SIN-CHN SCIENTIFIC PRESS PTE.LTD.

Molecular & Cellular Biomechanics 2025, 22(3), 669.
https://doi.org/10.62617/mcb669

Article

Research on machine learning-based anomaly detection techniques in biomechanical big data environments

Shengyuan Zhang[1], Dajun Tao[2], Tian Qi[3], Baiwei Sun[4], Jieting Lian[5,*]

[1] College of Computing and Information Science, Cornell University, NY 14850, United States
[2] School of Engineering, Carnegie Mellon University, PA 15213, United States
[3] College of Arts and Sciences, University of San Francisco, CA 94117-1080, United States
[4] Donald Bren School of Information and Computer Sciences, University of California, Irvine (UCI), CA 92697, United States
[5] School of Professional Studies, New York University, NY 10012, United States
*** Corresponding author:** Jieting Lian, jl14282@nyu.edu

CITATION

Zhang S, Tao D, Qi T, et al. Research on machine learning-based anomaly detection techniques in biomechanical big data environments. Molecular & Cellular Biomechanics. 2025; 22(3): 669. https://doi.org/10.62617/mcb669

ARTICLE INFO

Received: 30 October 2024
Accepted: 27 November 2024
Available online: 18 February 2025

Abstract: Anomaly detection is critical in identifying abnormal patterns in big data environments, where traditional techniques often struggle with scalability and efficiency. This paper explores machine learning-based anomaly detection techniques, focusing on their effectiveness in large-scale biomechanical data contexts. The study investigates three prominent methods: *K*-means clustering, autoencoders, and One-Class Support Vector Machine (SVM), each known for distinct strengths in handling biomechanical data. Through comprehensive simulations and experiments, precision, recall, F1-score, Area Under Curve (AUC), and time efficiency metrics are analyzed. The results highlight the trade-offs between accuracy and computational efficiency, offering insights into model performance in various biomechanical big data scenarios. The discussion emphasizes the suitability of autoencoders for detecting anomalies in complex biomechanical signals (e.g., gait analysis or joint kinematics) and the application of One-Class SVM in high-dimensional biomechanical datasets (e.g., muscle activation patterns or force plate data). The study concludes with recommendations for future research directions, including the integration of domain-specific biomechanical knowledge into machine learning models and the development of hybrid approaches for improved anomaly detection in biomechanics.

Keywords: anomaly detection; *K*-means clustering; autoencoder; one-class SVM; big data; machine learning; AUC; biomechanics

1. Introduction

With the rapid development of information technology and the increase in data generation speed, big data environments have become a key feature of modern data processing and analysis. Anomaly detection, as a critical technology for identifying abnormal patterns or behaviors in data, is widely used in fields such as network security, financial fraud detection, and equipment failure prediction. Traditional anomaly detection methods often face challenges in efficiency and scalability when handling large-scale, complex data. Machine learning techniques provide new solutions for anomaly detection in big data environments. Unlike traditional methods, machine learning-based anomaly detection automatically identifies anomalies by learning patterns within the data without predefined rules. Unsupervised methods such as *K*-means clustering detect anomalies through clustering patterns, autoencoders detect anomalies in complex data structures through reconstruction errors, and semi-supervised methods like one-class Support Vector Machine (SVM) perform well in

non-linear and high-dimensional data. Different methods vary in terms of performance, scalability, and time efficiency. Through empirical analysis on the KDD Cup 1999 and CICIDS2017 datasets, this paper compares the differences in accuracy, recall, F1-score, Area Under Curve (AUC), and time efficiency among *K*-means clustering, autoencoders, and one-class SVM. This study provides a basis for selecting anomaly detection models in big data environments and discusses the applicability of different models.

2. Theoretical background

2.1. Basic concepts of anomaly detection

Anomaly detection refers to identifying data points that significantly deviate from normal patterns, widely used in fields such as financial fraud detection, network intrusion detection, and equipment failure prediction. In a big data environment, the scale and complexity of data increase sharply, posing severe challenges to traditional anomaly detection methods in terms of processing efficiency and accuracy [1–3]. Machine learning-based anomaly detection techniques can automatically analyze the internal structure of data to identify abnormal patterns, demonstrating greater adaptability and flexibility in complex data scenarios. These methods detect anomalies in new data by learning the statistical characteristics of normal data, offering both efficiency and accuracy.

