, appropriate encoding techniques convert the input into a binary-compatible format suitable for RBM training.

The effectiveness of the proposed method is validated through experiments on multiple synthetic and real-world datasets. The results highlight the RBM's capacity to model complex data distributions and reveal anomalies without supervision, underscoring its potential as a generative model for outlier detection tasks.

1.6 Outline

The remainder of this thesis is organized as follows.

Chapter 2 presents relevant background information. It reviews existing methods for outlier detection, introduces the theoretical foundations of the Restricted Boltzmann Machine (RBM), and outlines techniques for encoding categorical data into binary form. It also discusses ensemble learning strategies that are applied in Chapters 3 and 4.

Chapter 3 presents the core methodology and contributions of this thesis. We describe the RBM variants utilized in our approach and explore how their generative capabilities are leveraged for data synthesis. The chapter also introduces the evaluation metrics used to assess the quality of the generated data, along with the outlier detection strategies employed to identify anomalous patterns.

Chapter 4 presents the results, discussion, and analysis of our work. We detail the software tools used in our experiments, describe the datasets on which our models were tested, and outline the specific outlier injection techniques employed to evaluate detection performance.

Finally, Chapter 5 summarizes the contributions of this work. It reflects on the strengths and limitations of applying RBMs to outlier detection and outlines potential directions for future research aimed at extending and enhancing the proposed approach.

Chapter 2

Background and Related Works

This chapter provides the necessary background knowledge on outlier detection and related areas used throughout the following chapters. Outlier detection plays a critical role in many domains, such as identifying fraudulent activities in financial systems and detecting rare medical conditions in healthcare [59]. Given its wide range of applications, it is important to employ efficient methods capable of accurately identifying outliers. This chapter is structured to follow the key steps involved in the proposed approach.

2.1 Boltzmann Machines

Restricted Boltzmann Machines (RBMs) are generative neural network models particularly effective for unsupervised learning tasks, notably in the context of deep learning and artificial intelligence. Derived from the more general Boltzmann Machine (BM), RBMs leverage an energy-based probabilistic framework to model binary data efficiently. This section begins by situating RBMs within the broader landscape of deep learning, highlighting their historical significance and ongoing relevance. Subsequently, the theoretical foundations of Boltzmann Machines are introduced to provide essential intuition, followed by a detailed exploration of the RBM's structure, training methodologies, and generative capabilities. A comprehensive understanding of RBMs underpins the outlier detection framework central to this thesis.

2.1.1 Deep Learning and Artificial Intelligence

Deep learning is an advanced subfield within artificial intelligence (AI) characterized by neural network architectures composed of multiple layers. These networks have the capability to learn hierarchical representations from data automatically, bypassing the need for extensive manual feature engineering. Through backpropagation and large-scale data training, deep learning models can capture increasingly abstract and sophisticated patterns, significantly enhancing performance in areas such as computer vision, natural language processing, reinforcement learning, and anomaly detection [46, 32].

Central to the development of deep learning methodologies are Boltzmann Machines, particularly the RBM. RBMs have historically been critical in the successful implementation of Deep Belief Networks (DBNs[3]), which were among the earliest successful deep learning architectures. DBNs leverage RBMs for unsupervised pre-training, which initializes network weights effectively, overcoming the difficulties associated with training deep neural networks due to gradient instability or vanishing gradients [40].

Beyond DBNs, RBMs have influenced various generative modeling frameworks, including Variational Autoencoders [42] (VAEs) and Generative Adversarial Networks [33] (GANs). VAEs learn latent representations by optimizing a variational lower bound on data likelihood, while GANs generate realistic samples through an adversarial game between a generator and a discriminator. While newer models such as VAEs and GANs have expanded the field of generative modeling, RBMs remain relevant due to their straightforward probabilistic interpretation and their robust performance in tasks requiring explicit probability modeling, such as collaborative filtering and recommender systems [3]. Their capacity to capture underlying data distributions makes them particularly valuable in domains where interpretability and modeling uncertainty are crucial.

Moreover, RBMs facilitate semi-supervised learning and dimensionality reduction, significantly impacting fields such as data augmentation, representation learning, and anomaly detection. This adaptability highlights RBMs' enduring relevance and positions them as foundational models within the broader landscape of artificial intelligence and deep learning research.

2.1.2 The Boltzmann Machine

The Boltzmann Machine (BM) is a generative neural network model that defines a probability distribution over binary-valued inputs using an energy-based framework. Originally introduced by Hinton and Sejnowski in the 1980s [41], the BM is designed for unsupervised learning and serves as a foundational model for more efficient variants, most notably the Restricted Boltzmann Machine (RBM) [3].

A standard Boltzmann Machine consists of a set of visible units that represent the observed data and a set of hidden units that capture latent dependencies. Unlike more structured models, the BM permits symmetric, undirected connections between all units, including connections within the same layer. This results in a fully connected network in which each unit can influence the state of every other unit, allowing the model to capture complex statistical dependencies.

The BM models a joint distribution over visible and hidden units using an energy function, where lower energy corresponds to more probable configurations. The probability of a particular state is defined by the Boltzmann distribution and normalized by a partition function that sums over all possible configurations. However, due to intra-layer connections and the exponential number of potential states, computing this normalization constant becomes computationally intractable for large networks [39]. As a result, training Boltzmann Machines typically relies on approximation techniques such as Gibbs sampling and other Markov Chain Monte Carlo methods [49].

Although the BM provides a powerful framework for modelling complex data distributions, its dense connectivity and slow convergence limit its scalability. These challenges lead to the development of the Restricted Boltzmann Machine, which simplifies the architecture by removing intra-layer connections and enables more efficient training. The RBM is discussed in detail in the following sections.

2.1.3 Overview of RBMs

The Restricted Boltzmann Machine (RBM) is a probabilistic, energy-based neural network model designed for unsupervised learning, feature extraction, and generative modeling. Originally introduced in the 1980s by Paul Smolensky and later popularized by Geoffrey Hinton in the early 2000s [39], the RBM builds upon the more general Boltzmann Machine (BM)—an undirected probabilistic graphical model—by imposing a bipartite structure between the visible and hidden layers. This architectural constraint significantly simplifies the training process. The resulting structure enables efficient learning of a joint probability distribution over visible and hidden units, making the RBM particularly useful for tasks such as dimensionality reduction, collaborative filtering, and representation learning [62, 3].

2.1.4 RBM Architecture

In a traditional RBM, there are two layers of nodes: a visible layer and a hidden layer. The visible layer corresponds to the observed data on which the model is trained, while the hidden layer captures underlying features and representations that the model learns. Every unit in the visible layer is connected to every unit in the hidden layer, but no connections exist between units within the same layer. This bipartite structure eliminates intra-layer dependencies, simplifying the computation of conditional probabilities during training.

The connections between visible and hidden units are associated with trainable weights, which are updated during the learning process. Additionally, each unit has an associated bias term that contributes to the overall energy configuration of the system.

Due to the RBM being an energy-based model, it uses an energy function to determine the probability of a particular configuration of visible and hidden units. Configurations that are more likely to occur in the training data are assigned a lower energy. This energy function is defined in Equation 2.1 [3].

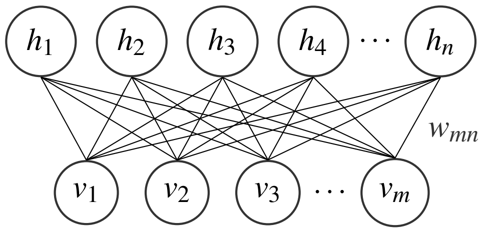


Figure 2.1: A simplified architecture of a RBM.

$$E(\mathbf{v},\mathbf{h}) = -\mathbf{v}^\top \mathbf{W}\mathbf{h} - \mathbf{b}^\top \mathbf{v} - \mathbf{c}^\top \mathbf{h} \tag{2.1}$$

where $\mathbf{v} \in \{0,1\}^n$ represents the visible unit vector, $\mathbf{h} \in \{0,1\}^m$ denotes the hidden unit vector, and $\mathbf{W} \in \mathbb{R}^{n\times m}$ is the weight matrix connecting these units. Additionally, $\mathbf{b} \in \mathbb{R}^n$ and $\mathbf{c} \in \mathbb{R}^m$ represent the bias vectors for the visible and hidden units, respectively. This energy function forms the foundation for defining a joint probability distribution over visible and hidden units, which is expressed in Equation 2.2 through the Boltzmann distribution.

$$P(\mathbf{v},\mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{v},\mathbf{h})} \tag{2.2}$$

where Z is the partition function, defined in Equation 2.3.

$$Z = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} \tag{2.3}$$

This partition function acts as a normalizing constant to ensure that the probabilities over all possible configurations add up to one. However, computing Z exactly is typically inadvisable, as it requires summing over an exponential number of configurations. As a result, approximate training algorithms, most notably Contrastive Divergence (CD) [39], are used to efficiently update weights and biases without evaluating the full partition function.

2.1.5 Training RBMs

Training a Restricted Boltzmann Machine (RBM) involves adjusting its parameters (namely the weights and biases) so that the probability distribution it defines assigns high likelihood to the observed data. In effect, the RBM is encouraged to favor configurations that resemble patterns in the training set, while assigning lower probability (or higher energy) to unlikely or unfamiliar data points.

The training objective is to maximize the log-likelihood of the training data. However, computing the exact gradient of this likelihood is computationally expensive due to the partition function in the denominator of the probability distribution. To address this, Hinton proposed an approximate method known as *Contrastive Divergence (CD)* [39], which is commonly used to train RBMs.

Contrastive Divergence approximates the gradient by comparing two expectations: one derived from the training data, and one derived from the model's reconstructions. The weight update rule for a weight w_{ij} between visible unit i and hidden unit j is given by:

$$\Delta w_{ij} = \alpha \left(\langle v_i, h_j \rangle_{\text{data}} - \langle v_i, h_j \rangle_{\text{model}} \right) \tag{2.4}$$

where α is the learning rate. The term $\langle v_i, h_j \rangle_{\text{data}}$ represents the expectation of the product v_i, h_j when the visible vector is mapped to the training data—this is referred to as the *positive phase*. The second term, $\langle v_i h_j \rangle_{\text{model}}$, is the corresponding expectation under the model's reconstructed distribution—this is the *negative phase*.

To approximate the negative phase, CD typically uses a single step of Gibbs sampling, a variant called CD-1. The process begins by sampling hidden units $\mathbf{h}$ given the input visible vector $\mathbf{v}$, reconstructing a new visible vector $\mathbf{v}'$ from $\mathbf{h}$, and then resampling hidden units $\mathbf{h}'$ from $\mathbf{v}'$. These samples are then used to compute the expectations in the update rule.

The biases are updated similarly, using the following rules:

$$\Delta b_i = \alpha \left(\langle v_i \rangle_{\text{data}} - \langle v_i \rangle_{\text{model}} \right) \tag{2.5}$$

$$\Delta c_j = \alpha \left(\langle h_j \rangle_{\text{data}} - \langle h_j \rangle_{\text{model}} \right) \tag{2.6}$$

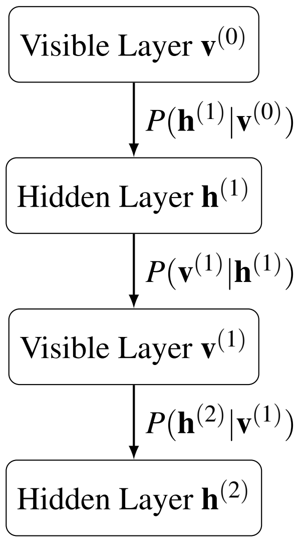
These updates encourage the RBM to reduce the energy of configurations observed in the data and increase the energy of configurations not supported by the data. This process allows the model to capture meaningful structure in the data and learn a useful latent representation.

While CD is effective and widely used due to its simplicity and computational efficiency, it is only an approximation of the true gradient. As such, it may not always converge to the global optimum. More advanced methods such as *Persistent Contrastive Divergence (PCD)* [60] have been introduced to address some of these limitations. PCD enhances traditional Contrastive Divergence [39] by maintaining a persistent set of Markov chains. These chains are continuously updated during training and serve to approximate the negative phase in the gradient calculation. PCD enables more stable learning and improved convergence, especially in deep or complex architectures.

2.1.6 Generative Capabilities of the RBM

One of the most powerful aspects of the Restricted Boltzmann Machine (RBM) is its ability to act as a generative model. After being trained on a dataset, the RBM captures the joint probability distribution between visible and hidden variables. This enables it to generate new data samples that resemble those in the training set, effectively simulating realistic inputs based on what it has learned [39, 40].

The generative process in an RBM typically involves running a Gibbs sampling process. Starting with a random or initial visible vector $\mathbf{v}^{(0)}$, the RBM alternates between sampling the hidden vector $\mathbf{h}^{(t)}$ given the current visible vector, and then sampling a new visible vector $\mathbf{v}^{(t+1)}$ from the hidden vector. This alternating process continues for several steps until the chain converges to the model's learned distribution, as seen in 2.2.


Visible Layer $\mathbf{v}^{(0)}$
$P(\mathbf{h}^{(1)}|\mathbf{v}^{(0)})$
Hidden Layer $\mathbf{h}^{(1)}$
$P(\mathbf{v}^{(1)}|\mathbf{h}^{(1)})$
Visible Layer $\mathbf{v}^{(1)}$
$P(\mathbf{h}^{(2)}|\mathbf{v}^{(1)})$
Hidden Layer $\mathbf{h}^{(2)}$

Repeat multiple times to generate a sample from the learned distribution.

Figure 2.2: Simplified view of Gibbs sampling in an RBM. Alternating updates between visible and hidden layers.

The conditional distributions used in Gibbs sampling are tractable due to the RBM's restricted structure, and are given by:

$$P(h_j = 1 \mid \mathbf{v}) = \sigma\left(\sum_i w_{ij} v_i + c_j\right) \tag{2.7}$$

$$P(v_i = 1 \mid \mathbf{h}) = \sigma\left(\sum_j w_{ij} h_j + b_i\right) \tag{2.8}$$

where

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{2.9}$$

is the logistic sigmoid function [13], **W** is the weight matrix, and **b** and **c** are the bias vectors for the visible and hidden layers, respectively.

