International Journal of Computers and Informatics

Journal Homepage: https://www.ijci.zu.edu.eg

Int. j. Comp. Info. Vol. 5 (2024) 1–12

Paper Type: Original Article

New Autoencoder-Based Method for Efficient Anomaly Detection in ECG Bio-Signals

Ahmed Daoud [1,*], Walid Khedr [1], Osama Elkomy [1] and Khalid Hosny [1]

[1] Department of Information Technology, Faculty of Computers and Informatics, Zagazig University, Zagazig 44519, Egypt.
Emails: developerdaoud@gmail.com;wkhedr@zu.edu.eg; omelkomy@fci.zu.edu.eg; k_hosny@zu.edu.eg.

Received: 05 Jul 2024 **Revised:** 30 Sep 2024 **Accepted:** 24 Oct 2024 **Published:** 26 Oct 2024

Abstract

Anomaly detection is crucial in many fields, including finance, healthcare, and cybersecurity. It can identify irregular patterns, ensure system integrity, and prevent significant losses. In this paper, we explore using an autoencoder for anomaly detection. Autoencoder is a type of neural network that is suitable for unsupervised learning. This work introduces a new autoencoder-based method and examines the architecture and training process of the autoencoder, evaluates its performance on the ECG dataset, and compares its effectiveness with baseline anomaly detection methods. The results indicate that the autoencoder significantly outperforms conventional machine learning models, achieving an accuracy of 96.96%, a high precision of 98.85%, and a balanced recall of 95.88%. Additionally, it attains an F1-score of 97.34% and a ROC-AUC of 97.17%, demonstrating superior detection ability and minimal false positives compared to PCA, MCD, and Isolation Forest models. Despite its strengths, the paper identifies critical drawbacks of autoencoders, including their sensitivity to hyperparameter selection and the need for extensive training datasets to achieve adequate performance. These challenges underscore the importance of fine-tuning and large-scale data, which can be resource-intensive. Finally, it recommends enhancing the reliability of anomaly detection systems, emphasizing the need for robust methodologies to address these limitations. It also identifies areas for future research, suggesting that further investigation could lead to more flexible and efficient anomaly detection methods.

Keywords: Anomaly Detection; Autoencoders; Unsupervised Learning; High-Dimensional Data; Machine Learning.

1 | Introduction

Anomaly detection [1], also referred to as outlier detection, plays a fundamental role in many real-world applications where identifying deviations from expected patterns can indicate potential issues such as fraud [2], equipment failures [3], or medical conditions [4]. The challenge of anomaly detection lies in the diversity of anomalies and the complexity of the data, mainly when dealing with high-dimensional datasets. Traditional methods of anomaly detection, such as Z-score analysis [5], IQR (Interquartile Range) [6], and clustering techniques like k-means [7], have been extensively used but often fall short when applied to complex, high-dimensional data [8]. These methods are too simplistic or require labeled data, which is not always available. Autoencoders [9, 10], a type of neural network, have emerged as a powerful tool for unsupervised anomaly detection. An autoencoder learns to reconstruct its input by encoding it into a lower-dimensional

Corresponding Author: ahmed.dawood@zu.edu.eg

representation and then decoding it back to the original dimensions. The reconstruction error, or the difference between the input and the reconstructed output, indicates how well the model understands the data. Data points with high reconstruction errors are flagged as anomalies. A simple diagram, Figure 1, of an autoencoder shows the input, encoder, latent space, decoder, and output layers. To learn the neuron weights and, thus, the coding, the autoencoder seeks to minimize some loss function, such as mean squared error (MSE), that penalizes X' for being dissimilar from X: $Minimize\ L = f(X, X')$.

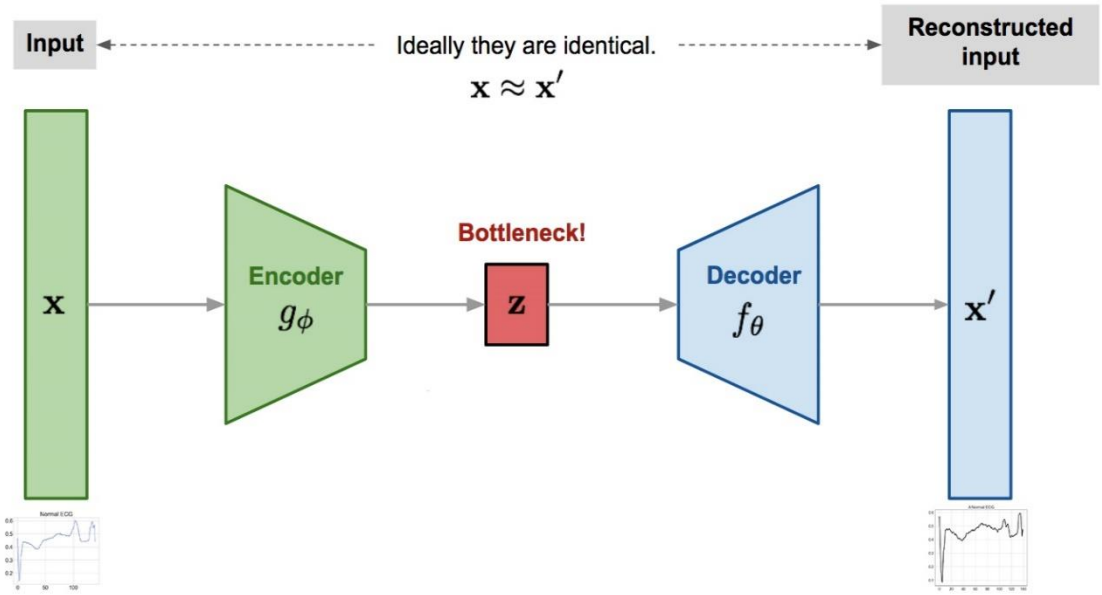

Figure 1. Autoencoder architecture.