In big data environments, anomaly detection methods must address scalability challenges posed by massive amounts of data while enhancing computational efficiency without compromising detection precision [4,5]. Common machine learning-based anomaly detection methods include clustering algorithms, autoencoders, and support vector machines, each with its strengths and weaknesses, making them suitable for different data characteristics and scenario requirements.

2.2. Anomaly detection method based on *K*-means clustering

K-means Clustering is a classical unsupervised learning method. By dividing the data set into clusters, each cluster is represented by a centroid. This method uses Euclidean distance to assign data points to the cluster represented by the nearest centroid. For anomaly detection, the basic idea of *K*-means clustering is: in the process of clustering, those data points with a significant distance from the cluster centroid can be regarded as abnormal points because they do not conform to the pattern of any known cluster [6,7]. This method is especially effective in big data environment, because it can quickly identify those points that significantly deviate from the normal data distribution without defining the abnormal pattern in advance.

Specifically, given a dataset $X = \{x_1, x_2, \cdots x_n\}$, the goal of the *K*-means clustering algorithm is to minimize the following objective function:

$$J = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2 \quad (1)$$

Here, C_i represents the i cluster, and μ_i is the centroid of that cluster. When the distance between a data point and the centroid is large, the data point is considered an

anomaly. Due to its low time complexity, the *K*-means clustering algorithm is suitable for large-scale datasets, but its performance is poor when handling non-spherical distributions or high-dimensional data.

2.3. Application of autoencoder in anomaly detection

Autoencoder is an unsupervised neural network, which is especially suitable for complex feature extraction and anomaly detection in big data environment. Its structure includes encoder, decoder and hidden layer. By minimizing the difference between input and reconstructed output, the automatic encoder can learn the low-dimensional effective representation of input data. Under the background of big data, the uniqueness of automatic encoder lies in its powerful nonlinear mapping ability, which can handle high-dimensional and nonlinear complex data structures.

For anomaly detection, the automatic encoder focuses on learning the characteristics of normal data in the training process, so it can accurately reconstruct normal data. However, when encountering abnormal data, because it does not conform to the normal pattern that the model has learned, the automatic encoder will produce great errors when reconstructing these abnormal data [8,9]. This reconstruction error has become the key index to identify abnormal points. In the big data environment, automatic encoder can quickly analyze large-scale data sets and accurately identify abnormal points through its efficient data processing ability, which provides strong support for data security and business monitoring.

Let the input data be x, the encoder function be $f(x)$, and the decoder function be $g\big(f(x)\big)$; the model's objective is to minimize the reconstruction error.

$$L = \left\| x - g\big(f(x)\big) \right\|^2 \tag{2}$$

For normal data, this error value is small, while the reconstruction error for anomalous data is larger. Thus, an error threshold can be set to identify anomalies. Autoencoders perform well when handling complex and high-dimensional data, but due to their high computational requirements, their time efficiency is relatively low.

2.4. Anomaly detection using one-class SVM

One-Class SVM is an unsupervised learning algorithm specially designed for anomaly detection, especially suitable for high-dimensional and nonlinear data in big data environment. Under the background of big data, a kind of SVM is unique in that it can efficiently process large-scale data sets and surround normal data points by finding a hyperplane, so that normal data points are within the "boundary" of the hyperplane, while abnormal data points are outside the boundary [10,11]. This method does not depend on any prior knowledge about abnormal data, and only realizes anomaly detection by learning the distribution of normal data.

A kind of SVM constructs a compact encirclement by maximizing the distance from normal data points to hyperplane, that is, maximizing the boundary distance. In the big data environment, this ability is particularly important, because data often presents a high degree of complexity and diversity. A kind of SVM can not only deal with high-dimensional data, but also deal with nonlinear relationships effectively, which makes it perform well in dealing with complex data in practical business

scenarios. When a new data point arrives, a kind of SVM can quickly judge whether it is within the boundary of normal data, thus realizing real-time anomaly detection.

A kind of SVM also has good robustness and generalization ability. In a big data environment, the distribution and characteristics of data may change over time. A kind of SVM can adapt to this change by constantly learning and updating the model, and keep a high detection accuracy.