After running enough steps of Gibbs sampling, the visible units can be interpreted as a new sample drawn from the distribution learned by the RBM. These synthetic samples often preserve the statistical properties and structure of the original training data. As a result, RBMs are particularly suitable for several tasks. They can be used for data imputation, by filling in missing values through sampling from the learned distribution. They also support data augmentation by generating new samples to expand limited datasets. In the context of outlier detection, RBMs allow for comparison between real data points and generated samples to identify deviations. Finally, they can facilitate feature visualization, offering insight into the kinds of patterns and structures the model has learned to represent.

Although RBMs are relatively simple compared to more modern generative models, such as Variational Autoencoders (VAEs) or Generative Adversarial Networks (GANs), they remain a foundational tool for learning the underlying structure of binary and categorical data. Their ability to model complex dependencies in an unsupervised fashion makes them an effective building block in deep belief networks and other hierarchical models.

2.1.7 RBMs for Outlier Detection

Outlier detection is a critical component in many real-world applications, such as fraud detection, system monitoring, and data cleaning, due to the need for reliable, high-quality data in downstream analysis. Traditional methods, as discussed in Section 2.2, often struggle when applied to high-dimensional or complex data. In this context, Restricted Boltzmann Machines (RBMs) offer a promising alternative due to their ability to model complex data distributions in an unsupervised fashion.

As generative models, RBMs learn the underlying joint distribution of the observed input data, allowing them to capture what is “normal” about the dataset. This capability enables RBMs to inherently identify inputs that deviate significantly from learned patterns, making them well-suited for outlier detection.

Several approaches are available to support outlier identification. Abnormal data points typically exhibit higher energy under the RBM’s energy function, indicating poor alignment with the learned distribution. Reconstruction error provides a similar signal—inputs that are difficult to reconstruct are likely to be anomalous and yield higher reconstruction errors.

These two metrics offer quantitative evidence that a given data point does not conform to the training data distribution. While exact likelihood estimation is generally intractable due to the partition function, approximation methods such as Annealed Importance Sampling can provide rough estimates, though at a high computational cost [49].

Compared to Autoencoders and Variational Autoencoders (VAEs), both of which are used for outlier detection, RBMs offer the advantage of explicitly modeling a probability distribution. Unlike deterministic encoders, RBMs provide a probabilistic framework that can better capture uncertainty in the data. Furthermore, RBMs operate in an unsupervised manner and do not require labeled data, making them suitable for anomaly detection in settings where labels are unavailable.

Despite these strengths, RBMs also have limitations. Their performance can be sensitive to hyperparameters such as the learning rate and the number of hidden units, and training can be computationally demanding, particularly for large datasets. Moreover, while their probabilistic formulation is advantageous, exact inference is generally intractable and requires approximation techniques such as Contrastive Divergence.

2.2 Outlier Detection Methods

The problem of detecting outliers in data remains a highly important area of research due to the significant impact outliers can have on results. An outlier is defined as any data point that does not exhibit the expected behavior shown by the majority of other points [63]. This deviation from expected behavior can skew results and introduce bias into any downstream analysis or application.

A variety of approaches exist for determining whether a data point is an outlier, each with strengths and weaknesses depending on the context in which they are applied [2]. These methods typically fall into four main categories: statistical, machine learning, clustering based, and distance-based techniques. While the approach proposed in this thesis belongs to the machine learning category, we briefly review several representative methods from the other categories.

2.2.1 Z-Score

A simple statistical approach to outlier detection is the use of the *z-score* test [8]. Introduced in the early 20th century, the z-score measures how many standard deviations a data point lies from the mean of the dataset. This value is then used to determine whether the point should be considered an outlier. Due to its ease of implementation—requiring only the mean and standard deviation—the z-score is often used as a quick method for detecting outliers. However, caution is advised when applying it to small sample sizes, as unstable estimates of the standard deviation may lead to incorrect confidence levels [23].

2.2.2 Interquartile Range

The *Interquartile Range (IQR)* is a statistical method that identifies the range covering the middle 50% of a dataset. A threshold is then defined beyond this range to determine whether a data point qualifies as an outlier. This threshold typically extends slightly beyond the IQR to avoid overly strict labelling. IQR-based detection can be less effective on small datasets, where the calculated quartiles may not be well-defined. However, the method offers the advantage of being simple to compute and computationally inexpensive, making it a suitable choice when quick outlier detection is required.

IQR has seen success in data mining applications due to its ease of use and its robustness to extreme values, which can otherwise skew the results of other statistical methods [24].

2.2.3 Isolation Forest

The *isolation forest* is an unsupervised machine learning technique that identifies outliers based on how easily a data point is isolated from the rest of the dataset. The algorithm operates by randomly selecting a feature from the dataset to use as a splitting criterion. A split value is then chosen randomly between the feature's minimum and maximum values, dividing the data into two branches. This process is repeated recursively, with each resulting subset undergoing the same splitting procedure until every data point is isolated in its own branch.

Outliers are identified based on the number of splits required for isolation. Points that are typical within the dataset generally require more splits to be separated, whereas outliers tend to be isolated with fewer splits [47]. Due to the inherent randomness in feature selection and split thresholds, several variations of the isolation forest have been proposed to improve its efficiency and overall effectiveness [5].

2.2.4 Autoencoder

An *autoencoder* is a neural network model that learns to reconstruct the data it is trained on, enabling the identification of outliers based on reconstruction performance. The input and output layers of the network have the same dimensionality, while the intermediate layers are first compressed (i.e., contain fewer neurons than the previous layer) and then expanded as the network approaches the output layer. This structure forces the model to learn internal representations that capture the essential characteristics of the input data [3, 7].

When applied to outlier detection, the autoencoder is trained on data that is assumed to be normal and largely free of outliers. New data is then passed through the trained model, and instances that are reconstructed with low error are considered normal. In contrast, data points that result in high reconstruction error are flagged as potential outliers [12].

2.2.5 k-Means

A popular clustering-based approach to outlier detection is the *k-means* algorithm. This algorithm follows a sequence of steps: first, a predefined number of clusters is selected, and initial cluster centres (centroids) are randomly assigned. Each data point is then assigned to the nearest centroid, after which new centroids are computed as the mean of the points within each cluster. Data points are reassigned to the updated centroids, and this process repeats until the clusters stabilize [19].

Variations of this approach, such as *k-medians*, use the median instead of the mean to compute cluster centroids. Outliers are typically identified as data points that lie far from their assigned centroid. A threshold distance—often defined as being *n* standard deviations away from the mean—is used to determine whether a point should be considered an outlier.

2.2.6 Mahalanobis Distance

Mahalanobis Distance (MD) is a distance-based approach to outlier detection that is commonly used with multivariate data due to its ability to handle high-dimensional spaces [31]. The method begins by computing the mean and the *covariance matrix* of the dataset. Then, for each data point, the Mahalanobis distance is calculated and compared to a pre-defined threshold to determine whether the point is an outlier. A key advantage of this approach is that it accounts for correlations between variables, as the covariance matrix captures the relationships among the features in the data.

2.3 Related Works

Outlier detection is extensively studied across various domains, including fraud detection, network intrusion detection, and medical diagnosis. Traditional methods such as z-score, Interquartile Range (IQR), and Mahalanobis Distance remain widely used due to their simplicity and efficiency [23, 24, 31]. However, these techniques often struggle with high-dimensional or complex datasets where underlying relationships between features are nonlinear or unknown.

To address these limitations, machine learning models such as Isolation Forests [47], Autoencoders [7], and clustering-based approaches like k-means [19] are proposed. These models learn more flexible representations of the data but often rely on assumptions about data distribution or require significant hyperparameter tuning. More recently, advances in deep learning have led to the emergence of deep anomaly detection techniques that leverage the representational power of neural networks [51].

The Restricted Boltzmann Machine (RBM) is explored in anomaly detection tasks due to its generative nature and its ability to model complex data distributions in an unsupervised manner. Studies such as [27] demonstrate that RBMs can effectively capture normal data patterns and identify anomalous points based on reconstruction error or free energy. Extensions like Deep Belief Networks (DBNs) and Deep Boltzmann Machines (DBMs) are also applied to anomaly detection in structured data [65, 30].

While prior work demonstrates the potential of RBMs for detecting anomalies, few studies investigate their use for generating synthetic samples for comparison or assess their performance across both binary and categorical datasets using multiple encoding strategies. Similarly, ensemble-based approaches that combine multiple RBMs—either as disjoint learners or stacked architectures—remain under explored in the context of outlier detection. This thesis builds on existing work by proposing a generative, RBM-based framework that is systematically evaluated across different data types and architectural configurations.

2.4 Data Encoding Techniques

Due to the structure of the Restricted Boltzmann Machine, any data that is not in a binary format needs to be processed before it can be used. This processing of converting the form of the data while retaining the information is called *data encoding*. Various data encoding techniques exists, with each having advantages and disadvantages associated. For the work that will be discussed in Chapter 4, we will briefly explore the *categorical encoding* techniques of transforming categorical data into binary [43].

2.4.1 One-hot Encoding

One-hot encoding is a simple categorical encoding technique based on each unique value of an attribute having it's own vector. This vector contains the same number of positions as there are valid values of the attribute in the original data. Each vector then has only one bit as ”1” and the others are all ”0”. This encoding scheme suffers when used with high dimensionality data, as the increase in feature space can be large when working with many categories [32]. Guo and Berkhahn [34] propose an alternative approach using entity embeddings that capture semantic similarity between categories in dense vector spaces.

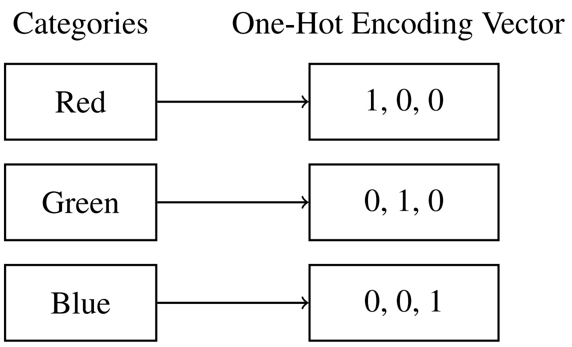


Figure 2.3: Converting categorical data into an one-hot encoding.

2.4.2 Binary Encoding

Binary Encoding is a categorical encoding strategy that converts the data into a more compact binary representation. Instead of the resulting vector having the same number of positions as number of values of an attribute, we get a binary number that relates to which value the original attribute had. To achieve this, each categorical feature is transformed into a numerical value using a ordinal encoder, then converted into its binary representation. While this method does reduce the interpretability of the encoded data, the benefit of reducing the dimensionality of the data that is then fed to a model makes it very useful [58].

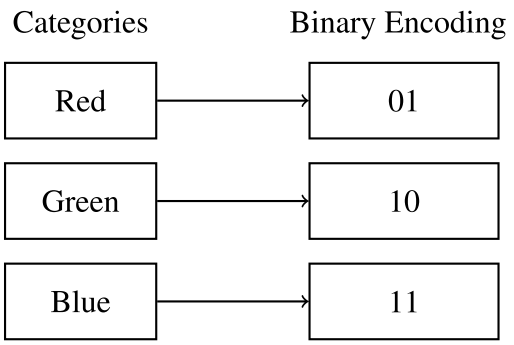


Figure 2.4: Converting categorical data into a binary encoding.

2.5 Ensemble Learning Methods

In machine learning, a common issue is that models can over fit and begin memorizing the training data and performing poorly on unseen data. To address this, ensemble methods combine multiple models, known as *base learners* [66], to improve accuracy and reduce variance or bias. By aggregating the outputs of multiple base detectors, these methods reduce the variance associated with individual models. This is a key advantage of ensemble learning [25], which has already shown effectiveness in outlier detection tasks [45]. Aggarwal [1] presents several strategies for combining weak outlier detectors and demonstrates their effectiveness across diverse datasets.

2.5.1 Boosting

One approach to ensemble learning is Boosting [36]. This technique trains models in a sequential fashion, where the next model learns to address the mistakes that are present in the one before. If the model does misclassify a portion of the training data, the weight on that data is increased to allow the next model to focus its attention on data with higher weights as they are harder to learn examples. The process of training the next model continues until the entire training dataset has been predicted accurately, or a limit of models is reached. Due to the number of models that potentially may be used, this method can become computationally expensive, and more difficult to tune hyper-parameters. A popular approach to boosting ensemble learning is the Gradient Boosting algorithm [29].

2.5.2 Stacking

Another way to combine the learning abilities of models is through stacking [64]. Each model's output is then used as training for a top level model that uses these predictions to make a final decision. These base learners can be of the same type of model, or be different to diversify the training. This method can be used with different types of base learners to allow more complex relationships in the data be identified and learned. A diverse selection of models also has the additional benefit of lower bias that a specific type of model may inherently posses. Recommendation systems use this approach to blend different machine learning models to achieve better results [4].

2.5.3 Bagging

Bootstrap Aggregation [16] (or bagging) is an easy to use yet powerful ensemble learning technique that limits the data that each model is trained on. The various base learning models are each trained on an different, yet equal in size, subsets of the training data to help eliminate over fitting. Learning happens in parallel, so each model is independent from each other. This approach can achieve higher accuracy when compared to a standalone model as the composite model reduces the variance of the individual base learner [36].

2.6 Summary

This chapter provided a comprehensive overview of essential concepts and techniques relevant to our research. It began by summarizing commonly used outlier detection methods, followed by a discussion of existing research that applies these methods, including approaches utilizing Restricted Boltzmann Machines (RBMs). The chapter then detailed the theoretical foundations of RBMs, emphasizing their particular relevance for outlier detection tasks.

Additionally, various data encoding techniques were examined, highlighting methods for transforming categorical datasets into binary formats to ensure compatibility with RBM-based models. The importance of encoding in enabling effective data processing and pattern learning was underscored. Lastly, ensemble methods were explored, illustrating how combining multiple models can significantly enhance the accuracy and overall performance of the outlier detection system.