The anomaly detection systems can be developed in two main ways [11]: either by experts who manually define thresholds for the data or automatically by using machine learning (ML) techniques that learn patterns from the data.

Manually by experts: This approach involves human experts analyzing the data and determining specific rules or thresholds. For example, if a value goes beyond a particular limit, it is considered an anomaly. This method relies on human intuition and domain knowledge.

Automatically through machine learning (ML): In this method, machine learning algorithms automatically analyze the data and identify patterns or unusual behaviors. The system learns from the data itself without the need for manually defined rules, making it more adaptable to complex and dynamic environments.

This paper aims to thoroughly investigate the use of autoencoders for anomaly detection, discussing their architecture, training, and evaluation. We also compare their performance with existing methods to assess their advantages and limitations.

The rest of this paper is organized into six sections. Section 2 presents an overview of the prior work relating to our work, showing the latest developments and methodologies. Section 3 explains the proposed approach. Section 4 discusses the experiments that were conducted and the results. Section 5 summarizes the results and discussion. Finally, section 6 provides a conclusion of the paper with future work.

2 | Related Work

The field of anomaly detection has seen significant advancements, with numerous techniques developed over the years. Traditional methods include statistical approaches like Z-score and IQR, which effectively detect outliers in univariate data. However, these methods struggle with complex, high-dimensional data where relationships between variables are not linear.

Machine learning approaches have broadened the scope of anomaly detection, introducing clustering-based methods such as k-means, classification-based methods like Support Vector Machines (SVM) [12], and ensemble methods like Isolation Forest [13]. The Isolation Forest operates by "isolating" observations through the random selection of features and split values between their maximum and minimum values. The recursive partitioning is structured as a tree, where the number of splits required to isolate a sample correlates with the path length from the root to the terminating node. The average path length across a forest of such random trees serves as a measure of normality and influences the decision function. In contrast, the Principal Component Analysis (PCA) [14] provides an effective means of detecting outliers through linear dimensionality reduction via Singular Value Decomposition (SVD). This technique decomposes the covariance matrix into orthogonal vectors known as eigenvectors associated with eigenvalues. Eigenvectors that carry higher eigenvalues encapsulate most of the data's variance, enabling the construction of a low-dimensional hyperplane that effectively represents this variance. Outliers, however, manifest distinctly from average data points, becoming more evident on the hyperplane formed by eigenvectors with lower eigenvalues. Whereas the (LOF) [15], in which the anomaly score of each sample is called the Local Outlier Factor. It measures the local deviation of the density of a given sample concerning its neighbors. It is local in that the anomaly score depends on how isolated the object is regarding the surrounding neighborhood. The KNN algorithm [16] is a simple, non-parametric classification, regression, and anomaly detection method. It classifies a query point based on the majority label of its closest neighbors or predicts a value by averaging their outputs. While kNN is easy to implement and versatile, it can be computationally expensive and sensitive to noise, especially in large datasets or high-dimensional spaces. The One-Class Support Vector Machine (OCSVM) [17] is an unsupervised algorithm primarily used for anomaly detection, where it identifies a decision boundary that encompasses most of the data points, assuming they belong to a single class (normal behavior). It works by learning a hyperplane that separates the data from the origin, classifying points outside this boundary as anomalies. OCSVM is effective in high-dimensional spaces but can be sensitive to parameter selection and may struggle with complex, non-linear data distributions.

For Gaussian-distributed datasets, outlier detection can also be achieved using the Minimum Covariance Determinant (MCD) [18], a robust estimator of covariance. While MCD is best suited for Gaussian data, it can also be applied to unimodal, symmetric distributions, but it struggles with multi-modal data where fitting may be ineffective. In such cases, projection pursuit methods are preferred. The detection process involves fitting an MCD model and then calculating the Mahalanobis distance to determine the extent to which data points are outliers.

These methods can handle higher-dimensional data and complex relationships but often require labeled datasets and may suffer from scalability issues. The following Table 1 shows a comparison between the traditional statistical-based methods, clustering-based methods, classification-based methods, and autoencoders.

Table 1. Comparison between various methods used for Anomaly detection.