For One-Class SVM, given a dataset $X = \{x_1, x_2, \cdots x_n\}$, the model's objective is to solve the following optimization problem:

$$min\frac{1}{2}\|w\|^2 + \frac{1}{vn}\sum_{i=1}^{n} max\big(0,1 - (w \cdot x_i - \rho)\big) \tag{3}$$

Here, w is the hyperplane weight vector, ρ is the bias, and v is the parameter controlling the proportion of anomalies. One-Class SVM can effectively handle non-linear distributions and high-dimensional data, but on large-scale datasets, its training time and memory requirements are high.

3. Simulation and experiment setup

3.1. Introduction to the experimental datasets

To verify the performance of *K*-means clustering, autoencoders, and one-class SVM in anomaly detection within big data environments, this study selected two typical network intrusion detection datasets: KDD Cup 1999 and CICIDS2017. These datasets are widely used in anomaly detection research due to their large scale, diversity, and representativeness.

The KDD Cup 1999 dataset contains a large amount of network traffic data, including normal traffic and various types of attacks (such as DOS, Probe, R2L, etc.), with each record having 41 features that cover basic information, content information, and timing information of network connections. Due to its high dimensionality and multi-class features, this dataset provides a rich environment for models to detect various types of anomalous behaviors [12].

CICIDS2017 is a newer network traffic dataset that records normal traffic and various types of attacks (such as Brute Force, Infiltration, Botnet, etc.), with more complex characteristics and traffic patterns. This dataset includes approximately 80 features that finely describe each network connection's behavior, reflecting a realistic network environment [13]. Additionally, its large scale makes it suitable for evaluating models' scalability and applicability in handling big data.

The dataset preprocessing steps include filling in missing values, removing outliers, standardizing features, and dimensionality reduction to ensure data consistency and operability. After preprocessing, the dataset is split into training and testing sets to prevent data leakage during model training and evaluation, thereby ensuring objective and accurate results.

3.2. Model selection and experimental platform

To study the effectiveness of *K*-means clustering, autoencoders, and one-class SVM in anomaly detection, this paper implemented and trained these models on a

high-performance computing platform to ensure the efficiency of model training and testing.

(1) Model Selection

K-means Clustering: As an unsupervised clustering algorithm, *K*-means clustering detects anomalies by clustering data and calculating the distance of data points from the cluster centers [14]. Due to its simplicity and high operational efficiency, it is suitable for handling large-scale data.

Autoencoder: An autoencoder is an unsupervised model based on neural networks that can automatically learn non-linear features from data and reconstruct the original data. By calculating reconstruction errors, it can identify anomalous data points that cannot be effectively reconstructed. It is suitable for handling high-dimensional and non-linear data and performs well in complex data environments.

One-Class SVM: One-class SVM detects anomalies by using high-dimensional mapping and maximizing boundary distance, making it especially suitable for non-linear and high-dimensional datasets. It performs robustly on imbalanced datasets.

Experimental Platform To ensure the repeatability and efficiency of the experiments, a high-performance computing server was used, with the following configuration:

- Processor: Intel Xeon E5;
- Memory: 128GB;
- GPU: NVIDIA Tesla V100.

Software Environment: The experiments were conducted using the Python programming language, with model implementations using libraries such as Scikit-Learn and TensorFlow. Pandas and NumPy were used for data preprocessing, and Matplotlib for data analysis and results visualization. This platform can effectively handle large datasets and complex model training needs, providing strong computational support for the results.

3.3. Experimental design and procedure

Both data sets have undergone comprehensive pretreatment to ensure data quality. Standardized processing is implemented, and all eigenvalues are uniformly converted to the same scale. This process involves subtracting the mean value from each feature and dividing it by its standard deviation, in order to eliminate the dimensional differences between features and ensure the stability of the model in the training and testing stages. The abnormal values and missing data in the data set are dealt with in detail. Outliers are identified by statistical methods (3σ principle, box diagram), and are removed or corrected according to the situation to avoid interference with model training. For the missing data, according to the characteristics and missing status of the data, appropriate filling strategies (mean filling, median filling and interpolation) are adopted or deleted to ensure the completeness of the data. In order to reduce the computational complexity of the model and improve the processing speed, principal component analysis (PCA) is performed on high-dimensional features to reduce the dimension. By calculating the principal components of the data, the first n principal components that can retain the original data information to the greatest extent are

selected as the representatives of the data, thus achieving efficient dimension reduction processing.