Chapter 3

Restricted Boltzmann Machine based Outlier Detection

3.1 Overview of Our Approach

In Chapter 2, we have explored the theoretical foundations of Restricted Boltzmann Machines (RBMs) in outlier detection. This chapter presents the original methodology developed in this research for unsupervised outlier detection using RBMs, RBM-OD. Instead of analyzing raw data directly, our approach detects anomalies through RBMs by comparing real data samples with synthetic data generated from the learned distribution.

This work introduces a novel generative outlier detection framework that relies on the RBM's ability to model the underlying data distribution and produce representative synthetic samples. The core idea is to train RBMs on preprocessed binary or encoded categorical data, and then use the learned model to generate data points that reflect typical patterns. This process is represented by the first box in Figure 3.1. By comparing these generated samples to real inputs, outliers can be identified as those that diverge significantly from the expected distribution. These steps are encompassed by the second box of Figure 3.1. Algorithm 1 shows the process of our RBM-based data generation.

Our methodology is distinguished by the integration of several unique design elements. First, we conduct a comparative study of four RBM variants: single-layer RBMs, stacked RBMs, ensemble of RBMs, and ensembles of stacked RBMs. This systematic evaluation provides an insight into how architectural choices impact detection effectiveness and generative fidelity. Second, we implement validation mechanisms for generated samples,

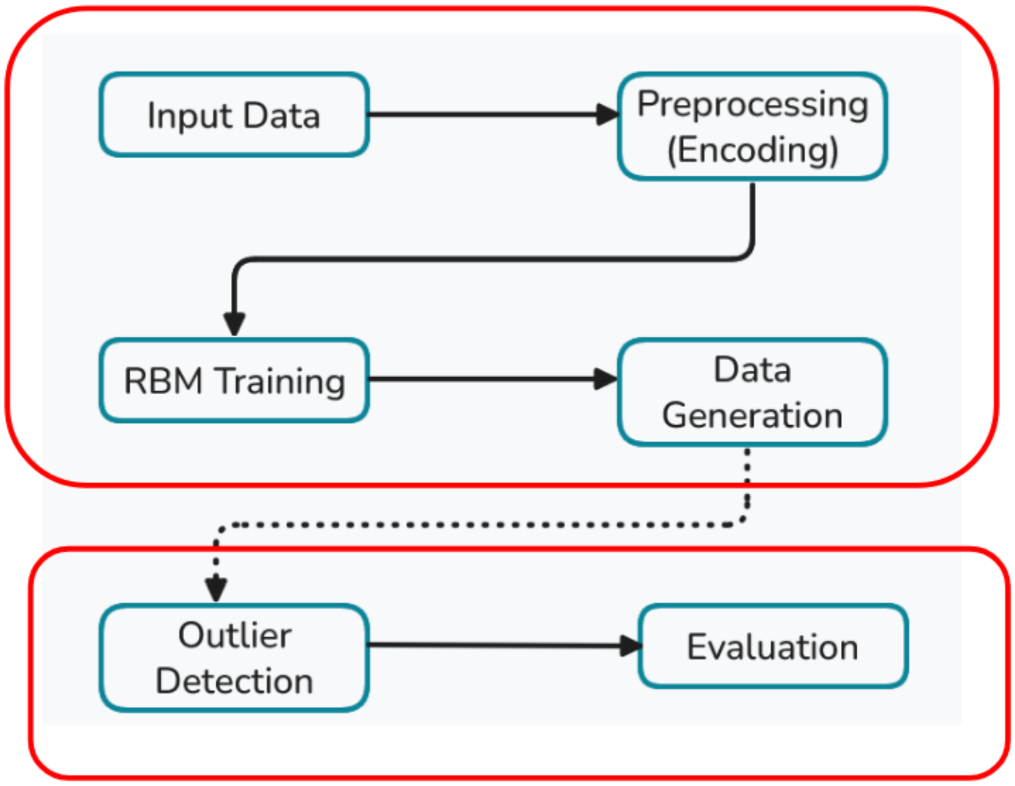


Figure 3.1: Overview of the key stages of RBM-OD.

ensuring that data derived from categorical inputs adheres to encoding constraints. This preserves semantic integrity and prevents invalid outputs from influencing results.

Finally, the detection process combines both a distance-based strategy, using Hamming distance, and a model-based approach via the RBM energy function. These scoring mechanisms are applied together as seen in Algorithm 2 and their outputs are fused using fusion and intersection thresholds to enhance robustness. To assess model quality, we adopt a suite of evaluation techniques appropriate to the nature of the data, including reconstruction error, encoding validity, structural constraint preservation, and anomaly score analysis.

3.2 Data Preparation

Before training, all input data is converted into a binary format that can be used for processing by the Restricted Boltzmann Machine. This preparation step differs depending on the nature of the data and whether it is already in a binary form or categorical data.

Algorithm 1 RBM-based model training and synthetic data generation.

```
Require: Initial training dataset $D$
Ensure: Generated dataset $D_{gen}$
 1: if $D$ is categorical then
 2:     Encode $D$ using one-hot or binary encoding
 3: end if
 4: Split $D$ into training and test sets
 5: Select model variant: single RBM, ensemble of RBMs, stacked RBM, or ensemble of
    stacked RBMs
 6: for
 7:      doTrain a single RBM on the training data
 8:     Generate synthetic samples using Gibbs sampling
 9: end for
10: for using ensemble of RBMs do
11:     Partition training data into $n$ subsets (with or without overlap)
12:     for each subset do
13:         Train an independent RBM on the subset
14:     end for
15:     Aggregate synthetic samples from all ensemble models
16: end for
17: for using stacked RBM do
18:     Train first RBM on training data
19:     for each additional layer do
20:         Generate data from RBM
21:         Train next RBM on generated data
22:     end for
23:     Generate synthetic samples using the final RBM
24: end for
25: for using ensemble of stacked RBMs do
26:     Partition training data into $n$ subsets
27:     for each subset do
28:         Train a stacked RBM on the subset:
29:         Train first RBM, generate data, and continue stacking
30:     end for
31:     Generate and aggregate synthetic samples from final RBM of all stacked models
32: end for
```

Algorithm 2 RBM-based outlier detection.

```
Require: Dataset $D$ and generated dataset $D_{gen}$
Ensure: Set of detected outliers $O$
 1: Inject 50 outliers into $D_{gen}$ for testing
 2: Compute free energy for each sample in $D$ using trained RBM(s)
 3: Compute Hamming distance between each sample in $D$ and samples in $D_{gen}$
 4: Set thresholds for free energy and Hamming distance (e.g., percentile-based)
 5: Initialize empty set $O$
 6: for each sample in $D$ do
 7:     if free energy of sample exceeds threshold OR Hamming distance exceeds threshold then
 8:         Add sample to $O$
 9:     end if
10: end for
11: return $O$
```

For datasets that are already in binary form no additional transformation is required. These are used directly as input to the RBM. For categorical datasets however, encoding is necessary to represent each feature in binary form. In this work, two distinct encoding methods are applied: one-hot encoding and binary encoding. The theory for these encoding strategies are discussed in Chapter 2, and we will discuss the details for our approach in Section 4.3.1.

Both encoding strategies are applied to the same datasets in order to compare their impact on the performance of the RBM models. This comparison is essential to the methodology due to the effects each have on the ability of the model to learn the data distribution.

No normalization or additional preprocessing is applied. As RBMs operate directly on binary input, there is no need for scaling or centreing. This minimal preprocessing approach aligns with the probabilistic, unsupervised nature of the RBM and helps preserve the structural integrity of the original datasets.

3.3 Variants of our RBM-based framework

This section outlines the architecture and training configurations of the Restricted Boltzmann Machines (RBMs) used in this study. Multiple RBM variants are explored to investigate their effectiveness in learning data distributions and detecting outliers. These include a single RBM, stacked RBMs, ensembles of single RBMs, and ensembles of stacked RBMs.

3.3.1 Single RBM

The single RBM is the base model, which will be referred to as M_S. It consists of one layer of visible units and one layer of hidden units, connected symmetrically. This can be seen in Figure 3.2. It is trained using *Persistent Contrastive Divergence (PCD)*, which will be further discussed in the next section. Typical values used for the number of hidden units range from 50 to 500, depending on the dataset dimensionality.

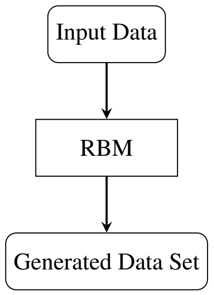


Figure 3.2: A single RBM is trained on the input data, and used to generate a synthetic data set.

3.3.2 Stacked RBM

To deepen the representational capacity of the model, a stacked RBM is constructed by training multiple RBMs. This variation will be referred to as M_{ST}. The generated output of one RBM becomes the input for the next RBM's visible layer, as seen in Figure 3.3. Each layer is trained independently, allowing the model to capture hierarchical features in the data. Stacking RBMs has been shown to enhance feature extraction and modelling power. Larochelle and Bengio [44] demonstrate this in a classification setting, where each RBM layer contributes to a richer representation of the input data.

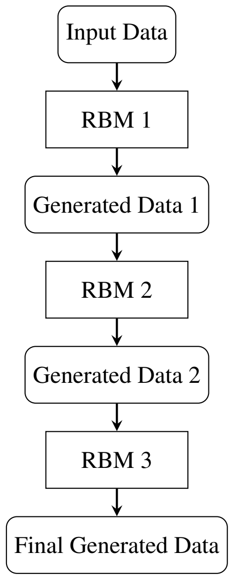


Figure 3.3: A stack of RBMs where each successive RBM is trained on the generated data of the one before.

3.3.3 Ensemble of Single RBMs

To introduce model diversity and reduce variance, multiple independent RBMs are trained on the same dataset with different initial weights and training splits. We refer to this variation as M_E. Their outputs are combined before the evaluation phase, creating a dataset that contains generated data points from all the models. This variation can be seen in Figure 3.4.

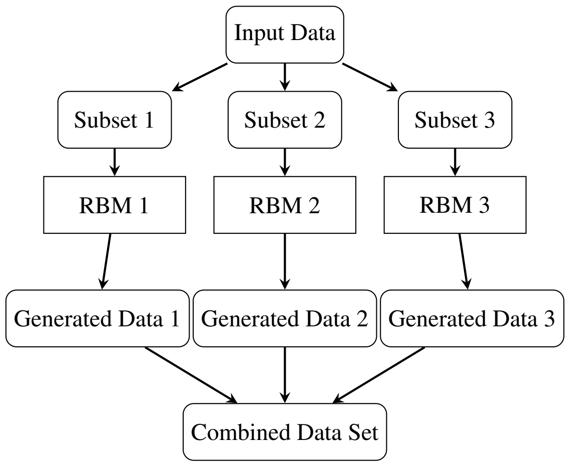


Figure 3.4: An ensemble of RBMs are each trained on different portions of the same dataset, and each generate their own synthetic data that is combined together.

3.3.4 Ensemble of Stacked RBMs

This variant combines the representational depth of stacked RBMs with the diversity that ensemble learning provides. Multiple stacked RBMs are trained independently, and their outputs are aggregated to improve generalization and reduce sensitivity to initializations or over fitting. This variation is named M_{EST} and can be found in Figure 3.5.

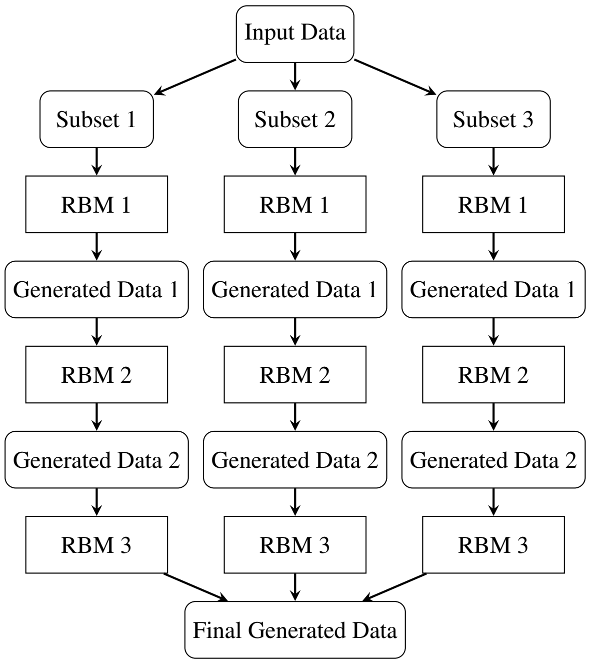


Figure 3.5: An ensemble of chained RBMs.

3.4 RBM Data Generation

Each variant is kept consistent across experiments, with architectural modifications made to best suit the dataset that is used. This allows a more direct comparison to be made when comparing each variant with each other. After training, each RBM model is used to generate synthetic datasets by sampling from the learned probability distribution, as shown in Algorithm 3. The synthetic data serves two main purposes: 1) evaluate the model's ability to replicate the underlying structure of the original data, and 2) facilitate outlier detection by comparing real and synthetic data samples.

The generation process begins by randomly initializing the visible layer with binary val-

Algorithm 3 Data generation algorithm.

```
Require: Trained RBM, Gibbs steps $k$, samples $N$
Ensure: Generated dataset $\mathcal{D}_{gen}$
 1: Initialize empty dataset $\mathcal{D}_{gen}$
 2: for $n = 1, 2, \ldots, N$ do
 3:     Randomly initialize visible units
 4:     for $t = 1, 2, \ldots, k$ do
 5:         Sample hidden units from visible units
 6:         Sample visible units from hidden units
 7:     end for
 8:     Store final visible units in $\mathcal{D}_{gen}$
 9: end for
10: return $\mathcal{D}_{gen}$
```

ues. From this random starting point, 1000 Gibbs sampling steps are performed to allow the network to approximate the equilibrium distribution learned during training (see Chapter 2 for more details).

3.5 Outlier Detection via Generative Modelling

This section outlines the core of our proposed methodology: using the generative capabilities of Restricted Boltzmann Machines (RBMs) to detect outliers in binary and categorical data. Rather than evaluating input data through reconstruction or clustering, our approach leverages the RBM's learned distribution to generate synthetic samples that represent typical data. By comparing real data samples to the generated samples, we can identify outliers as data points that deviate significantly from the learned norm.