Characteristic	Traditional Statistical Methods	Clustering-Based Methods	Classification-Based Methods	Autoencoders
Data Dimensionality	-Usually low to moderate; struggles with high-dimensional data.	- Can handle high-dimensional data with dimensionality reduction techniques like PCA.	- Handles moderate to high dimensional data - high dimensions can lead to overfitting.	- Effective with high-dimensional data - often used for dimensionality reduction.
Requirement for Labeled Data [19]	-Typically requires labeled data for supervised learning - unlabeled data for unsupervised learning.	- Unsupervised, does not require labeled data.	- Requires labeled data for training.	- Unsupervised, does not require labeled data.
Scalability [20]	- Limited scalability - often struggles with large datasets.	- Scales well with large datasets, though	- Varies by algorithm; some methods scale	- Generally, scales well, especially in

		performance depends on the algorithm.	well, and others scale less.	deep learning contexts.
Interpretability [21]	- High interpretability; results are often easily understandable.	- Moderate interpretability; clusters may not always be easily interpretable.	- Varies; some models, like decision trees, are interpretable, while others, like deep networks, are not.	- Low interpretability; focuses on reconstruction rather than interpretability.
Training Time [22]	- Often fast for small datasets but can be slow for large datasets.	- Typically, fast for minor to moderate datasets - can slow down with increasing complexity or dataset size.	- Can be fast or slow depending on model complexity and dataset size.	- Can be slow due to complex neural network training.
Algorithm Complexity	Usually low to moderate complexity, based on statistical formulas.	Moderate complexity depends on distance metrics and cluster initialization.	Varies from simple (e.g., logistic regression) to complex (e.g., neural networks).	High complexity due to deep learning architectures.
Handling of Non-Linear Data [10]	- Typically struggles with non-linear relationships. - often requires transformations.	- Can handle non-linear relationships depending on the algorithm (e.g., DBSCAN).	- Can handle non-linear relationships effectively, especially with neural networks.	- Handles non-linear relationships well due to non-linear activation functions.
Robustness to Outliers [23]	- Sensitive to outliers can skew results significantly.	- Varies; some methods, like K-Means, are sensitive to outliers, while others, like DBSCAN [24], are more robust.	- Varies; some algorithms, like decision trees, are robust, while others, like SVM, are sensitive.	- Moderately robust; performance can degrade with extreme outliers.
Examples of Techniques	- Linear regression, t-tests, ANOVA [25], PCA [26].	- K-Means, DBSCAN , hierarchical clustering.	- Decision trees, SVM, neural networks [27].	- Vanilla autoencoder [28]., variational autoencoder, sparse autoencoder [29]
Use Case Examples	- Hypothesis testing - Regression analysis - Descriptive statistics.	- Market segmentation - Customer profiling - Image segmentation.	- Spam detection - Medical diagnosis - Sentiment analysis.	- Anomaly detection - Data compression - Feature extraction.

The recent advancements in deep learning have brought autoencoders to the head of anomaly detection research. Unlike traditional methods, autoencoders do not require labeled data and can effectively capture complex, non-linear relationships within the data. Several studies have demonstrated the superiority of autoencoders in detecting subtle anomalies in datasets with high dimensionality. However, autoencoders are not without their challenges. Their performance is highly dependent on the architecture and hyperparameters chosen, and they require a large, representative training dataset to perform well. This study implements and evaluates an autoencoder-based anomaly detection system, comparing its performance with other popular methods and discussing its strengths and limitations.

3 | Proposed Method

3.1 | Data Preprocessing

Data preprocessing is crucial for ensuring the model's performance and robustness. The following steps were undertaken:

1. *Normalization*: All features were normalized to have zero mean and unit variance. This step ensures that the autoencoder can efficiently learn the underlying patterns in the data without being biased by the scale of different features.

2. *Train-Test Split*: The dataset was split into training and testing sets. The training set included only average data, enabling the autoencoder to learn the characteristics of non-anomalous data. The testing set included normal and anomalous data, which was used to evaluate the model's performance.

3.2 | Autoencoder Architecture

The autoencoder architecture consists of an encoder and a decoder:

1. *Encoder:* The encoder compresses the input data into a lower-dimensional representation known as the latent space. The encoder consists of several fully connected layers with decreasing sizes, which helps capture the most significant features of the data.
2. *Latent Space:* The compressed representation in the latent space is a bottleneck, forcing the model to retain only the most crucial information about the data.
3. *Decoder:* The decoder reconstructs the original data from the latent space representation. It consists of increasingly sized, connected layers that mirror the encoder's architecture.
4. *Output Layer*: The output layer reconstructs the input data with the same dimensionality as the input. The goal is for the output to match the input as closely as possible.

The model is trained using the mean squared error (MSE) loss function, quantifying the difference between the original and reconstructed inputs. Anomalies are detected by setting a threshold for reconstruction errors. Data points with a mistake above this threshold are classified as anomalies. Figure 2 shows an example of the input, output, and difference (reconstruction error) highlighted.