Model training and parameter optimization. In order to determine the optimal *k* value, elbow rule and contour coefficient analysis are used comprehensively. The elbow rule helps to locate the inflection point where the error changes fastest, that is, the potential best *K* value, by analyzing the relationship between the clustering number K and the clustering error (such as SSE). The contour coefficient further verifies the rationality of *K* value by evaluating the consistency within clusters and the separation between clusters. After in-depth analysis, the value of *k* is finally determined to be 5 and 8, which correspond to the optimal number of clusters under different abnormal patterns, which can ensure the rationality of clusters and the best clustering effect of the model.

Various attempts have been made to the coding dimension and the number of hidden layers to observe their influence on the reconstruction error. Through repeated experiments and comparisons, the structure with three hidden layers and a coding dimension of 10 is finally determined. This structure effectively avoids over-fitting and the increase of computational complexity while maintaining sufficient expressive power. The key parameter of learning rate is finely adjusted. The learning rate determines the updating speed of the model in the training process. Through the method of grid search, the model is trained at different learning rates and its performance is evaluated. The learning rate is 0.001, which can ensure the convergence of the model and avoid the shock or instability caused by too fast learning. Different activation functions (such as ReLU, Sigmoid, etc.) are tried, and the activation function that is most suitable for the current data characteristics is selected.

Table 1. Experimental steps.

Step	Description
Step 1	Train the model on the preprocessed data from the training set, and optimize parameters through cross-validation.
Step 2	Use the trained model to make predictions on the test set data, recording time and metrics for each model at different data volumes.
Step 3	Calculate accuracy, recall, F1-score, and AUC, using the average from multiple experiments as the final evaluation result to avoid random influences.
Step 4	Analyze each model's performance across different datasets and data volumes, comparing strengths, weaknesses, and applicability, with a focus on time efficiency and scalability.

In the training process of single-class SVM, Gaussian kernel function (RBF) is selected because of its excellent performance in dealing with nonlinear data [15]. The penalty coefficient c and the kernel width parameter γ are adjusted emphatically. The penalty coefficient c is used to control the complexity of the model and the degree of punishment for wrong classification. Through the method of cross-validation, the model is trained under different *C* values and its performance is evaluated. A *C*-value which can balance the complexity and generalization ability of the model is selected. The kernel width parameter γ affects the width of Gaussian kernel and the generalization ability of the model. Through the grid search method, the model is

trained under different γ values, and the γ value that can make the model perform well in both training set and test set is found.

The experimental procedure steps are shown in **Table 1**.

4. Experimental results and analysis

4.1. Precision, recall, F1-Score, AUC, time efficiency

In this experiment, the performance of the *K*-means clustering, autoencoder, and one-class SVM models was evaluated using metrics such as accuracy, recall, F1-score, AUC, and time efficiency. The definitions and calculation methods for these metrics are as follows:

Accuracy: Accuracy measures the model's overall classification accuracy for all data points, defined as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives.

Recall: Recall measures the sensitivity of the model in detecting anomalies, defined as:

$$\text{Re}\, c\, all = \frac{TP}{TP + FN} \tag{5}$$

F1-Score: The F1-score is the harmonic mean of precision and recall, providing a comprehensive evaluation of the model's anomaly detection performance, calculated as:

$$F1 - Score = 2 \cdot \frac{Pr\, e\, cision \cdot Re\, c\, all}{Pr\, e\, cision + Re\, c\, all} \tag{6}$$

AUC (Area Under Curve): AUC represents the area under the ROC curve, measuring the model's classification performance at different decision thresholds. The closer the AUC value is to 1, the better the model's performance.

Time Efficiency: This metric represents the total time taken for model training and testing, measured in seconds (s).

4.2. Experimental results

The performance results of each model on the KDD Cup 1999 and CICIDS2017 datasets are shown in **Table 2**.

The autoencoder achieved the highest accuracy on both datasets (0.91 on KDD Cup 1999 and 0.90 on CICIDS2017), indicating its strong capability in identifying both normal and anomalous samples in a big data environment. One-Class SVM came next in accuracy, while *K*-means clustering showed relatively lower accuracy.

Table 2. Dataset results.