3.5.1 Outlier Detection Strategies

After the models are trained, each RBM model is used to generate synthetic data through Gibbs sampling. These samples reflect what the model has learned as "normal" and are used as a baseline for comparison against real test data. Outliers are detected using two complementary scoring mechanisms:

1. **Hamming Distance (Distance-Based Detection):** For each real test sample, we

calculate the Hamming distance [35] to every generated sample and retain the minimum value. A high minimum distance suggests that the sample is unlike anything the model considers typical, and thus may be anomalous.

2. Free Energy (Model-Based Detection): The free energy of each real sample is also computed under the trained RBM, and can be found in Section 2.1.4. Samples with high free energy are assigned low likelihood by the model, indicating that they are statistically unusual or out-of-distribution [39].

To identify outliers, percentile-based thresholds are applied independently to both score types. For our experiments, the top 10% of samples with the highest Hamming distances or energy values are flagged as outliers to help balance over detection with minimizing false positives. This flexible threshold approach eliminates the need for fixed cutoffs or prior knowledge of anomaly rates.

3. Decision Fusion and Detection Rationale: By combining both detection strategies, we form a more comprehensive outlier detection system. In practice, the results from energy-based and distance-based methods can be analyzed separately or fused using logical or (which we refer to as fusion in this work) to increase sensitivity or intersection to improve specificity. This dual-scoring framework provides robustness to a wide variety of anomaly types and ensures that the strengths of each scoring method are fully utilized. We also report results where both energy and distance detection identify the same data point as an outlier; these are shown in the *Intersect* column.

Ultimately, this strategy re-frames the RBM from a feature learner or dimensionality reducer into a generative anomaly detector, an approach that is central to the contributions of this research.

3.6 Evaluation Criteria for RBM-OD

The final stage of the methodology involves evaluating the performance of the RBM-based models for generating data and in detecting outliers. Since the task is conducted in an unsupervised setting without ground truth labels, traditional supervised evaluation metrics such as precision, recall, and F1-score could not be directly computed [18, 67]. Therefore, alternative evaluation strategies are employed.

For data generation, there are a few tests that we use. For all datasets, Reconstruction mean squared error (MSE) [9] provides a measure of how well a trained Restricted Boltzmann Machine (RBM) is able to reproduce its input data. After training, each model attempts to reconstruct the visible layer from its learned hidden representations. The reconstruction MSE is then computed as the average squared difference between the original input and the reconstructed output, with lower values indicating better preservation of the original data structure. In this work, MSE is evaluated on both the training set and a held-out test set, allowing assessment of both model fit and generalization. MSE is particularly meaningful for evaluating models trained on binary or one-hot encoded data, where even small deviations from 0 or 1 can imply structural inconsistency. Notably, MSE scores are only reported for models where reconstruction is defined directly in terms of the visible layer(M_S and M_E). For M_{ST} and M_{EST}, where subsequent RBMs are trained on generated data rather than on intermediate representations, end-to-end reconstruction is not defined, and therefore MSE is omitted.

For two of the binary datasets used, average Hamming distance is employed to assess the similarity between the generated data and the original training data. The Hamming distance measures the proportion of differing bits between two binary vectors and is particularly suitable for evaluating binary-encoded structures. In this context, a low average Hamming distance indicates that the generated samples closely resemble the training distribution while still allowing for minor variability, which is desirable in a generative model. Conversely, a high average Hamming distance suggests that the generated data significantly deviates from the training data, potentially indicating poor model convergence or structural degradation during sampling.

For the third binary dataset, several features are intentionally constructed with known mathematical dependencies to enable structural validation of the generated data. These relationships included linear combinations, inversions, and averages between features (e.g., one feature being half of another, or the average of two others). After training, the RBM's ability to preserve these dependencies is assessed by computing the average deviation between the expected and actual values for each constructed relationship in the generated samples. This method provided an interpretable measure of how well the generative model captured the underlying structure of the data, beyond simple distributional similarity. Lower average deviations indicated stronger structural fidelity, while higher deviations signaled a breakdown in the learned relationships. This approach is particularly useful for evaluating continuous-valued data where structural constraints are explicitly defined.

For categorical datasets, the structure imposed by the encoding schemes enabled a clear mechanism for evaluating the quality of generated data. Specifically, the generated samples could be assessed for adherence to encoding constraints that define valid categorical representations. For one-hot encoding, a valid sample requires exactly one active bit per encoded category, whereas invalid samples may contain multiple active bits or none at all. Similarly, binary encoding imposes its own structural rules based on bit patterns representing distinct categorical values. By checking for violations of these constraints, it is possible to quantify how many generated data points are structurally valid versus invalid. This provided a discrete and interpretable metric for assessing the fidelity of the generative model when applied to structured categorical data.

Next, we asses how many outliers are detected from the models used. The distributions of the anomaly scores using both Hamming distance and energy functions are analyzed. By inspecting the spread and behaviour of these scores, it is possible to identify whether a clear distinction existed between typical data points and potential outliers. In particular, the presence of a long-tailed distribution, where a small proportion of samples exhibited significantly higher outlier scores, is interpreted as evidence that the model successfully distinguished between normal and non-normal samples.

Second, the number and characteristics of samples flagged as outliers are studied. The flagged samples are examined to check for consistency and plausibility, considering the dataset context. In datasets where certain points are known or suspected to be unusual, it is verified whether these samples are among those detected as outliers.

Third, comparisons are made across the different RBM model architectures used, single RBM, ensemble of RBMs, stacked RBMs, and ensemble of stacked RBMs. The number of detected outliers and the nature of their anomaly scores are compared to evaluate the impact of model complexity and diversity on detection performance.

A comprehensive evaluation of the RBM-based outlier detection approach is conducted by combining analysis of anomaly score distributions, examination of detected samples, cross-model comparisons, and assessment of generated data quality. Together, these components provided insight into both the effectiveness of outlier identification and the model's ability to faithfully replicate the underlying data structure.

3.7 Summary

This chapter detailed the methodology employed for developing and evaluating an outlier detection framework based on Restricted Boltzmann Machines (RBMs).

Initially, the input datasets are preprocessed to ensure binary representations, either by directly using binary datasets or by encoding categorical data through one-hot or binary encoding methods. Following preprocessing, datasets are split into training and testing subsets.

The training phase involves fitting various RBM architectures to the training data, including single RBMs, ensembles of RBMs, stacked RBMs, and ensembles of stacked RBMs. The RBMs are trained using PCD, with hyperparameters such as learning rate and number of hidden units selected based on experimental considerations.

Outlier detection is carried out by comparing real test samples to synthetic data, using both Hamming distance and energy-based scoring methods. Samples are flagged as outliers based on whether their scores exceeded predefined thresholds, and a union strategy is used to combine the two scoring methods for final detection.

Finally, the results are evaluated by analyzing the distributions of anomaly scores, examining the characteristics of detected outliers, and comparing the performance across different RBM model variants.

Chapter 4

Empirical Evaluations of RBM-based Outlier Detection

This chapter presents the experimental setup and results for evaluating the performance of the Restricted Boltzmann Machine (RBM) in detecting outliers. The experiments are structured around two types of data: binary data and categorical data. Each type is discussed in its own section, covering the datasets used, the experimental procedure, and the results.

The goal of these experiments is to evaluate how effective our proposed RBM models can learn the underlying data structure and detects outliers. To assess the models' ability to learn the data's patterns and relationships, several tests are applied, varying by the datasets. Detection performance is evaluated using two primary methods: an energy-based approach relying on the model's free energy, and a distance-based approach using Hamming distance. These methods are detailed in Section 3.5.1. Synthetic outliers are injected into the datasets to simulate anomalous patterns and enable controlled evaluation.

The remainder of this chapter is divided into three sections. First, Section 4.1 describes the setup used for the experiments, including hardware, the software packages used, and the custom pipeline used to run all experiments. Section 4.2 discusses experiments involving binary data, while Section 4.3 focuses on experiments using categorical data.

4.1 Experiment Setup

4.1.1 Hardware

All experiments were conducted on a personal workstation equipped with an AMD Ryzen 9 7950X3D processor, 32 GB of DDR5 RAM, and an NVIDIA GeForce RTX 4070 Ti SUPER graphics card. Although the base operating system is Windows 10, all experimental tasks were carried out within a Linux environment using Ubuntu 22.04 via the Windows Subsystem for Linux (WSL). Despite the presence of a high-performance GPU, model training is performed on the CPU due to the initial setup of WSL, as well as the package used for the RBM implementation only supporting CPU computation. This hardware configuration provided sufficient computational power to support the training and evaluation of various RBM-based models, including single RBMs, stacked architectures, and ensemble methods across multiple datasets.

4.1.2 Software

We used Python 3.10.12[1] for all experiments. The scikit-learn [2] library is used extensively, including the `BernoulliRBM` module for model training, `OneHotEncoder` for one hot data encoding, and `train_test_split` for partitioning datasets into training and testing subsets. For binary and alternative encoding schemes, the `category_encoders` (ce) library is employed, providing a range of encoding techniques compatible with scikit-learn workflows. Numerical computations are supported by NumPy[3], while pandas facilitated data loading and preprocessing. Categorical datasets such as Car Evaluation[14] and Nursery[52] are retrieved using the `ucimlrepo` library, which offers standardized access to datasets from the UCI Machine Learning Repository.

[1]`https://www.python.org/downloads/release/python-31012/.`
[2]`https://scikit-learn.org/stable/.`
[3]`https://numpy.org/.`

4.1.3 Our Implementation

To conduct the experiments in this thesis, a customized Python program was developed to manage the full experimental workflow, including data preprocessing, model training, outlier injection, synthetic data generation, and evaluation. The code base is modular in structure, with distinct functions and scripts handling tasks such as dataset loading, categorical encoding (one-hot or binary), RBM training, Gibbs sampling for synthetic data generation, and threshold-based outlier detection.

The program supports multiple experimental variants, including single RBMs, stacked RBMs, ensembles of RBMs, and stacked ensembles. It includes built-in utilities for injecting synthetic outliers using different strategies (e.g., inversion or constraint violations) and for computing evaluation metrics such as reconstruction error, Hamming distance, and energy-based scores.

Threshold sweeps for energy and distance detection methods are automated, with options to record true positives and detection counts at specified percentiles. The results of each run are saved for later aggregation and comparison. The program is designed to be extensible and reproducible, allowing for the rapid testing of new models or datasets with minimal changes to the core logic.

4.2 Outlier Detection using Binary Data

This section presents the experiments conducted using binary data. The goal is to evaluate how well the RBM performs in identifying outliers when trained on binary inputs. Three datasets are used: one real-world dataset and two synthetic datasets, each designed to test different aspects of the RBM's ability to learn and generalize binary patterns. The results are analyzed using both the energy-based and distance-based outlier detection approaches described in Chapter 3. Each dataset and its associated results are discussed in the subsections that follow.

4.2.1 Datasets

The experiments on binary data are conducted using one real-world dataset and two synthetic datasets designed to reflect different data distributions and groupings of patterns. These datasets, as summarized in Table 4.1, are suited for RBM input without further encoding, allowing direct evaluation of the model's outlier detection performance on binary features.

Table 4.1: Summary of binary datasets used in experiments.

Dataset	**Samples**	**Binary Attributes**	**Missing Values**
D_{GRO}	10,000	30	None
D_{SPE}	267	22	None
D_{NOR}	10,000	10	None

The SPECT Heart (D_{SPE}) dataset [22] is a publicly available medical dataset containing 267 samples with 22 binary attributes derived from cardiac Single Proton Emission Computed Tomography (SPECT) images. The attributes indicate the presence or absence of specific cardiac features used for diagnostic classification, all of which are labelled as $F1, F2, \ldots, F22$

The synthetic grouping dataset (D_{GRO}) is generated with 10,000 samples, with 30 features in each sample. Each sample is drawn from one of three predefined configurations of binary features, where each configuration represents a distinct "grouping" of related attributes. For instance, one configuration may take the form:

```
[1111100010 0011000000 0000000011]
```

another as:

```
[0000011001 0000111100 0001110000]
```

and a third as:

```
[0000000100 1100000011 1110001100]
```

This size is selected to ensure adequate representation of each group's characteristic binary patterns, enabling the Restricted Boltzmann Machine to effectively learn the underlying structure.

To provide the RBM with a learnable yet structured continuous dataset, a synthetic dataset D_{NOR} was generated using samples drawn from clustered multivariate normal distributions. This type of experimentation was enabled due to the RBM package implementation, which permits continuous values constrained to the range $[0,1]$. Specifically, data points are created using the `make_blobs` method to simulate three distinct Gaussian clusters in a two-dimensional space, after which the features are normalized to the $[0,1]$ range using min-max scaling. By constraining the input values to a bounded interval, the RBM can still process continuous-valued inputs in a manner analogous to probabilities, allowing the model to learn smooth variations in the data distribution rather than only discrete binary activations.

Two base features, f_1 and f_3, are first sampled from Gaussian distributions and serve as the foundation of the dataset. From these, additional features are derived to introduce explicit structural dependencies. Specifically, f_2 is defined as a linear transformation of f_1 ($f_2 = 0.5 \cdot f_1$), while f_4 is constructed as the complement of f_3 ($f_4 = 1 - f_3$). Another dependent feature, f_6, is calculated as the average of the two base features ($f_6 = \frac{f_1+f_3}{2}$). To introduce variability and background noise, the remaining features (f_5, f_7 through f_{10}) are independently sampled from a Gaussian distribution over $[0,1]$. This design embeds both deterministic constraints and random variability, providing a robust basis for evaluating the RBM's generative behaviour and detection performance.

4.2.2 Experiment Parameters

The hyperparameters used to train each RBM are tuned individually for each dataset and encoding scheme to optimize data generation quality and model stability. These values are determined empirically through preliminary experiments and held consistent across all

Table 4.2: Training hyperparameters for binary datasets.

Dataset	**Learning Rate**	**Hidden Units**	**Epochs**	**Gibbs Steps**
D_{GRO}	0.001	100	100	1000
D_{NOR}	0.005	200	300	1000
D_{SPE}	0.001	100	100	1000

model architectures applied to a given dataset-encoding pair. Table 4.2 summarizes the learning rate, number of hidden units, training epochs, and Gibbs sampling steps used in each variation.