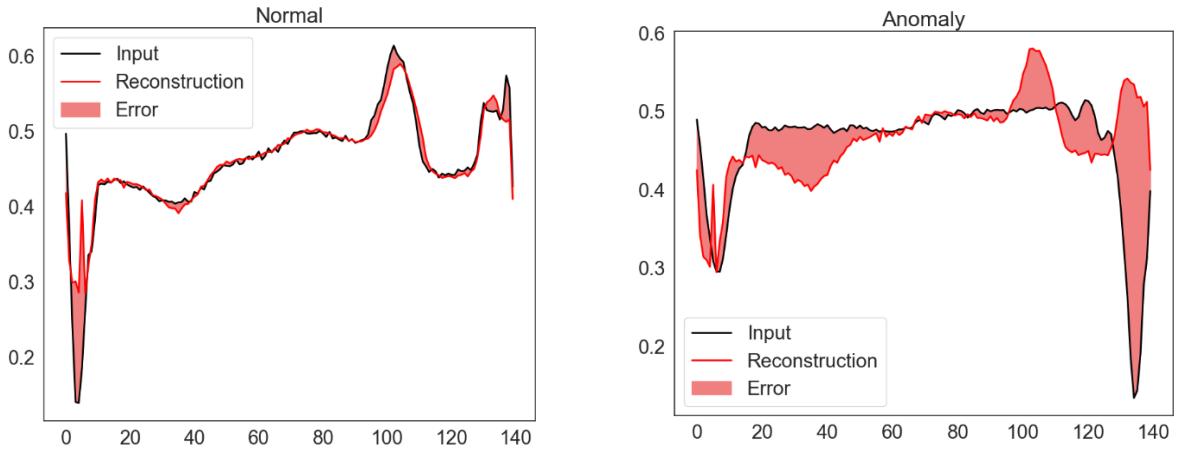

Figure 2. Normal and anomaly examples show the input and output and highlight reconstruction errors.

3.3 | Proposed Algorithm

The process shown in Figure 3 begins by splitting the dataset into X_train, which contains only average data, and X_test, which may include both regular and anomalous data. First, both datasets are normalized to have a zero mean and unit variance, ensuring the data is scaled appropriately for the neural network. An autoencoder model is then defined. This neural network consists of several layers: an input layer (with a size equal to the number of features in the dataset), an encoder made up of multiple dense layers that progressively reduce the dimensionality, a latent space layer with the smallest dimensionality, a decoder that mirrors the encoder by gradually increasing the dimensionality, and finally, an output layer that reconstructs the original input. Once the architecture is set, the autoencoder is trained using X_train. The goal is to minimize the reconstruction error, often measured by the Mean Squared Error (MSE) between the input and output of the network. After training, the model learns to reconstruct average data accurately. Next, the trained autoencoder is used to process the X_test dataset. For each data point in X_test, the model encodes and decodes the input, producing a reconstructed version of the data point. The reconstruction error is calculated for each data point by comparing the original and reconstructed data. Anomalies are detected by setting a predefined threshold

for the reconstruction error. If the reconstruction error for a data point exceeds this threshold, the point is classified as an anomaly. These anomalous data points are then collected into a list of detected anomalies.

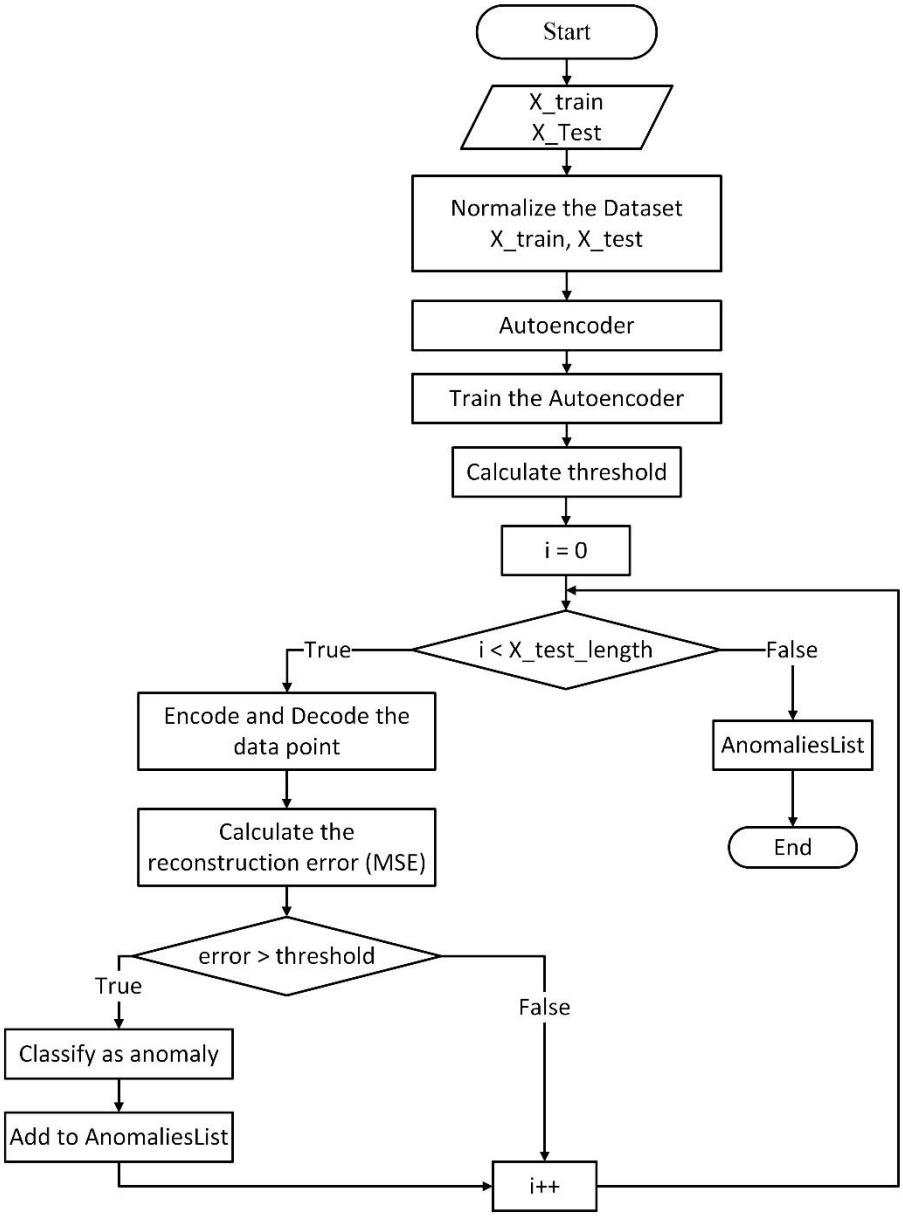

Figure 3. The proposed algorithm.