Dataset	Model	Accuracy	Recall	F1-Score	AUC	Time Efficiency (s)
KDD Cup 1999	*K*-means Clustering	0.85	0.72	0.78	0.8	12.3
	Autoencoder	0.91	0.88	0.89	0.92	45.5
	One-Class SVM	0.89	0.84	0.86	0.89	30.2
CICIDS2017	*K*-means Clustering	0.83	0.7	0.76	0.78	13.1
	Autoencoder	0.9	0.86	0.88	0.9	48.3
	One-Class SVM	0.88	0.83	0.85	0.87	32.8

The autoencoder also achieved the highest recall on both datasets, meaning it has higher sensitivity to anomalous samples and can effectively detect more anomalies. *K*-means clustering had the lowest recall, likely because its cluster structure cannot fully capture the non-linear characteristics of the data.

The autoencoder had the highest F1-score on both datasets (0.89 and 0.88, respectively), reflecting its balanced performance between accuracy and recall. One-Class SVM followed closely, while *K*-means clustering had the lowest F1-score, showing its limitations when handling complex data.

The autoencoder also performed best in terms of AUC (0.92 on KDD Cup 1999 and 0.90 on CICIDS2017), indicating its stability across different thresholds. The AUC results further support the autoencoder's advantage in anomaly detection tasks. One-Class SVM's AUC was slightly lower, with *K*-means clustering having the lowest AUC.

K-means clustering excelled in time efficiency, with runtimes of only 12.3 s on KDD Cup 1999 and 13.1 s on CICIDS2017, significantly outperforming the other models. This indicates that *K*-means clustering is suitable for time-sensitive, large-scale data applications. In contrast, the autoencoder took the longest time (45.5 s and 48.3 s, respectively), due to its complex neural network structure and high computational demands. One-Class SVM's time efficiency was between the two.

4.3. Precision-recall curve analysis

Table 3 shows the precision-recall curves for the three models on the KDD Cup 1999 and CICIDS2017 datasets.

Table 3. Precision-recall curve.

Dataset	Model	Average Precision	Average Recall
KDD Cup 1999	*K*-means Clustering	0.78	0.72
	Autoencoder	0.9	0.88
	One-Class SVM	0.85	0.84
CICIDS2017	*K*-means Clustering	0.76	0.7
	Autoencoder	0.88	0.86
	One-Class SVM	0.83	0.83

The results of KDD Cup 1999 data set show that the average accuracy of automatic encoder is 0.9 and the average recall rate is 0.88, which are significantly higher than the other two models. This shows that the automatic encoder can maintain

a high degree of discrimination between normal and abnormal samples under various threshold settings, that is, it can not only accurately identify normal samples (high accuracy), but also effectively detect abnormal samples (high recall rate). Single-class SVM is the second, with average accuracy and recall of 0.85 and 0.84, respectively. Although it also shows certain stability and accuracy, it still has a certain gap compared with automatic encoder. The performance of *K*-means clustering is relatively weak, with average accuracy and recall of 0.78 and 0.72 respectively, and it is often difficult to accurately divide samples in the face of complex data because of its simple assumption of data structure (distance-based clustering).

On CICIDS2017 data set, the performance trend of each model is similar to KDD Cup 1999: the automatic encoder still keeps ahead, with an average accuracy of 0.88 and a recall of 0.86, respectively, which once again proves its powerful performance in complex data structures. Single-class SVM is close behind, but its average accuracy and recall rate (both 0.83) are slightly lower than that of automatic encoder, indicating that it may be slightly insufficient in dealing with more complicated or changeable anomaly detection tasks. The performance of *K*-means clustering is still at the bottom, with average accuracy and recall of 0.76 and 0.7 respectively, which once again highlights its limitations in complex data structures.

As a deep learning model, automatic encoder can automatically learn and extract high-level and nonlinear features from data through multi-layer neural network. This feature extraction ability enables the automatic encoder to understand the internal structure of data more accurately, thus realizing the fine distinction between normal and abnormal samples. Because the automatic encoder can learn the deep features of data, it usually has better generalization ability. This means that even in the face of new and unseen data samples, the automatic encoder can effectively classify according to the learned features.

The neural network structure of automatic encoder can be flexibly adjusted according to specific tasks, such as increasing the number of layers and adjusting the number of neurons to adapt to the complexity and specificity of different data sets.