For all binary datasets, four different RBM-based model architectures are evaluated to assess their effectiveness in outlier detection. These include M_S (single RBM), M_E (ensemble of RBMs), M_{ST} (stacked RBM), M_{EST} (ensemble of stacked RBMs), as discussed in Chapter 3. Each variation is trained exclusively on pure data, with no exposure to outliers during training.

Each RBM variant generates a master dataset that is used to conduct our experiments on. For M_E, each RBM model generates a final dataset that are combined together into one master dataset. To reduce redundancy, duplicate data points are removed. We practice the same procedure with M_{EST}, with the RBMs at the end of each stack contributing to the final dataset.

As shown in Table 4.2, all RBM models are trained with consistent learning rates and epoch settings, which are then adjusted as needed for each dataset. Persistent Contrastive Divergence is applied with 500 Gibbs sampling steps per iteration during data generation.

Outliers are synthetically introduced to evaluate detection performance using two different approaches in all datasets except for D_{NOR}. First, outliers are generated using random binary vectors drawn from a Bernoulli distribution with $p = 0.5$, producing fully unstructured and patternless samples. We term this type of outlier, *Random Noise Outlier*. The other type of outliers are created by inverting the feature-wise means of the training data to produce rare but structured patterns. While these outliers may appear statistically unusual, their structure may still conform to patterns seen during training, offering a more subtle type of outlier to detect. This outlier type wll be refered to as *Anti-feature outliers*.

To evaluate the RBM's ability to detect potential outlier data for D_{NOR}, structured outliers are synthetically injected into the test dataset. These outliers are generated by deliberately violating the known feature relationships embedded in the normal data. Specifically, in each outlier sample, the dependent features f_2, f_4, and f_6 are assigned random values drawn uniformly from $[0,1]$, thereby breaking the constraints $f_2 = 0.5 \cdot f_1$, $f_4 = 1 - f_3$, and $f_6 = \frac{f_1+f_3}{2}$. The remaining features are left unchanged to ensure that the outliers only differed structurally in specific, measurable ways. This controlled approach to outlier injection provided a framework for evaluating the RBM's detection capabilities based on reconstruction error, energy score, and constraint violations, while ensuring that outliers are well-defined and reproducible.

As seen in the upcoming tables, *Avg. Hamming Distance* refers to the average Hamming distance of the models in each variant (M_S just uses the Hamming distance from the only RBM used, while M_E takes an average from all models in the ensemble).

4.2.3 Results, Analysis and Discussion

The performance of the RBM-based outlier detection method is evaluated on binary datasets containing 50 injected artificial outliers. The evaluation are based on the model's ability to detect these known anomalies using energy-based and distance-based detection techniques. In this section, we present the quantitative results, visualize key score distributions, and discuss the effectiveness and challenges of each approach.

Evaluation of Dataset D_{GRO}

Table 4.3 presents the data generation quality metrics for the D_{GRO} dataset across all four model variations. M_S produced the lowest reconstruction error, with a training MSE of 0.0191 and a test MSE of 0.0201, alongside an average Hamming distance of 0.0015. This indicates a highly faithful reproduction of the input data. In contrast, M_E yielded signifi-

Table 4.3: Generation quality metrics for D_{GRO}.

Model	**Train MSE**	**Test MSE**	**Avg. Hamming Distance**
M_S	0.0191	0.0201	0.0015
M_E	0.2064	0.2174	0.1202
M_{ST}	—	—	0.0022
M_{EST}	—	—	0.0017

cantly higher reconstruction errors, with a mean test MSE of approximately 0.2174, and a notably larger Hamming distance of 0.1202, suggesting poorer performance in the generated synthetic data despite aggregation. For the M_{ST} and M_{EST} models, full reconstruction is not computed; instead, generation quality is assessed through Hamming distance alone. These models achieved comparably low average Hamming distances (0.0022 for M_{ST} and 0.0017 for M_{EST}), indicating that the generated data remained structurally close to the original inputs even without full-stack reconstruction. This suggests that stacking can preserve critical structure in the data while relying on higher-level representations for generation.

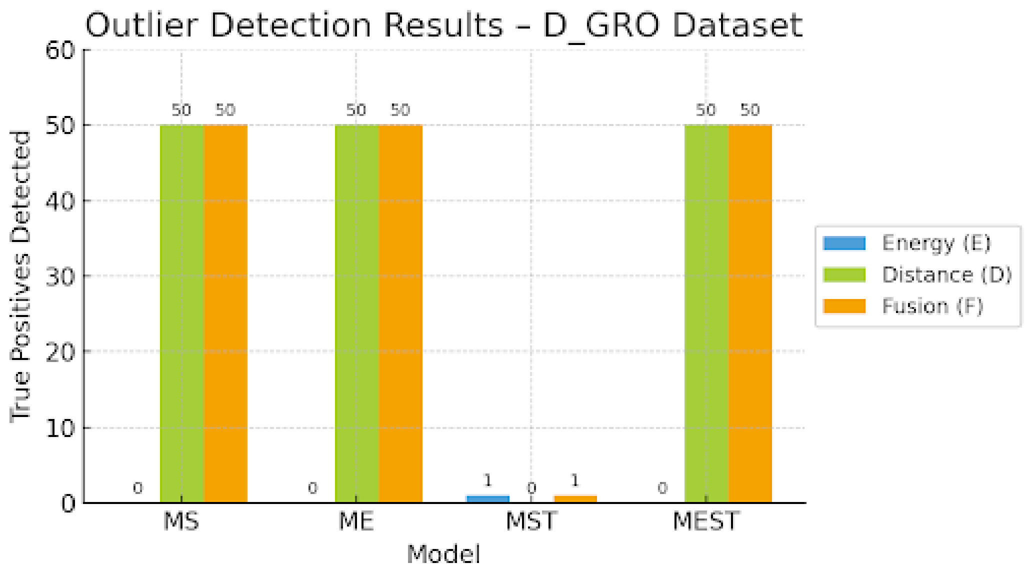


Figure 4.1: Random noise outlier detection results for dataset D_{GRO}.

Table 4.1 presents the detection performance using random noise outliers for all models applied to the D_{GRO} dataset at the 90% threshold. *TP* in the table above (and in future tables) refers to how many true positive outliers were detected by our approach. The results show that while most models effectively detected the full set of injected outliers using the distance-based method (50 true positives), the energy-based approach largely failed to flag these anomalies. Specifically, the energy-based method consistently yielded zero true positives across nearly all models. This suggests a disconnect between the energy scores and the semantic structure of outliers in this dataset. Notably, the M_{ST} model is the only one to identify a small number of true positives via distance in both outlier categories, albeit with poor overall detection. The fusion of distance and energy scores allowed for broader coverage but did not make any improvement over distance alone, except for M_{ST}. Furthermore, the intersection of both methods did not produce any overlapping true positives, showing that energy and distance-based scores captured different subsets of the data. These findings highlight the greater utility of the distance-based approach for structured, low-variance datasets like D_{GRO}.

Evaluation of Dataset D_{NOR}

Table 4.4: Generation quality metrics for D_{NOR}.

Model	Train MSE	Test MSE	Feature f2	Feature f4	Feature f6
M_S	0.0451	0.0449	0.2607	0.1132	0.2835
M_E	0.0449	0.0453	0.2651	0.1602	0.3144
M_{ST}	—	—	0.2543	0.2148	0.4916
M_{EST}	—	—	0.3368	0.3019	0.4851

The generation quality results for D_{NOR} is summarized in Table 4.4. Across all models, reconstruction MSE remained low and consistent, with M_S and M_E both achieving test MSE values around 0.045. These values indicate strong overall reconstruction performance, particularly for the single and ensemble RBM models. To further assess how well the models preserve underlying feature relationships, we measure how much each engineered feature deviates between the training data and generated data, with larger values indicating farther deviation. For M_S, deviations are minimal, especially for the simpler relationship $f_4 = 1 - f_3$, which recorded a deviation of 0.1132. While the ensemble model M_E displayed slightly higher deviation across all dependencies, its performance remained within acceptable bounds.

In contrast, the stacked models M_{ST} and M_{EST} demonstrated a trade-off. For M_{EST} in particular, deviations in $f_2 = 0.5 \cdot f_1$ and $f_6 = \text{avg}(f_1, f_3)$ exceeded 0.33 and 0.48 respectively, suggesting that the deeper stacking introduced noise or inconsistencies in preserving the relational constraints. Nevertheless, the results still show that all models are able to generate data that approximately conformed to the designed structure, with M_S yielding the most faithful reproduction.

Table 4.5: True positives and total detections across multiple thresholds for the constraint violation method using the D_{NOR} dataset.

Percentile	**TP**	**Detected**
85	8	8
90	5	5
92	4	4
95	3	3
97	2	2
99	1	1

While constraint violation detection is effective in identifying synthetically injected outliers, as shown in Table 4.5, it operates independently of the RBM architecture and its generative process. As such, it does not provide insight into model quality or reconstruction performance. Instead, this method serves primarily as a validation tool to confirm that the injected outliers exhibit detectable structural inconsistencies. For evaluating the effectiveness of RBM-based detection approaches, we therefore place greater emphasis on energy-based and distance-based methods, which more directly reflect the model's learned representation.

For the normally distributed dataset D_{NOR}, outlier detection is tested using outliers that knowingly violate the defined relationships discussed in Section 4.2.1. As shown in Table 4.6, all models exhibited relatively high detection counts across both energy- and

Table 4.6: True positives and total detections at 90% threshold for all models using the D_{NOR} dataset.

Model	**Energy**		**Distance**		**Fusion**		**Intersect TP**
	TP	Detected	TP	Detected	TP	Detected	
M_S	0	805	21	805	21	1605	0
M_E	0	805	12	805	12	1468	0
M_{ST}	0	805	6	805	6	1541	0
M_{EST}	2	805	0	0	2	805	0

distance-based thresholds, yet the number of true positives (TP) remained low overall. M_S achieved the highest number of true positives (21) under the distance-based metric, suggesting it is the most sensitive to constraint violations embedded within the outliers. Models M_E and M_{EST} had limited success in identifying true anomalies, with M_E detecting 12 and M_{EST} detecting only 2 true positives. M_{ST} also showed poor performance overall, with very few or no constraint violations correctly identified under energy or distance criteria.

These results suggest that while RBMs can effectively flag unusual samples in structured normal data, only specific variations can reliably identify true constraint-breaking samples. The limited success of the more complex ensemble and stacked models may be attributed to the subtlety of the injected outliers, which require precise internal representations that may be diluted or distorted when model complexity increases.

Evaluation of Dataset D_{SPE}

Figure 4.2 presents the generation quality metrics for D_{SPE}. As expected for this small and highly structured dataset, reconstruction mean squared error (MSE) values are relatively high across all models, with M_S and M_E showing comparable performance (Train MSE $\approx$ 0.2008, Test MSE $\approx$ 0.2107). These elevated MSE scores reflect the inherent challenge of modelling this dataset's limited and highly discrete patterns. All models produced consistent results, with M_S and M_E achieving distances of 0.1553 and 0.1550, respectively. The M_{ST} and M_{EST} models demonstrated similar or slightly elevated Hamming distances,

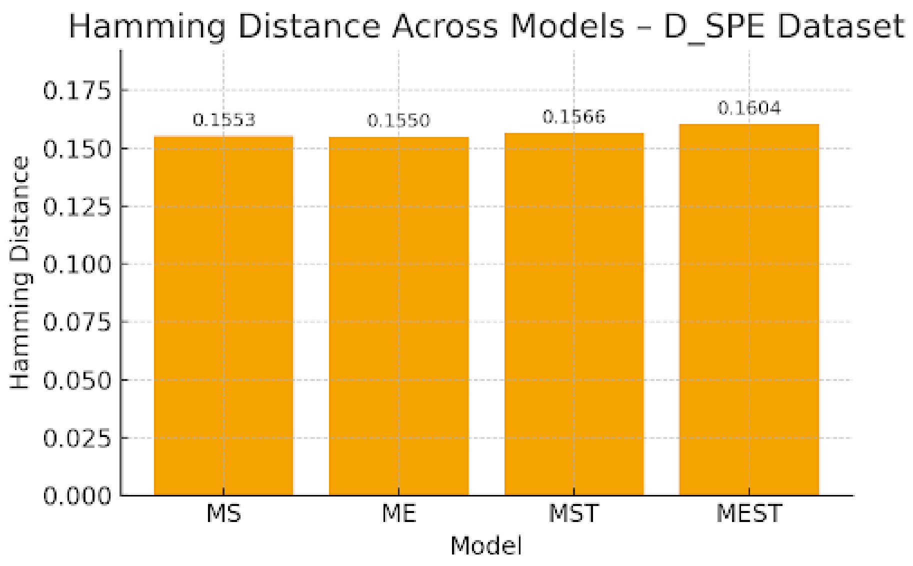


Figure 4.2: Hamming Distance values for dataset D_{SPE}.

averaging 0.1566 and 0.1604. This suggests that while stacking may introduce minor variability in the generated outputs, it does not substantially degrade the structural consistency of the generated data in this case.

Table 4.7: True positives and total detections at 90% threshold for all models using the D_{SPE} dataset.

Outlier Type	**Model**	**Energy**		**Distance**		**Fusion**		**Intersect TP**
		TP	Detected	TP	Detected	TP	Detected	
Random Noise	M_S	5	11	4	5	9	16	0
	M_E	2	11	6	10	8	21	0
	M_{ST}	6	11	2	5	8	16	0
	M_{EST}	6	11	1	2	7	13	0
Anti-Feature	M_S	1	11	7	8	8	18	0
	M_E	0	11	5	5	5	16	0
	M_{ST}	0	11	10	10	10	21	0
	M_{EST}	1	11	3	3	4	14	0

Table 4.7 shows the detection performance across all models for the D_{SPE} dataset at the 90% threshold. Unlike D_{GRO}, the relatively small size and low dimensionality of D_{SPE} led to more constrained performance, with fewer total detections and a narrower range of

true positives. Both energy-based and distance-based methods identified some outliers, but neither consistently outperformed the other across models and outlier types. The distance-based method showed modest success, with M_S and M_{ST} detecting up to 7 and 10 true positives respectively for anti-feature outliers. Energy-based detection remained limited, with most models capturing only a few true positives, and some registering none. The fusion of detection methods provided broader coverage but did not significantly increase the number of true positives, while the intersection failed to capture any shared outliers. These results suggest that for small and discrete datasets like D_{SPE}, both detection signals may struggle to isolate outliers reliably, though stacking models may provide some benefit for specific outlier types.