4 | Experiments and Results

4.1 | Training Process

The autoencoder was implemented using TensorFlow and trained on the normalized dataset. The training set, which only contained average data, was used to teach the autoencoder the typical patterns of non-anomalous instances. The model's architecture was carefully chosen to balance complexity with the ability to generalize well to unseen data.

The training process involved:

- *Optimization:* The Adam optimizer was used due to its adaptive learning rate, which improves convergence speed and model performance.

- *Early Stopping:* Early stopping was implemented to prevent overfitting and monitor the validation loss. The model training was halted when the validation loss stopped improving for a predefined number of epochs.

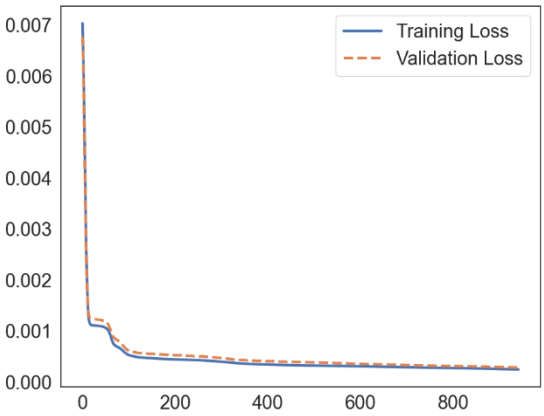

Figure 4. Training and validation loss over epochs.

Figure 4 shows a plot of the training loss and validation loss over epochs during the training process. The training loss (blue line) decreases steadily, indicating that the model is improving its performance on the training data. The validation loss (orange dashed line) also decreases, though it starts slightly higher than the training loss and converges more slowly. This suggests that the model is generalizing well to unseen data without significant overfitting, as the validation and training losses remain close throughout the training process. The plot illustrates the reduction in training (blue) and validation (orange) losses as the model trains, indicating improved performance and effective generalization.

4.2 | Evaluation Metrics

The autoencoder's performance was evaluated using several key metrics:

- Precision: The ratio of correctly identified anomalies to the number of points classified as anomalies. This metric indicates the model's accuracy in anomaly detection.
- Recall: The ratio of correctly identified anomalies to the total number of true anomalies. Recall measures the model's ability to detect all actual anomalies.
- F1-Score: The harmonic mean of precision and recall provides a metric that balances both aspects.
- ROC-AUC: The Area Under the Receiver Operating Characteristic Curve illustrates the model's ability to distinguish between normal and anomalous data points. A higher ROC-AUC score indicates better discrimination capability.

Table 2. Performance comparison of autoencoder and machine learning models.

Model/Metric	Accuracy	Precision	Recall	F1-Score	ROC-AUC
MCD	0.5825	0.8092	1.0	0.7362	0.8221
PCA	0.5825	0.4168	1.0	0.7362	0.2699
IF	0.4128	0.4618	0.2483	0.3299	0.3261
LOF	0.5825	0.6258	1	0.7362	0.5533
OCSVM	0.4295	0.5187	0.4422	0.4745	0.4108
KNN	0.5825	0.6237	1.0	0.7362	0.5562
Proposed	**0.9696**	**0.9885**	**0.9588**	**0.9734**	**0.9717**

As shown in Table 2, the proposed autoencoder performs better than machine learning models across all evaluation metrics. It achieves the highest accuracy (96.96%), significantly outperforming models like MCD, PCA, LOF, and KNN, which only reach around 58%. In terms of precision, the autoencoder scores an

impressive 98.85%, indicating far fewer false positives than other models, particularly PCA (41.68%) and Isolation Forest (46.18%). Although models like MCD, PCA, and KNN reach perfect recall (1.0), the autoencoder strikes a better balance with a near-perfect recall of 95.88%, ensuring high anomaly detection while maintaining high precision. Its F1-score (97.34%) reflects this balance, surpassing other models that average around 73%. Most notably, the autoencoder achieves a ROC-AUC of 97.17%, demonstrating exceptional ability to distinguish between normal and anomalous instances, far exceeding models like PCA (26.99%) and Isolation Forest (32.61%). Overall, the autoencoder provides a more reliable and practical approach to anomaly detection than other machine learning methods.

4.3 | Results

The autoencoder exhibited strong performance in detecting anomalies, with the following key results:

- Precision: The model achieved high precision, indicating that most detected anomalies were true positives. This suggests the model is effective at distinguishing anomalies from average data.

- Recall: The recall score was moderate, reflecting that while the model detected most anomalies, it missed some, particularly those less distinct from average data.

- F1-Score: The balanced F1-score showed that the model maintained a good balance between precision and recall, making it reliable for practical applications.

- ROC-AUC: Figure 5 The model achieved a high ROC-AUC score, demonstrating its ability to differentiate between normal and anomalous data points.