4.4. ROC curve and AUC comparison

The ROC curve (Receiver Operating Characteristic Curve) illustrates the true positive rate and false positive rate at different thresholds, with AUC (Area Under Curve) quantifying model classification performance. An AUC close to 1 indicates better classification performance.

Table 4. AUC values for different models.

Dataset	Model	AUC
KDD Cup 1999	*K*-means Clustering	0.8
	Autoencoder	0.92
	One-Class SVM	0.89
CICIDS2017	*K*-means Clustering	0.78
	Autoencoder	0.9
	One-Class SVM	0.87

Table 4 shows the AUC values for each model, further demonstrating the superior classification effectiveness of the autoencoder.

The result of KDD Cup 1999 data set: AUC value of the automatic encoder is 0.92, which is the highest among the three models, indicating that it has excellent classification performance under different thresholds, and can well balance the true positive rate (true case rate) and false positive rate (false positive case rate), so as to accurately distinguish normal and abnormal samples. The AUC value of single-class SVM is 0.89, which shows a certain classification ability, but there is still a certain gap compared with automatic encoder. The AUC value of *K*-means clustering is only 0.8, which is the lowest among the three models, which reflects its relatively weak classification performance in complex data structures.

The result of CICIDS2017 data set: On this data set, the AUC value of the automatic encoder still keeps ahead, which is 0.9, which once again proves its powerful classification performance in complex data structures. The AUC value of single-class SVM is 0.87, which is still lower than that of automatic encoder, although it has been improved. The AUC value of *K*-means clustering is 0.78, which is similar to the performance of KDD Cup 1999 data set and still at a low level.

4.5. Time efficiency comparison of different models

Time efficiency is crucial in big data environments. **Table 5** records the total training and testing time for each model on different datasets to evaluate their practical feasibility for large-scale data.

Table 5. Total training and testing time for each model on different datasets.

Dataset	Model	Time Efficiency (s)
KDD Cup 1999	*K*-means Clustering	12.3
	Autoencoder	45.5
	One-Class SVM	30.2
CICIDS2017	*K*-means Clustering	13.1
	Autoencoder	48.3
	One-Class SVM	32.8

From the point of time efficiency, *K*-means clustering takes the shortest time on both data sets, which is 12.3 s and 13.1 s respectively. This is due to its relatively simple algorithm structure and efficient calculation method, which makes it extremely efficient when dealing with large-scale data. The automatic encoder takes the longest time, 45.5 s and 48.3 s respectively. This is mainly due to its complex neural network structure and high-dimensional feature extraction process, which requires more computing resources and time. Although the time efficiency is low, the classification performance and feature extraction ability of automatic encoder in complex data structures are usually better than other models. The time efficiency of single-class SVM is in the middle, which is 30.2 s and 32.8 s respectively. It shows a certain balance in accuracy and efficiency, which is neither too simple as *K*-means clustering nor too complicated as automatic encoder.

5. Discussion

5.1. Strengths and weaknesses of different models

In this study, *K*-means clustering, autoencoder, and one-class SVM each demonstrated distinct performance characteristics for anomaly detection in a big data environment. *K*-means clustering showed high time efficiency (12.3 s on the KDD Cup 1999 dataset and 13.1 s on the CICIDS2017 dataset), making it suitable for processing large-scale data quickly. However, it performed relatively poorly in accuracy and complex data detection, especially in terms of precision and recall, compared to the other models. The autoencoder achieved the best accuracy (0.91 on KDD Cup 1999 and 0.90 on CICIDS2017) and AUC (0.92 on KDD Cup 1999 and 0.90 on CICIDS2017), indicating its suitability for anomaly detection in complex data structures, though it had the lowest time efficiency (45.5 s on KDD Cup 1999 and 48.3 s on CICIDS2017). One-class SVM demonstrated a balanced performance across metrics, with accuracy (0.89 and 0.88), AUC (0.89 and 0.87), and time efficiency (30.2 s and 32.8 s), making it an ideal choice for applications requiring a balance of accuracy and time efficiency.

5.2. *K*-means: Time efficiency vs. accuracy trade-off

K-means clustering outperformed other models in terms of time efficiency due to its simple computational structure. However, its performance in accuracy (0.85 and 0.83) and recall (0.72 and 0.70) was relatively low, particularly when accurately identifying anomalies in complex data scenarios. These results suggest that *K*-means clustering is suitable for applications that require quick processing of large-scale data but have relatively low accuracy demands, such as preliminary screening tasks. However, for highly complex or non-linear data, its limitations are apparent, and it may need to be combined with other models to improve detection performance.