4.3 Outlier Detection using Categorical Data

This section presents the experiments conducted using categorical data. The goal is to evaluate how well the RBM performs in identifying outliers when trained on categorical data using two different encoding techniques, one-hot and binary encoding. The results are analyzed using both the energy-based and distance-based outlier detection approaches described in Chapter 3, as well as evaluations of the quality of the generated data. Each dataset and its associated results are discussed in the subsections that follow.

4.3.1 Datasets

Three datasets are used for evaluating the performance of the RBM-based outlier detection approach on categorical data. These datasets included Car Evaluation (D_{CAR})[14], Mushroom (D_{MUS})[57], and Nursery (D_{NUR})[52]. Each dataset contains only categorical attributes and is commonly used in classification and pattern recognition tasks, making them ideal for experiments involving categorical encoding schemes. A brief description of each dataset is provided below, followed by a summary in Table 4.8.

The D_{CAR} [14] dataset consists of 1,728 sample and 6 categorical attributes, originally derived from a hierarchical decision model. All attributes represent discrete features of a car's condition and user preferences, these include buying price, maintenance, doors, persons, lug_boot, safety, and class. The dataset contains no missing values and is evenly distributed across different combinations of attribute values.

The D_{MUS} [57] dataset includes 8,124 samples with 22 categorical attributes, each describing various physical characteristics of mushrooms (e.g., cap shape, gill color, odor). It is commonly used in binary classification tasks (edible vs. poisonous). For this study, missing values in the original dataset are handled by treating the missing value as an additional option in the category.

The D_{NUR} [52] dataset contains 12,960 samples and 8 categorical attributes related to the evaluation of nursery school applications. Attributes include parents, has_nur, form, children, housing, finance, social, health, and class. The dataset is highly imbalanced in terms of class labels but balanced in terms of attribute distributions.

Table 4.8: Summary of categorical datasets used in experiments.

Dataset	**Samples**	**Categorical Attributes**
D_{CAR}	1,728	6
D_{MUS}	8,124	22
D_{NUR}	12,960	8

Categorical data is processed using two encoding techniques: one-hot encoding and binary encoding, both of which are discussed and visualized in Section 2.4. Both are applied to the same datasets to allow a fair comparison of their impact on RBM-based outlier detection. Table 4.9 shows the dimensionality changes that occur in the datasets using our encoding techniques.

4.3.2 Experiment Parameters

The hyperparameters used to train each RBM is tuned individually for each dataset and encoding scheme to optimize data generation quality and model stability. These values

Table 4.9: Input dimensions resulting from one-hot and binary encoding.

Dataset	Original Categorical Features	Dimensions After Encoding	
		One-hot	Binary
D_{CAR}	6	21	15
D_{MUS}	22	117	63
D_{NUR}	8	27	19

Table 4.10: Training hyperparameters by categorical dataset and encoding scheme.

Dataset	Encoding	Learning Rate	Hidden Units	Epochs	Gibbs Steps
D_{CAR}	One-Hot	0.01	300	500	1000
	Binary	0.02	200	400	1000
D_{NUR}	One-Hot	0.009	100	100	1000
	Binary	0.007	200	100	1000
D_{MUS}	One-Hot	0.01	150	400	1000
	Binary	0.02	250	200	1000

are determined empirically through preliminary experiments and held consistent across all model architectures applied to a given dataset-encoding pair. Table 4.10 summarizes the learning rate, number of hidden units, training epochs, and Gibbs sampling steps used in each variant.

The structure of the models used for categorical experiments are kept the same as the binary data experiments, which can be referenced in Section 4.2.2. This section also discusses the types of outliers we used, which is kept the same for categorical data.

4.3.3 Results, Analysis, and Discussion

This section evaluates the performance of several RBM based outlier detection models on categorical datasets. Each model variation is tested on the same types of injected outliers to perform an extensive comparative analysis.

Evaluation of Dataset D_{CAR}

One-Hot Encoding

Table 4.11: Generation quality metrics for D_{CAR} (one-hot encoding).

Model	**Train MSE**	**Test MSE**	**Valid / Invalid**	**Validity Rate**
M_S	0.0000	0.0000	3155 / 1845	63.1%
M_E	0.0000	0.0000	31726 / 18274	63.5%
M_{ST}	—	—	3223 / 1777	64.5%
M_{EST}	—	—	17688 / 7362	70.6%

Table 4.11 includes both raw sample counts and the corresponding validity rates for each model. MSE score are omited for M_{ST} and M_{EST} as no full-stack decoding is performed. While all models achieved extremely low reconstruction error, their effectiveness in generating structurally valid samples varies. The stacked ensemble achieves the highest validity rate at 70.6%, suggesting improved structural generalization. In contrast, the ensemble of RBMs produces the largest number of total samples but maintains a similar validity rate to the single RBM. These findings suggest that stacking may improve the generative consistency of RBMs when handling categorical structure.

Table 4.12: True positives and total detections at 90% threshold for all models using One-Hot Encoding on D_{CAR}.

Outlier Type	**Model**	**Energy**		**Distance**		**Fusion**		**Intersect TP**
		TP	Detected	TP	Detected	TP	Detected	
Random Noise	M_S	5	40	39	39	40	75	4
	M_E	4	40	37	37	39	75	2
	M_{ST}	5	40	39	39	40	75	4
	M_{EST}	12	40	35	35	38	66	9
Anti-Feature	M_S	2	40	33	33	34	72	1
	M_E	5	40	38	38	39	74	4
	M_{ST}	6	40	33	33	36	70	3
	M_{EST}	9	40	34	34	39	70	4

Table 4.12 presents the number of true positives (TP) and total detections at the 90% threshold for each RBM-based model using one-hot encoding on the D_{CAR} dataset. Results are reported across random noise and anti-feature outliers, and are evaluated using our scor-

ing strategies: energy-based, distance-based, their fusion, and intersect scores. Across both outlier types, distance-based detection consistently achieves higher TP values than energy-based detection, highlighting its effectiveness in identifying outliers. M_{EST} demonstrates the strongest overall performance, particularly in the random noise setting, achieving 12 TPs via the energy method and 9 TPs in the intersection of energy and distance. While M_{ST} and M_S show comparable performance, ensembles generally improved detection quality, as reflected by higher fusion detections. The anti-feature outliers proved more challenging across all models, with lower TP counts in the energy column. Nevertheless, M_{EST} and M_E still maintain relatively strong results. These findings suggest that ensemble and stacked architectures, when combined with distance-based scoring or fusion aggregation, enhance outlier detection capability in high-dimensional categorical data.

Binary Encoding

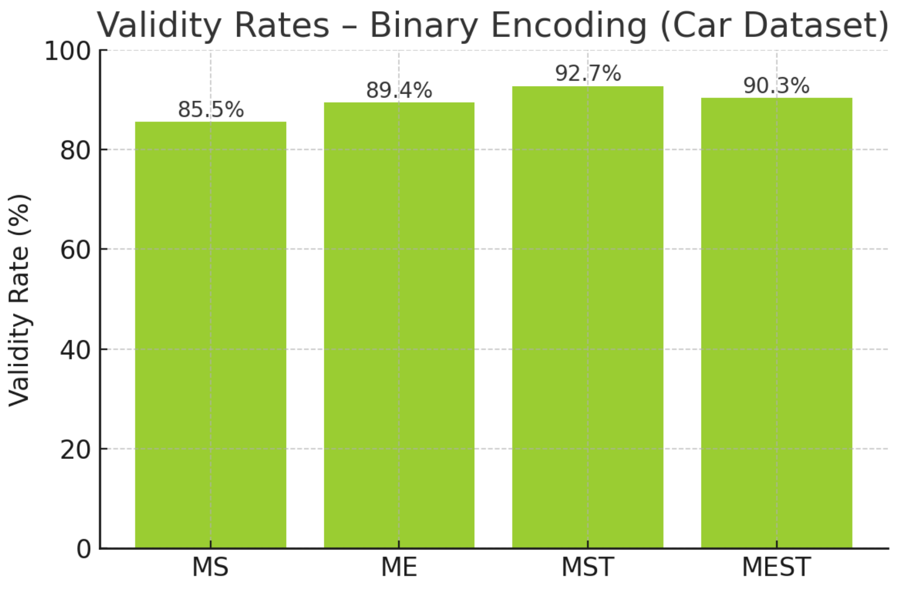


Figure 4.3: Data generation results for D_{CAR} using binary encoding.

Table 4.3 presents the data generation quality across all RBM-based models trained on D_{CAR} using binary encoding. M_E generated significantly more samples, achieving a validity rate of 89.4%. Notably, both M_{ST} and M_{EST} exhibited the strongest structural fidelity, with M_{ST} achieving the highest validity rate overall at 92.7%, and M_{EST} close behind at 90.3%. These results suggest that stacked architectures are more consistent at preserving the binary-encoded structure of categorical data during generation, even when reconstruction performance is not explicitly measured.

Table 4.13: True positives and total detections at 90% threshold for all models using Binary Encoding on D_{CAR}.

Outlier Type	**Model**	**Energy**		**Distance**		**Fusion**		**Intersect TP**
		TP	Detected	TP	Detected	TP	Detected	
Random Noise	M_S	4	40	18	18	22	58	0
	M_E	1	40	36	36	37	76	0
	M_{ST}	9	40	19	26	27	65	1
	M_{EST}	5	40	22	22	25	60	2
Anti-Feature	M_S	2	40	28	28	29	67	1
	M_E	2	40	35	35	36	74	1
	M_{ST}	4	40	24	24	28	64	0
	M_{EST}	5	40	34	34	36	71	3

The results in Table 4.13 evaluate the effectiveness of outlier detection across various RBM-based models when applied to the D_{CAR} dataset using binary encoding. In both outlier scenarios M_E and M_{EST} demonstrates consistently strong performance, particularly with the distance-based detection metric. For instance, M_E achieved 36 true positives for random noise and 35 for anti-feature outliers using distance scoring, outperforming all other models in both cases. While M_S and M_{ST} produced fewer true positives under energy scoring, they still contribute meaningfully to fusion-based detection, which combines the strengths of both scoring methods. The intersection scores, while low, indicate the small subset of outliers confidently identified by both energy and distance methods. Overall, distance scoring proves substantially more effective than energy scoring in this binary encoding setup, and models leveraging ensemble diversity benefit more from broader and more consistent outlier coverage.

Encoding Comparison for D_{CAR}

When comparing one-hot and binary encoding strategies for the D_{CAR} dataset, both approaches yield strong performance in different areas. From a data generation perspective, binary encoding exhibits higher structural fidelity, with validity rates exceeding 90% in M_{ST} and M_{EST}, compared to 64–71% for one-hot encoding. This suggests that RBMs find it easier to reproduce binary-encoded constraints, possibly due to reduced redundancy in the representation.

However, in terms of outlier detection performance, one-hot encoding demonstrates superior results, but this may be due to the injected outliers as discussed in Section 4.3.2. The models such as M_{EST} achieved significantly higher true positives using both energy and distance methods when applied to one-hot encoded data, detecting up to 12 true positives with energy-based scoring and 38 via fusion methods. In contrast, the same models under binary encoding detected fewer outliers across all detection strategies. This indicates that one-hot encoding, while structurally more demanding, may allow outliers to stand out more distinctly in the learned feature space, especially when using energy and distance-based scoring.

Overall, binary encoding performs better at structural consistency and generative reliability, while one-hot encoding appears to enhance anomaly separability, leading to improved detection performance.

Evaluation of Dataset D_{NUR}

One-Hot Encoding

Table 4.14: Generation quality metrics for D_{NUR} (one-hot encoding).

Model	Train MSE	Test MSE	Valid / Invalid	Validity Rate
M_S	0.0001	0.0001	3272 / 1728	65.4%
M_E	0.0001	0.0002	30796 / 19204	61.6%
M_{ST}	—	—	6191 / 3809	61.9%
M_{EST}	—	—	13658 / 17342	44.0%

Table 4.14 summarizes the generation quality for each RBM-based model trained on

D_{NUR} using one-hot encoding. M_S achieves the highest validity rate at 65.4%, with near-zero reconstruction error. M_E generates significantly more data but with a slightly lower validity rate of 61.6%. The M_{ST} model shows comparable structural fidelity to M_E, but M_{EST} exhibited the lowest validity rate overall at 44.0%, indicating inconsistent adherence to one-hot encoding constraints. These results suggest that, for the Nursery dataset, stacking does not necessarily improve generative fidelity and may introduce structural degradation when combined with ensembling.

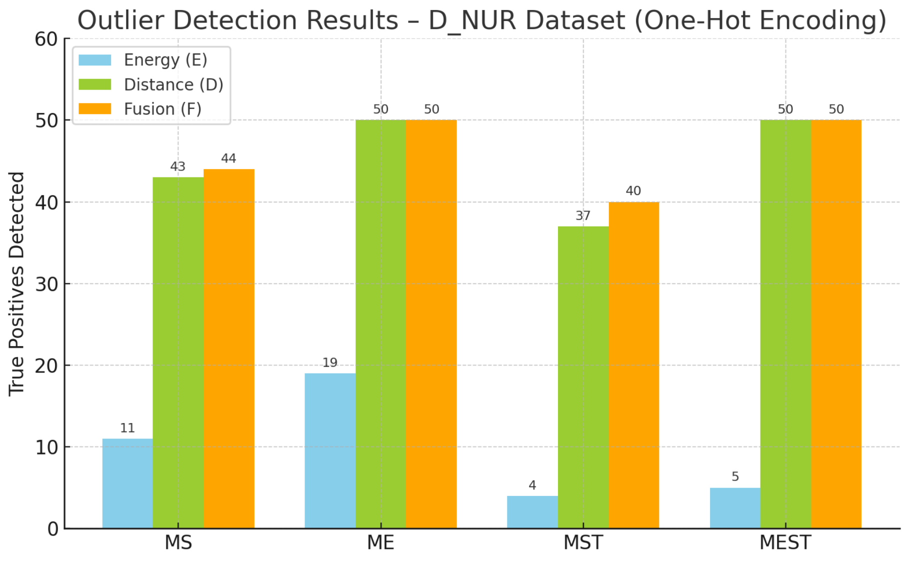


Figure 4.4: Random noise outlier detection results for dataset D_{GRO}.