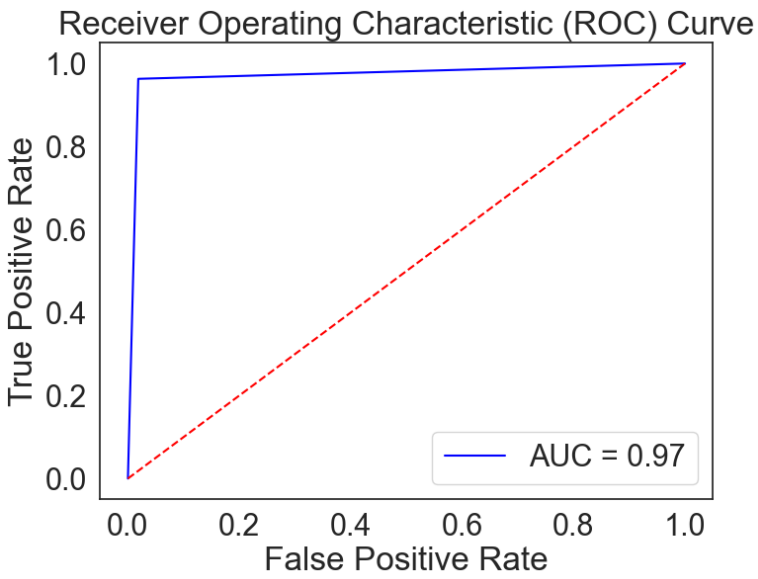

Figure 5. ROC Curve with AUC of 0.97.

These results confirm that autoencoders are highly effective for anomaly detection, especially in datasets where the relationships between features are complex and non-linear. However, the recall rate suggests a potential area for improvement in detecting more subtle anomalies.

5 | Discussion

The results of this study illustrate the significant potential of employing autoencoders for anomaly detection. Still, they also expose limitations that must be addressed for broader applicability and improved performance. This discussion explores the strengths and the challenges encountered during the research, as well as insights into the future development of anomaly detection methods using autoencoders.

5.1 | Strenghts

1. High Precision: The autoencoder demonstrated exceptionally high precision, a critical factor in anomaly detection systems, particularly in environments where false positives are costly or disruptive. For example, a false positive in medical diagnostics may lead to unnecessary interventions, and cybersecurity may waste resources addressing non-existent threats. The high precision achieved in this study suggests that the autoencoder is highly effective in distinguishing true anomalies from regular data points, making it suitable for such sensitive applications.

2. Handling High-Dimensional Data: One of the critical advantages of using autoencoders is their ability to manage high-dimensional data efficiently. Traditional anomaly detection methods, such as statistical approaches and clustering techniques, often struggle with the curse of dimensionality. Autoencoders, however, are designed to extract essential features by learning a compressed representation of the data in the latent space. This ability to capture complex patterns and relationships between variables makes autoencoders particularly useful for datasets where traditional linear methods fail to capture the intricacies of the data.

3. Unsupervised Learning: Another significant strength of autoencoders is their unsupervised learning nature. Labeled data is scarce or costly to obtain in many real-world scenarios, making supervised approaches less viable. The fact that autoencoders do not require labeled data for training makes them a highly flexible and adaptive solution for various domains, including fraud detection, equipment monitoring, and intrusion detection, where labeled anomalies are often rare. This reduces the need for extensive manual labeling efforts, allowing the model to learn from the data.

4. Flexibility in Model Design: The flexibility in designing autoencoders allows for tailoring the architecture to the specific problem. Variations such as sparse autoencoders, variational autoencoders (VAEs), and convolutional autoencoders (CAEs) can be explored depending on the data characteristics. For instance, convolutional layers are beneficial when dealing with image data or data with spatial dependencies. In contrast, VAEs offer probabilistic representations, which can be helpful in cases where uncertainty needs to be modeled explicitly.

5.2 | Limitations

1. Sensitivity to Hyperparameters: One of the critical challenges identified in this study is the autoencoder's sensitivity to hyperparameters, including network architecture, learning rate, batch size, and the threshold for anomaly detection. Choosing the right combination of these parameters is crucial for achieving optimal performance, and this often requires extensive experimentation. Furthermore, the lack of a clear guideline for setting these parameters in different anomaly detection tasks can result in suboptimal models if not fine-tuned carefully. Hyperparameter tuning methods such as grid search, random search, or more advanced techniques like Bayesian optimization may be necessary to improve model robustness.

2. Data Requirements: While autoencoders are highly effective at detecting anomalies in high-dimensional data, their performance is contingent on having a sufficiently large and representative training dataset. Suppose the training data is not diverse enough or contains too few examples of normal behavior. In that case, the autoencoder may fail to generalize well, leading to poor detection of rare or subtle anomalies. This limitation is especially problematic in domains where data is scarce or highly variable. Therefore, ensuring that the training data adequately covers the range of normal behavior is critical to the success of autoencoder-based anomaly detection systems.

3. Difficulty in Detecting Subtle Anomalies: Although the autoencoder performed well in detecting distinct anomalies, the recall rate suggests it struggled to identify more subtle deviations from standard patterns. This could be due to the nature of the autoencoder, which focuses on minimizing reconstruction errors for standard data. In cases where the difference between ordinary and anomalous data is slight, the reconstruction error may not exceed the threshold, leading to missed detections. This issue calls for more sophisticated

methods to handle such subtle cases, such as ensemble approaches or hybrid models that combine autoencoders with other anomaly detection techniques.