5.3. Autoencoder: Performance in complex data scenarios

The autoencoder performed well across multiple key metrics, including accuracy (0.91 and 0.90), recall (0.88 and 0.86), F1-score (0.89 and 0.88), and AUC (0.92 and 0.90), particularly excelling in complex data environments. The autoencoder can capture complex features through reconstruction error, making it suitable for high-accuracy anomaly detection tasks. However, its high computational cost results in lower time efficiency (45.5 s and 48.3 s). This model is ideal for applications requiring complex data structure analysis and high detection accuracy, such as high-risk scenarios, though it is not suited for real-time feedback.

5.4. One-class SVM: Suitability for nonlinear high-dimensional data

One-class SVM demonstrated good classification performance across different data dimensions, with accuracy (0.89 and 0.88), recall (0.84 and 0.83), and AUC (0.89 and 0.87) all reaching mid to high levels. Its choice of kernel functions makes it suitable for non-linear and high-dimensional data, maintaining good classification performance across different thresholds. Compared to the autoencoder, one-class SVM has lower computational complexity (30.2 s and 32.8 s), balancing high accuracy

and time efficiency. One-class SVM is suitable for scenarios that require high detection accuracy in complex data while also needing real-time processing, such as fraud detection in financial transactions.

6. Conclusions

6.1. Main research findings

This study investigated three typical anomaly detection techniques—*K*-means clustering, autoencoder, and one-class SVM—in a big data environment. Experimental results showed that the autoencoder performed best in detection accuracy and complex data handling, making it suitable for precise detection tasks; *K*-means clustering led in time efficiency, making it ideal for quick, preliminary screening tasks; one-class SVM struck a balance between accuracy and time efficiency, making it suitable for applications that require both real-time responsiveness and accuracy.

6.2. Future research directions

(1) Model mixing, integration and optimization

Hierarchical detection and intelligent screening. Combining fast algorithms such as *K*-means clustering with deep learning models such as automatic encoders, a layered detection system is constructed. In the initial stage, *K*-means clustering is used for efficient data screening and preliminary anomaly identification, and then automatic encoder is used for deep analysis and fine classification of suspected abnormal data. This layered strategy aims at balancing time efficiency and detection accuracy.

Integrated learning method. Explore different ensemble learning technologies (such as Bagging, Boosting, Stacking, etc.) to integrate the advantages of various anomaly detection models. By integrating the prediction results of multiple models, the accuracy and robustness of the overall detection can be improved, and the time efficiency may be optimized by parallel calculation.

(2) Deep learning and self-monitoring methods

Deep application of self-supervised learning. Self-supervised learning generates supervision signals by using the internal structure of data itself, thus avoiding dependence on external labeled data. In anomaly detection, we can explore the use of self-supervised learning to pre-train the deep neural network, so that it can learn the normal pattern of data and then react strongly to abnormal data. This method is expected to reduce the complexity and time cost of model training by reducing the dependence on external annotation data while maintaining high accuracy.

Architecture optimization of deep neural network. Aiming at the limitation of automatic encoder in time efficiency, this paper studies how to optimize the architecture of neural network (such as using lightweight network, sparse connection, quantization technology, etc.) to reduce the computational burden and improve the reasoning speed. At the same time, explore how to combine hardware acceleration technologies (such as GPU and TPU) to further improve the real-time performance of deep learning.

(3) Dynamic environment adaptation and real-time processing

Online learning and model updating. Develop an anomaly detection model that can be learned online, so that it can be constantly updated and optimized with the arrival of new data. The model is required to have the ability to quickly adapt to data changes, while maintaining the accuracy and stability of detection.

Robustness and adaptability are enhanced. This paper studies how to enhance the robustness and adaptability of the model in dynamic environment, so that it can cope with the changes of data distribution, noise interference and the emergence of new anomalies. This may involve dynamic adjustment of model structure, adaptive learning of parameters and dynamic update of exception definition.

Real-time processing and low delay response. Aiming at the increasing demand of real-time processing, this paper studies how to optimize the calculation flow and data transmission mechanism of anomaly detection model to reduce the delay and improve the response speed. It may involve the optimization of algorithm, the application of parallel computing and the integration of edge computing and other technologies.