Table 4.4 presents the true positives and total detections for all RBM-based models applied to the D_{NUR} dataset with one-hot encoding. Notably, M_E and M_{EST} yields the highest true positive rates across random noise outliers, especially when using the distance-based detection method. In particular, M_E achieves 50 true positives via distance scoring, suggesting it effectively captured the broader structural deviations introduced by outlier injection. While energy-based scores generally resulted in fewer detections, M_E still performs best among the models in that category. M_S and M_{ST} produces more conservative results but contributed meaningfully to the overall detection landscape, particularly when fusion-based aggregation is used. The consistently high total detections from fusion scoring indicate that combining energy and distance methods improves coverage. Additionally, the higher intersect true positives for M_E demonstrate that it is often able to confidently detect outliers across both scoring strategies.

Binary Encoding

Table 4.15: Generation quality metrics for D_{NUR} (binary encoding).

Model	**Train MSE**	**Test MSE**	**Valid / Invalid**	**Validity Rate**
M_S	0.0072	0.0072	4549 / 451	91.0%
M_E	0.0064	0.0064	45800 / 4200	91.6%
M_{ST}	—	—	9435 / 565	94.3%
M_{EST}	—	—	24818 / 5182	82.7%

Table 4.15 presents the data generation quality for D_{NUR} using binary encoding. All models achieves relatively low reconstruction error, with M_E obtaining the lowest average train and test MSE values (0.0064 each). Validity rates across models are high, with M_{ST} producing the most structurally accurate samples at 94.3%. Both M_S and M_E also maintaining strong validity, exceeding 91%. The ensemble of stacked models (M_{EST}), however, showed a reduced validity rate of 82.7%, suggesting more variability in structure across the individual stacks. These findings reinforce that binary encoding yields better data generation quality overall from the models when compared to one-hot encoding.

The binary-encoded D_{NUR} results in Table 4.16 reveal a consistent pattern of stronger performance by M_E and M_{EST} compared to M_S and M_{ST}, particularly when using the distance-based scoring method. Notably, M_E achieves the highest number of true positives (TP = 46) in the random noise scenario, and M_{ST} achieves TP = 47 under the anti-feature condition, both using distance scores. Despite high total detection counts, the number of intersecting true positives remains low across models, highlighting the limited overlap in samples deemed to be outliers by both scoring metrics. Interestingly, the energy-based method by itself yielded very low TP values across all models, suggesting that the subtler structure captured in binary encoding may not align as well with energy-based scoring. The

Table 4.16: True positives and total detections at 90% threshold for all models using Binary Encoding on D_{NUR}.

Outlier Type	**Model**	**Energy**		**Distance**		**Fusion**		**Intersect TP**
		TP	Detected	TP	Detected	TP	Detected	
Random Noise	M_S	0	653	31	433	31	1064	0
	M_E	1	653	46	90	46	736	1
	M_{ST}	3	653	13	233	16	885	0
	M_{EST}	1	653	41	384	41	997	1
Anti-Feature	M_S	2	653	38	386	38	1031	2
	M_E	1	653	48	48	49	701	0
	M_{ST}	2	653	47	496	47	1136	2
	M_{EST}	2	653	21	143	22	791	1

overall trend indicates that while the binary representation allows for successful detection under certain conditions, the choice of model and scoring method has a pronounced effect on performance, with ensemble strategies and distance-based scores offering more reliable detection capability.

Encoding Comparison for D_{NUR}

When evaluating encoding strategies for the D_{NUR} dataset, a clear distinction emerges between generation quality and detection results. Binary encoding produced significantly higher structural validity rates across most models, with M_{ST} achieving 94.3% validity and both M_S and M_E maintaining rates above 91%. In contrast, one-hot encoding resulted in lower validity, with only M_S surpassing 65%, and M_{EST} dropping to 44%. These findings confirm that RBMs more easily preserve structural constraints when trained on compact binary representations.

However, despite weaker generation fidelity, one-hot encoding led to superior outlier detection performance, but this may be due to the nature of the injected outliers as discussed in Section 4.3.2. Models such as M_E and M_{EST} consistently achieve the maximum of 50 true positives under distance-based scoring for both outlier types. In contrast, the best-performing binary models (M_E and M_{ST}) only reached 46 and 47 true positives, respectively. Moreover, one-hot encoding achieves higher intersect true positives in most cases, indicating stronger agreement between energy and distance scoring. These results suggest that while binary encoding improves generative quality, one-hot encoding better amplifies the differences between normal and outlier samples in the RBM's learned representation, enhancing detection sensitivity. Consequently, the choice between encoding schemes reflects a trade-off: structural accuracy versus discriminative clarity for outlier detection.

Evaluation of Dataset D_{MUS}

One-Hot Encoding

Table 4.17: Generation quality metrics for D_{MUS} (one-hot encoding).

Model	Train MSE	Test MSE	Valid / Invalid	Validity Rate
M_S	0.0015	0.0017	2024 / 2976	40.5%
M_E	0.0018	0.0019	15777 / 34223	31.5%
M_{ST}	—	—	1742 / 8258	17.4%
M_{EST}	—	—	6743 / 23257	22.5%

Table 4.17 shows the generation quality of each RBM-based model trained on the Mushroom dataset using one-hot encoding. M_S achieves a validity rate of 40.5%, with moderate reconstruction error. M_E generated a larger volume of data but had a lower validity rate of 31.5%, indicating greater difficulty in maintaining structural integrity. Stacked models perform poorly in this setting, with M_{ST} and M_{EST} achieving only 17.4% and 22.5% validity rates, respectively. These results suggest that D_{MUS} poses a significant challenge for generative modelling under one-hot encoding, due to the number of features that are introduced during the one-hot encoding process.

The results in Table 4.18 demonstrate a clear dominance of the distance-based scoring approach across all models for detecting outliers using one-hot encoding with D_{MUS}. While energy-based method consistently failed to identify any true positives, the distance metric is able to correctly flag all 50 injected outliers across most models and outlier types. Interest-

Table 4.18: True positives and total detections at 90% threshold for all models using One-Hot Encoding on D_{MUS}.

Outlier Type	Model	Energy		Distance		Fusion		Intersect TP
		TP	Detected	TP	Detected	TP	Detected	
Random Noise	M_S	0	249	50	72	50	320	0
	M_E	0	249	50	114	50	360	0
	M_{ST}	0	249	50	157	50	401	0
	M_{EST}	0	249	50	222	50	463	0
Anti-Feature	M_S	0	249	50	106	50	351	0
	M_E	0	249	50	88	50	334	0
	M_{ST}	0	249	50	85	50	332	0
	M_{EST}	0	25	24	24	24	49	0

ingly, the intersected true positives remained at zero for every model, indicating no overlap between the two detection method, further emphasizing the shortcomings of energy-based scoring in this context. M_{EST} still exhibits the broadest detection coverage, reaching the highest total detections under the fusion approach. Overall, these findings suggest that for D_{MUS}, model performance is more dependent on the scoring method than on the underlying RBM architecture, with distance-based detection offering a clear advantage.

Binary Encoding

Table 4.5 summarizes the generation quality of each model for the Mushroom dataset using binary encoding. M_E achieves the highest validity rate of 73.8%. M_{ST} followed closely with a 71.9% validity rate, while M_S performs noticeably worse at 64.2%. M_{EST} produces a moderate validity rate of 65.8%, indicating that while ensemble methods improved diversity, it may introduce inconsistencies into the structure. Overall, these results show that binary encoding is more robust for D_{MUS} than one-hot encoding, due to the lower number of features, and the more forgiving structure of binary encoding.

As seen in Table 4.19, the binary encoding of D_{MUS} led to an improved outlier detection performance across all models compared to the one-hot encoded variant. Most notably, the distance-based method identified close to all 50 injected random noise and anti-feature

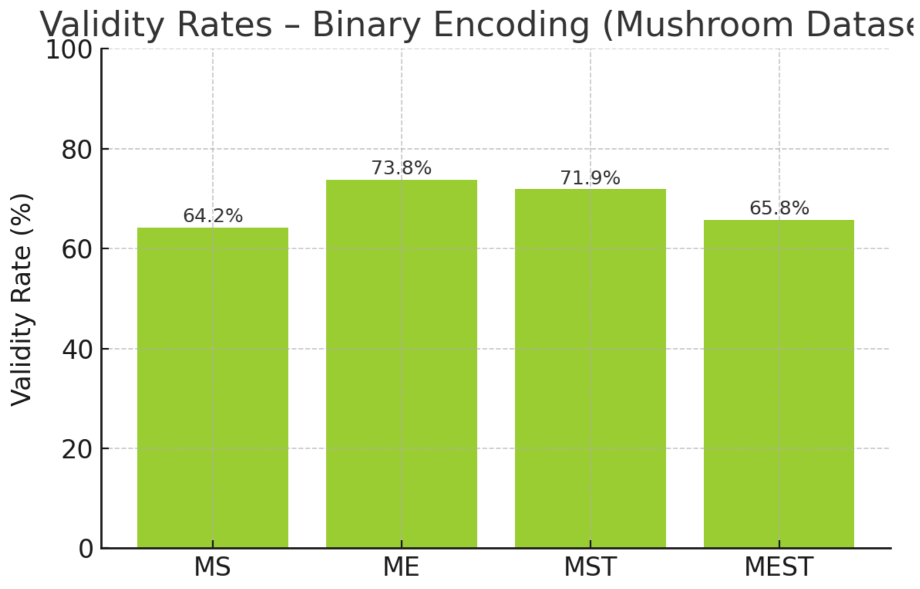


Figure 4.5: Data generation results for D_{MUS} using binary encoding.

outliers. The energy-based scores, however, remained largely ineffective, with nearly all models failing to detect any true positives using this method. Only M_{EST} showed any improvement, recording one true positive in both outlier types. This reinforces the notion that energy scores may not be a suitable choice on its own in categorical or complex binary-encoded use cases. M_{EST} offers the highest total detections under the fusion strategy and is the only model to achieve a non-zero intersected true positive count in both outlier types, suggesting some complementarity between the scoring approaches when leveraged in deeper or aggregated architectures. Overall, the binary encoding showed measurable benefits over one-hot encoding on D_{MUS}, particularly when paired with distance-based detection.

Encoding Comparison for D_{MUS}

The D_{MUS} dataset reveals significant contrasts in how encoding schemes influence generative fidelity and outlier detection effectiveness. In terms of generation quality, binary encoding clearly outperforms one-hot encoding across all models. The highest validity rate achieved under binary encoding is 73.8% by M_E, while the best one-hot encoded model, M_S, reached only 40.5%. Notably, stacked models under one-hot encoding struggles to pro-

Table 4.19: True positives and total detections at 90% threshold for all models using Binary Encoding on D_{MUS}.

Outlier Type	Model	Energy		Distance		Fusion		Intersect TP
		TP	Detected	TP	Detected	TP	Detected	
Random Noise	M_S	0	87	46	66	46	153	0
	M_E	0	87	50	85	50	171	0
	M_{ST}	0	87	49	70	49	157	0
	M_{EST}	1	87	47	64	47	150	1
Anti-Feature	M_S	0	87	50	84	50	166	0
	M_E	0	87	50	79	50	166	0
	M_{ST}	1	87	50	79	50	162	1
	M_{EST}	1	87	50	87	50	173	1

duce valid samples, with M_{ST} falling to 17.4% validity. This degradation is likely due to the substantial feature expansion introduced by one-hot encoding, which complicates learning the correct structural dependencies.

Outlier detection metrics further support the superiority of binary encoding for D_{MUS}. While both encoding strategies saw minimal contribution from energy-based scores, binary encoding enabled the distance-based method to detect nearly all 50 injected outliers across both noise and anti-feature types in every model. In contrast, one-hot encoding yields consistently poorer results: though distance-based detection identified all 50 outliers in most cases, it did so with fewer intersecting detections. Importantly, only binary encoding enabled any model to produce intersecting true positives, reinforcing the idea that binary representations not only preserve structure better but also support complementary detection mechanisms.

Collectively, these findings indicate that binary encoding is markedly more effective for both data generation and outlier detection on the D_{MUS} dataset, especially when paired with distance-based scoring and ensemble-stacked architectures.

4.4 Summary

This chapter evaluates the performance of our proposed models for detecting outliers using RBMs across multiple datasets, encoding schemes, architectural variations, and types of outliers. Through extensive experimentation on both binary and categorical data, key insights emerged. First, distance-based scoring consistently outperformed energy-based scoring across all datasets and outlier types, especially in detecting synthetically injected outliers. Energy-based methods often failed to capture meaningful deviations, producing few or no true positives despite high detection counts. Second, ensemble and stacked architectures offered measurable advantages in both data generation quality and outlier detection, particularly when combined into the M_{EST} model. Third, the choice of encoding scheme has a significant impact on model performance. One-hot encoding preserves explicit categorical structure, but introduces high dimensionality that hinders generative fidelity. On the other hand, binary encoding yields more compact representations, resulting in higher validity rates and more stable outlier detection.

Overall, these results demonstrate that the effectiveness of RBM-based outlier detection is shaped not only by the model architecture but also by encoding strategy and detection method. Taken together, the findings highlight that model architecture, encoding strategy, and scoring method each play a critical role in RBM-based outlier detection. Among the variations tested, approaches that combined binary encoding with ensemble-stacked models and distance-based scoring generally produces the most reliable and consistent results.

Chapter 5

Conclusion

5.1 Our Work

This final chapter concludes the thesis by reflecting on the limitations of the proposed methodology and outlining opportunities for future research. The work began in Chapter 1 with an introduction to neural networks, culminating in a focus on Restricted Boltzmann Machines (RBMs) and their potential application to outlier detection tasks. In Chapter 2, we provide the background information to our work, so the reader may contextualizes our approach within the field of outlier detection. We begin by introducing the Boltzmann Machine and its more practical variant, the RBM, which forms the foundation of our approach. We then review a range of contemporary outlier detection techniques to situate our work within the broader research landscape. This is followed by a discussion of data encoding strategies used to convert categorical variables into binary representations, along with ensemble learning methods designed to enhance the performance and robustness of individual models.