5.3 | Future Directions

1. Automated Hyperparameter Tuning: One of the critical areas for future work is automating the hyperparameter optimization process. Bayesian optimization [30], genetic algorithms, or other search strategies can be employed to explore a wide range of parameter combinations efficiently, reducing the reliance on manual tuning. This would improve performance and make autoencoder-based systems more accessible to non-experts who may not have the expertise to tune the models manually.

2. Exploration of Advanced Architectures: Several variants of autoencoders could be explored to enhance performance. For example, Variational Autoencoders (VAEs) [31] Introduce a probabilistic approach to anomaly detection, which could help handle uncertainty and detect more subtle anomalies. Similarly, Convolutional Autoencoders (CAEs) [32] could be applied to data with spatial relationships, such as image data or time-series data with temporal dependencies. Another promising direction is Recurrent Autoencoders (RAEs), which could be particularly useful in anomaly detection for time-series data, where temporal patterns need to be captured effectively.

3. Hybrid Models and Ensemble Methods: Combining autoencoders with other anomaly detection techniques could lead to more robust systems. For instance, ensemble approaches incorporating decision trees, isolation forests, or clustering techniques could complement the autoencoder's strengths and mitigate its weaknesses. Hybrid models could also be explored, where the autoencoder is used for dimensionality reduction, followed by other machine learning algorithms that can better handle specific types of anomalies.

4. Incorporation of Domain Knowledge: Integrating domain-specific knowledge into anomaly detection could significantly improve the model's ability to detect subtle anomalies. For example, in industrial settings, knowledge about machine operating conditions or failure modes could inform feature selection or anomaly detection thresholds. Combining autoencoder-based models with expert systems could enhance the detection system's precision and interpretability.

5. Focus on Real-Time Anomaly Detection: Another critical avenue for future research is improving the model's ability to detect real-time anomalies. Identifying anomalies is crucial for mitigating potential harm in many applications, such as cybersecurity or fraud detection. This calls for developing more efficient algorithms that can continuously process data streams and update the model in real time without the need for retraining on static datasets.

The discussion reveals both the promise and challenges of autoencoders for anomaly detection. Their ability to handle high-dimensional, complex data makes them a valuable tool, particularly when labeled data is scarce. However, challenges like hyperparameter sensitivity, data requirements, and subtle anomaly detection require further investigation. Addressing these limitations through advanced architectures, hybrid models, and automated tuning will pave the way for more robust and flexible anomaly detection systems.

6 | Conclusion and Future Work

This paper highlights the importance of using autoencoders for detecting anomalies in high-dimensional datasets, emphasizing their ability to learn complicated patterns without labeled data. The autoencoder model demonstrated great precision and competitive recall, making it an essential tool for applications requiring reliable anomaly detection. Despite its capabilities, the study revealed areas for improvement in the model, particularly in dealing with minor abnormalities and optimizing hyperparameters. Future research should concentrate on improving the robustness of the autoencoder methodology, investigating advanced architectures, and incorporating other data sources or detection methods. Finally, autoencoders appear to be a promising technique for anomaly detection, especially in complex and complicated datasets. With additional study and development, its applicability could be expanded to a broader range of fields.

Acknowledgments

The authors are grateful to the editorial and reviewers, as well as the correspondent author, who offered assistance in the form of advice, assessment, and checking during the study period.

Author Contribution

All authors contributed equally to this work.

Funding

This research has no funding source.

Data Availability

The datasets generated during and/or analyzed during the current study are not publicly available due to the privacy-preserving nature of the data but are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that there is no conflict of interest in the research.

Ethical Approval

This article does not contain any studies with human participants or animals performed by any of the authors.

References

[1] N. R. Prasad, S. Almanza-Garcia, and T. T. Lu, "Anomaly detection," Computers, Materials, & Continua, vol. 14, no. 1, pp. 1-22, 2010.
[2] T. Pourhabibi, K.-L. Ong, B. H. Kam, and Y. L. Boo, "Fraud detection: A systematic literature review of graph-based anomaly detection approaches," Decision Support Systems, vol. 133, p. 113303, 2020.
[3] Z. Li, J. Li, Y. Wang, and K. Wang, "A deep learning approach for anomaly detection based on SAE and LSTM in mechanical equipment," The International Journal of Advanced Manufacturing Technology, vol. 103, pp. 499-510, 2019.
[4] T. Fernando, H. Gammulle, S. Denman, S. Sridharan, and C. Fookes, "Deep Learning for Medical Anomaly Detection-A Survey," ACM Comput. Surv., vol. 54, no. 7, pp. 141:1-141:37, 2022.
[5] E. M. Ferragut, J. Laska, and R. A. Bridges, "A new, principled approach to anomaly detection," in 2012 11th International Conference on Machine Learning and Applications, 2012, vol. 2, pp. 210-215: IEEE.
[6] J. Yang, S. Rahardja, and P. Fränti, "Outlier detection: how to threshold outlier scores?," in Proceedings of the international conference on artificial intelligence, information processing and cloud computing, 2019, pp. 1-6.
[7] R. Kumari, M. Singh, R. Jha, and N. Singh, "Anomaly detection in network traffic using K-mean clustering," in 2016 3rd international conference on recent advances in information technology (RAIT), 2016, pp. 387-393: IEEE.
[8] S. Thudumu, P. Branch, J. Jin, and J. Singh, "A comprehensive survey of anomaly detection techniques for high dimensional big data," Journal of Big Data, vol. 7, pp. 1-30, 2020.
[9] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Conditional variational autoencoder for prediction and feature recovery applied to intrusion detection in iot," Sensors, vol. 17, no. 9, p. 1967, 2017.
[10] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in Proceedings of the MLSDA 2014 2nd workshop on machine learning for sensory data analysis, 2014, pp. 4-11.
[11] B. Steenwinckel, "Adaptive anomaly detection and root cause analysis by fusing semantics and machine learning," in The Semantic Web: ESWC 2018 Satellite Events: ESWC 2018 Satellite Events, Heraklion, Crete, Greece, June 3-7, 2018, Revised Selected Papers 15, 2018, pp. 272-282: Springer.
[12] M. M. Inuwa and R. Das, "A comparative analysis of various machine learning methods for anomaly detection in cyber attacks on IoT networks," Internet of Things, vol. 26, p. 101162, 2024.
[13] V. Yepmo, G. Smits, M.-J. Lesot, and O. Pivert, "Leveraging an Isolation Forest to Anomaly Detection and Data Clustering," Data & Knowledge Engineering, vol. 151, p. 102302, 2024.