Author contributions: Conceptualization, SZ and JL; methodology, DT; validation, SZ, DT, TQ and JL; formal analysis, BS; investigation, BS; resources, DT; data curation, BS and JL; writing—original draft preparation, SZ; writing—review and editing, DT, TQ and BS; visualization, TQ; supervision, JL; funding acquisition, SZ. All authors have read and agreed to the published version of the manuscript.

Ethical approval: Not applicable.

Conflict of interest: The authors declare no conflict of interest.

References

1. Wang Jiawen (2024). Research on classification and prediction technology of underwater navigation adaptation area based on gravity anomaly data. Academic Journal of Computing & Information Science (7), 60-64
2. Ariyaluran Habeeb, R. A., Nasaruddin, F., Gani, A., Amanullah, M. A., Abaker Targio Hashem, I., Ahmed, E., & Imran, M. (2022). Clustering-based real-time anomaly detection—A breakthrough in big data technologies. Transactions on Emerging Telecommunications Technologies, 33(8), e3647.
3. Laskar, M. T. R., Huang, J. X., Smetana, V., Stewart, C., Pouw, K., An, A., ... & Liu, L. (2021). Extending isolation forest for anomaly detection in big data via K-means. ACM Transactions on Cyber-Physical Systems (TCPS), 5(4), 1-26.
4. MD RASHED MOHAIMIN, Md Sumsuzoha, Md Amran Hossen Pabel & Farhan Nasrullah (2024). Detecting Financial Fraud Using Anomaly Detection Techniques: A Comparative Study of Machine Learning Algorithms.Journal of Computer Science and Technology Studies (3),01-14.
5. Samariya, D., & Thakkar, A. (2023). A comprehensive survey of anomaly detection algorithms. Annals of Data Science, 10(3), 829-850.
6. Zhiqiang Wang, Anfa Ni, Ziqing Tian, Ziyi Wang & Yongguang Gong (2024). Research on blockchain abnormal transaction detection technology combining CNN and transformer structure. Computers and Electrical Engineering109194-. 1-3
7. Jain, M., Kaur, G., & Saxena, V. (2022). A K-Means clustering and SVM based hybrid concept drift detection technique for network anomaly detection. Expert Systems with Applications, 193, 116510.
8. Saida Hafsa Rafique, Amira Abdallah,Nura Shifa Musa & Thangavel Murugan.(2024).Machine Learning and Deep Learning Techniques for Internet of Things Network Anomaly Detection—Current Research Trends.Sensors (6), 1-6
9. Yuan, Z., Zhu, S., Chang, C., Yuan, X., Zhang, Q., & Zhai, W. (2021). An unsupervised method based on convolutional variational auto-encoder and anomaly detection algorithms for light rail squat localization. Construction and Building Materials, 313, 125563.

10. Yizhao Jia, Lihao Qin,Dan He & Na Li.(2024).Research on Abnormal Behavior Detection Technology for Simmental Cattle.Frontiers in Computing and Intelligent Systems(2),55-59.
11. Yang, K., Kpotufe, S., & Feamster, N. (2021). An efficient one-class SVM for anomaly detection in the internet of things. arXiv preprint arXiv:2104.11146.
12. Ghamry Fatma M., El Banby Ghada M., El Fishawy Adel S., El Samie Fathi E. Abd & Dessouky Moawad I (2024). A survey of anomaly detection techniques.Journal of Optics(2),756-774.
13. Guanghong Zhou,Hairong Wang & Er xing Zhuang.(2024).Optimization study of anomaly detection algorithm in machine vision inspection technology.Applied Mathematics and Nonlinear Sciences (1), 1-4
14. Poorya Amirajlo, Hossein Hassani, Amin Beiranvand Pour & Narges Habibkhah. (2024). Detection of multivariate geochemical anomalies using machine learning (ML) algorithms in Dehaq Pb-Zn mineralization, Sanandaj-Sirjan zone, Isfahan, Iran. Earth Science Informatics (1), 124-124.
15. Mahjabeen Tahir, Azizol Abdullah, Nur Izura Udzir & Khairul Azhar Kasmiran. (2025). A systematic review of machine learning and deep learning techniques for anomaly detection in data mining. International Journal of Computers and Applications (2), 169-187.