In Chapter 3, the implementation of various RBM configurations was described in detail, including single RBMs, ensemble of single RBMs, stacked RBMs, and ensemble of stacked RBMs. The chapter also presented the outlier detection strategies employed, which were based on free energy and reconstruction error for binary data, and extended to categorical data through encoding schemes and synthetic outlier injection.

Chapter 4 presented the experimental evaluation of these models across multiple binary and categorical datasets. Results highlighted the strengths and weaknesses of each configuration under different encoding schemes and outlier types, offering insights into model performance and data generation quality.

Collectively, this thesis contributes a systematic exploration of how RBMs and their architectural variants can be adapted for unsupervised outlier detection. It introduces a novel ensemble-based methodology that combines multiple RBM variants to enhance detection robustness, investigates the effects of different encoding strategies on categorical data, and proposes a synthetic data generation process to support evaluation in the absence of ground truth labels. These contributions aim to broaden the applicability of RBMs within the field of outlier detection and demonstrate their potential as competitive alternatives to existing methods.

The rest of this chapter critically examines the limitations encountered throughout the study and proposes directions for future work that can address these challenges and extend the contributions of this research.

5.2 Limitations of Our Approach

While this research demonstrates the viability of applying Restricted Boltzmann Machines to outlier detection tasks, several important limitations emerged during experimentation that constrain the generalizability and performance of the proposed methods.

One of the most significant challenges encountered was the inconsistent performance of energy-based outlier detection. Although free energy is theoretically well-suited for identifying inputs that stray from the learned distribution, in practice, its reliability varied across datasets and RBM configurations. In particular, energy scores often failed to distinguish subtle outliers from normal samples, especially in higher-dimensional spaces or when the synthetic data did not fully capture the variability of the training distribution. This limited the overall effectiveness of energy as a standalone metric for outlier detection, and in some cases, distance-based methods or reconstruction error provided more meaningful separations.

A second key limitation arose from the encoding schemes required to apply RBMs to categorical data. One-hot encoding, while preserving category fidelity, significantly increased the dimensionality of the input space. This expansion introduced sparsity, increased training time, and made it more difficult for the RBM to learn meaningful feature dependencies. Although binary encoding offered a more compact alternative, it sometimes distorted the semantic structure of the original categorical features, leading to reduced performance. The overall sensitivity of the models to the chosen encoding scheme highlighted the lack of a native mechanism within standard RBMs to efficiently handle discrete data, forcing compromises in either expressiveness or tractability.

Finally, although this thesis explored multi-layer RBM architectures, it did not implement a true stacked RBM as originally proposed in the deep learning literature. Instead, a simplified approximation was used where each successive RBM was trained on the generated dataset from the previous RBM. A proper stacked RBM, where each layer is trained using the hidden activations of the previous one and the full stack is then fine-tuned, could offer deeper abstraction and improved modelling capacity. However, implementing such a system would require additional mechanisms for backpropagation through layers or more complex layer-wise training, since each RBM must first be trained independently on the transformed feature space of the layer below, and subsequent fine-tuning demands propagating learning signals across multiple stochastic layers. This process is computationally intensive and less straightforward than training a single RBM, as it involves repeated sampling, aligning distributions between layers, and carefully coordinating updates so that the stacked model learns coherently as a whole.

These limitations do not diminish the overall findings but rather underscore the complexity of applying generative models to outlier detection, particularly when working with heterogeneous data types and layered model designs. Addressing these constraints offers a clear path for future improvement and further research.

5.3 Future Work

Addressing the limitations identified in this study presents several promising directions for future research that could enhance the robustness and generalizability of RBM-based outlier detection systems.

A key area for improvement involves refining the use of energy-based detection. Future work could explore calibration techniques to better align energy scores with the underlying data distribution, such as fitting thresholds based on statistical modelling of the free energy distribution or combining energy with auxiliary metrics like reconstruction error or activation sparsity. Additionally, investigating ensemble aggregation strategies that incorporate uncertainty or variance in energy across models may yield more stable detection thresholds. Another direction is to revisit the way synthetic data is generated, as better-quality samples may enhance the contrast between normal and anomalous instances when using energy as a discriminative signal.

To mitigate the challenges posed by high-dimensional input spaces resulting from encoding schemes, future research should consider adapting or extending RBM architectures to natively support categorical inputs. One possible direction is to incorporate architectural modifications that allow RBMs to more effectively model categorical data. This may involve integrating multinomial visible units, which enable the model to represent discrete variables directly without relying on high-dimensional one-hot encodings [54], or exploring the use of learned categorical embeddings, which reduce input dimensionality while preserving semantic relationships between categories [61]. Another possibility is to combine RBMs with dimensionality reduction techniques such as autoencoders or variational bottlenecks prior to training, thereby reducing the burden of sparse, one-hot-encoded representations without sacrificing feature interpretability.

The implementation of a true stacked RBM remains an important and unresolved area for extension. A full stacked model would involve training each RBM layer sequentially using the hidden activations of the previous layer and optionally fine-tuning the entire network as a generative hierarchy. Such an architecture could better capture hierarchical patterns and higher-order feature interactions, especially in more complex datasets. Implementing a true stacked RBM would also allow for deeper insights into feature abstraction progresses across layers and how this affects both generative quality and outlier detection performance. Future studies could also evaluate whether pre-training and fine-tuning schemes drawn from deep belief networks or deep Boltzmann machines improve convergence and model expressiveness.

Finally, these extensions should be evaluated on real-world outlier detection tasks, where outlier are both rare and ill-defined. Such scenarios would test the generalizability of the proposed improvements and help validate the practical utility of RBM-based outlier detection in applied domains. Taken together, it is expected that these directions may offer a clear path for enhancing both the theoretical depth and practical impact of this research.

Bibliography

[1] C Aggarwal. Outlier ensembles: Position paper. *ACM SIGKDD Explorations Newsletter*, 17(1):49–58, 2015.

[2] C Aggarwal. *Outlier analysis*. Springer, 2017.

[3] C Aggarwal. *Neural networks and deep learning: A textbook*. Springer, 2019.

[4] Y Akkem, BS Kumar, and A Varanasi. Streamlit application for advanced ensemble learning methods in crop recommendation systems–a review and implementation. *Indian J Sci Technol*, 16(48):4688–4702, 2023.

[5] W Al Farizi, I Hidayah, and M Rizal. Isolation forest based anomaly detection: A systematic literature review. In *2021 8th International Conference on Information Technology, Computer and Electrical Engineering (ICITACEE)*, pages 118–122. IEEE, 2021.

[6] V Babu and K Banana. A study on narrow artificial intelligence—an overview. *Int. J. Eng. Sci. Adv. Technol*, 24:210–219, 2024.

[7] D Bank, N Koenigstein, and R Giryes. Autoencoders. arxiv. *arXiv preprint arXiv:2003.05991*, pages 2593–2613, 2020.

[8] V Barnett and T Lewis. *Outliers in Statistical Data*. Wiley, 3rd edition, 1994.

[9] Y Bengio and O Delalleau. Justifying and generalizing contrastive divergence. *Neural Computation*, 21(6):1601–1621, 2009.

[10] Y Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[11] Y Bengio, P Lamblin, D Popovici, and H Larochelle. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19, 2006.

[12] K Berahmand, F Daneshfar, E Salehi, Y Li, and Y Xu. Autoencoders and their applications in machine learning: a survey. *Artificial Intelligence Review*, 57(2):28, 2024.

[13] C Bishop and N Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[14] M Bohanec. Car evaluation. UCI Machine Learning Repository, 1988. DOI: https://doi.org/10.24432/C5JP48.

[15] N Bostrom. *Superintelligence: Paths, dangers, strategies*. Oxford University Press, 2014.

[16] L Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[17] S Bubeck, V Chadrasekaran, R Eldan, J Gehrke, E Horvitz, E Kamar, P Lee, Y Lee, Y Li, S Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4, 2023.

[18] V Chandola, A Banerjee, and V Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(3):1–58, 2009.

[19] S Chawla and A Gionis. k-means–: A unified approach to clustering and outlier detection. In *Proceedings of the 2013 SIAM international conference on data mining*, pages 189–197. SIAM, 2013.

[20] L Chen, S Li, Q Bai, J Yang, S Jiang, and Y Miao. Review of image classification algorithms based on convolutional neural networks. *Remote Sensing*, 13(22):4712, 2021.

[21] H Cheng, M Zhang, and J Shi. A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[22] K Cios, L Kurgan, and L Goodenday. SPECT Heart. UCI Machine Learning Repository, 2001. DOI: https://doi.org/10.24432/C5P304.

[23] D Cousineau and S Chartier. Outliers detection and treatment: a review. *International journal of psychological research*, 3(1):58–67, 2010.

[24] C Dash, A Behera, S Dehuri, and A Ghosh. An outliers detection and elimination framework in classification task of data mining. *Decision Analytics Journal*, 6:100164, 2023.

[25] T Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

[26] Ivo D. Dinov. *Data Science and Predictive Analytics: Biomedical and Health Applications Using R*. The Springer Series in Applied Machine Learning. Springer, 2nd edition, 2023.

[27] S M Erfani, S Rajasegarar, S Karunasekera, and C Leckie. High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning. *Pattern Recognition*, 58:121–134, 2016.

[28] J Eschmann. Reward function design in reinforcement learning. *Reinforcement learning algorithms: Analysis and Applications*, pages 25–33, 2021.

[29] J Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[30] Y Gao and D Yeung. Deep learning for detecting abnormal events in crowded scenes. *arXiv preprint arXiv:1401.3539*, 2014.

[31] H Ghorbani. Mahalanobis distance and its application for detecting multivariate outliers. *Facta Universitatis, Series: Mathematics and Informatics*, pages 583–595, 2019.

[32] I Goodfellow, Y Bengio, and A Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[33] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

[34] C Guo and F Berkhahn. Entity embeddings of categorical variables, 2016.

[35] R Hamming. Error detecting and error correcting codes. *The Bell System Technical Journal*, 29(2):147–160, 1950.

[36] J Han, J Pei, and H Tong. *Data mining: concepts and techniques*. Morgan kaufmann, 2022.

[37] D. M. Hawkins. *Identification of Outliers*. Monographs on Applied Probability and Statistics. Chapman and Hall, London and New York, 1980.

[38] S Hicks, I Strümke, V Thambawita, M Hammou, M Riegler, P Halvorsen, and S Parasa. On evaluation metrics for medical applications of artificial intelligence. *Scientific reports*, 12(1):5979, 2022.

[39] G Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

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

[41] G Hinton, T Sejnowski, et al. Learning and relearning in boltzmann machines. *Parallel distributed processing: Explorations in the microstructure of cognition*, 1(282-317):2, 1986.

[42] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2014.

[43] N Kosaraju, S Sankepally, and K Mallikharjuna Rao. Categorical data: Need, encoding, selection of encoding method and its emergence in machine learning models—a practical review study on heart disease prediction dataset using pearson correlation. In *Proceedings of International Conference on Data Science and Applications: ICDSA 2022, Volume 1*, pages 369–382. Springer, 2023.

[44] H Larochelle and Y Bengio. Classification using discriminative restricted boltzmann machines. In *Proceedings of the 25th international conference on Machine learning*, pages 536–543, 2008.

[45] Aleksandar Lazarevic and Vipin Kumar. Feature bagging for outlier detection. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 157–166, 2005.

[46] Y LeCun, Y Bengio, and G Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015.

[47] F Liu and Za Ting, Kand Zhou. Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(1):1–39, 2012.

[48] W McCulloch and W Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133, 1943.

[49] R Neal. Annealed importance sampling. *Statistics and computing*, 11:125–139, 2001.

[50] H Pallathadka, E Ramirez-Asis, T Loli-Poma, K Kaliyaperumal, R Ventayen, and M Naved. Applications of artificial intelligence in business management, e-commerce and finance. *Materials Today: Proceedings*, 80:2610–2613, 2023.

[51] G Pang, C Shen, L Cao, and A Hengel. Deep learning for anomaly detection: A review. *ACM Computing Surveys*, 54(2):1–38, 2021.

[52] V Rajkovic. Nursery. UCI Machine Learning Repository, 1989. DOI: https://doi.org/10.24432/C5P88W.

[53] F Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[54] R Salakhutdinov and G Hinton. Deep boltzmann machines. In *Artificial Intelligence and Statistics*, pages 448–455, 2009.

[55] W Samek, G Montavon, S Lapuschkin, C Anders, and K Müller. Explaining deep neural networks and beyond: A review of methods and applications. *Proceedings of the IEEE*, 109(3):247–278, 2021.

[56] A Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, 3(3):210–229, 1959.

[57] J Schlimmer. Mushroom dataset. UCI Machine Learning Repository, 1987. Donated 27 April 1987; DOI: `https://doi.org/10.24432/C5959T`.

[58] C Seger. An investigation of categorical variable encoding techniques in machine learning: binary versus one-hot and feature hashing, 2018.

[59] A Smiti. A critical overview of outlier detection methods. *Computer Science Review*, 38:100306, 2020.

[60] T Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, pages 1064–1071, 2008.

[61] S Vora and R Subramanian. Categorical embedding techniques for machine learning models: A survey. *IEEE Access*, 9:180888–180909, 2021.

[62] J Vrábel, P Pořízka, and J Kaiser. Restricted boltzmann machine method for dimensionality reduction of large spectroscopic data. *Spectrochimica Acta Part B: Atomic Spectroscopy*, 167:105849, 2020.

[63] H Wang, M Bah, and M Hammad. Progress in outlier detection techniques: A survey. *Ieee Access*, 7:107964–108000, 2019.

[64] D Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992.

[65] S Zhai, Y Cheng, W Lu, and Z Zhang. Deep structured energy based models for anomaly detection. In *ICML*, pages 1100–1109, 2016.

[66] Z Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman and Hall/CRC, 2012.

[67] A Zimek, E Schubert, and H Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 5(5):363–387, 2012.