[14] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang, "A novel anomaly detection scheme based on principal component classifier," in Proceedings of the IEEE foundations and new directions of data mining workshop, 2003, pp. 172-179: IEEE Press Piscataway, NJ, USA.

[15] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: identifying density-based local outliers," in Proceedings of the 2000 ACM SIGMOD international conference on Management of data, 2000, pp. 93-104.

[16] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in Proceedings of the 2000 ACM SIGMOD international conference on Management of data, 2000, pp. 427-438.

[17] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," Neural computation, vol. 13, no. 7, pp. 1443-1471, 2001.

[18] J. Hardin and D. M. Rocke, "Outlier detection in the multiple cluster setting using the minimum covariance determinant estimator," Computational Statistics & Data Analysis, vol. 44, no. 4, pp. 625-638, 2004.

[19] M. E. Villa-Pérez, M. A. Alvarez-Carmona, O. Loyola-González, M. A. Medina-Pérez, J. C. Velazco-Rossell, and K.-K. R. Choo, "Semi-supervised anomaly detection algorithms: A comparative summary and future research directions," Knowledge-Based Systems, vol. 218, p. 106878, 2021.

[20] N. Laptev, S. Amizadeh, and I. Flint, "Generic and scalable framework for automated time-series anomaly detection," in Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, 2015, pp. 1939-1947.

[21] D. L. Aguilar, M. A. Medina-Pérez, O. Loyola-Gonzalez, K.-K. R. Choo, and E. Bucheli-Susarrey, "Towards an interpretable autoencoder: A decision-tree-based autoencoder and its application in anomaly detection," IEEE transactions on dependable and secure computing, vol. 20, no. 2, pp. 1048-1059, 2022.

[22] K. Choi, J. Yi, C. Park, and S. Yoon, "Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines," IEEE access, vol. 9, pp. 120043-120065, 2021.

[23] D. Hendrycks, M. Mazeika, and T. Dietterich, "Deep anomaly detection with outlier exposure," arXiv preprint arXiv:1812.04606, 2018.

[24] C. Retiti Diop Emane et al., "Anomaly Detection Based on GCNs and DBSCAN in a Large-Scale Graph," Electronics, vol. 13, no. 13, p. 2625, 2024.

[25] A. P. Daga, L. Garibaldi, A. Fasana, and S. Marchesiello, "ANOVA and other statistical tools for bearing damage detection," in Proceedings of the International Conference Surveillance, Fez, Morocco, 2017, pp. 22-24.

[26] Z. Zahedeh, "Anomaly detection in system log files using machine learning algorithms/Zahedeh Zamanian," University of Malaya, 2019.

[27] B. R. Maddireddy and B. R. Maddireddy, "Neural Network Architectures in Cybersecurity: Optimizing Anomaly Detection and Prevention," International Journal of Advanced Engineering Technologies and Innovations, vol. 1, no. 2, pp. 238-266, 2024.

[28] Z. Cheng, S. Wang, P. Zhang, S. Wang, X. Liu, and E. Zhu, "Improved autoencoder for unsupervised anomaly detection," International Journal of Intelligent Systems, vol. 36, no. 12, pp. 7103-7125, 2021.

[29] J. Bi, Z. Guan, H. Yuan, J. Yang, and J. Zhang, "Network Anomaly Detection with Stacked Sparse Shrink Variational Autoencoders and Unbalanced XGBoost," IEEE Transactions on Sustainable Computing, 2024.

[30] J. Nayak, B. Naik, P. B. Dash, S. Vimal, and S. Kadry, "Hybrid Bayesian optimization hypertuned catboost approach for malicious access and anomaly detection in IoT nomalyframework," Sustainable Computing: Informatics and Systems, vol. 36, p. 100805, 2022.

[31] S. Zavrak and M. Iskefiyeli, "Anomaly-based intrusion detection from network flow features using variational autoencoder," IEEE Access, vol. 8, pp. 108346-108358, 2020.

[32] J. K. Chow, Z. Su, J. Wu, P. S. Tan, X. Mao, and Y.-H. Wang, "Anomaly detection of defects on concrete structures with the convolutional autoencoder," Advanced Engineering Informatics, vol. 45, p. 101105, 2020.