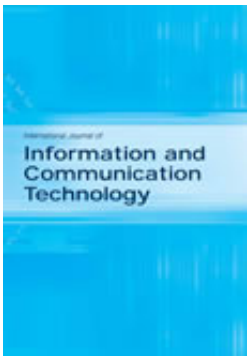
INDERSCIENCE PUBLISHERS
Linking academia, business and industry through research



International Journal of Information and Communication Technology

ISSN online: 1741-8070 - ISSN print: 1466-6642
https://www.inderscience.com/ijict

Deep learning and layered architecture-based anomaly detection method for IoT

Guifeng Zhong

DOI: 10.1504/IJICT.2025.10072223

Article History:

Received:	12 March 2025
Last revised:	21 March 2025
Accepted:	22 March 2025
Published online:	20 July 2025

Deep learning and layered architecture-based anomaly detection method for IoT

Guifeng Zhong

College of Computer Science and Engineering,
Guangzhou Institute of Science and Technology,
Guangzhou 510540, China
Email: 18520006939@163.com

Abstract: With the rapid development of the internet of things (IoT), the proliferation of IoT devices has led to massive data generation and transmission. However, security issues, especially anomaly detection, have become a major challenge. Traditional anomaly detection methods often rely on rule-based techniques or conventional machine learning models, which face issues such as low accuracy and high computational costs when handling large-scale, high-dimensional IoT data. To address these challenges, this paper presents a novel IoT anomaly detection method based on deep learning and hierarchical architecture (DeepIoT-HAD). This approach combines deep autoencoders (AE) with a hierarchical architecture to efficiently process the diversity and complexity of IoT data, improving detection accuracy while reducing computational resource consumption. Experimental results show that DeepIoT-HAD outperforms traditional methods and existing deep learning models in terms of detection accuracy and computational efficiency across multiple benchmark datasets.

Keywords: internet of things; IoT; anomaly detection; deep learning; DL; autoencoder; AE; layered architecture.

Reference to this paper should be made as follows: Zhong, G. (2025) 'Deep learning and layered architecture-based anomaly detection method for IoT', *Int. J. Information and Communication Technology*, Vol. 26, No. 27, pp.52–66.

Biographical notes: Guifeng Zhong obtained her Master's degree from Guangdong University of Technology in 2010. She is an Associate Professor at the Guangzhou Institute of Science and Technology. Her research interests include data analysis and visualisation, deep learning, and artificial intelligence big models.

1 Introduction

As internet of things (IoT) technology develops quickly, more and more devices are linked to the network producing enormous volumes of real-time data (Tien, 2017). Apart from providing daily convenience, IoT is extensively applied in numerous sectors like industrial, healthcare, and smart home (Alaa et al., 2017). But with the growing number of devices and the complexity of data, IoT systems' security and stability problems have taken front stage (Allioui and Mourdi, 2023). Particularly in data transmission, device

control and management, aberrant behaviour in these areas sometimes seriously affects system regular operation. Consequently, in present study, the effective and precise identification of aberrant behaviour in IoT systems becomes a major concern.

Regarding IoT anomaly detection, experts have put out several approaches and strategies to improve identification. Early research included conventional techniques based on statistical analysis including distance-based anomaly detection methods, clustering-based detection methods, and threshold-based detection approaches (Patcha and Park, 2007). These approaches have some success in some simple situations, but their limitations progressively show themselves due to the complexity and dynamics of IoT data, particularly when used on large-scale data and complicated contexts, where it is sometimes difficult to get the intended outcomes.

As machine learning technology develops constantly, more and more research on anomaly detection employing machine learning models is directed (Ma et al., 2021). Task involving anomaly detection makes frequent use of machine learning techniques including random forests, decision trees, and support vector machines (SVMs) (Rodriguez-Galiano et al., 2015). Usually, these approaches are more suited to handle the high-dimensional data and nonlinear interactions. Nevertheless, especially in high-dimensional data, which still makes it challenging to ensure the performance and accuracy of the models, these conventional machine learning approaches still suffer from the issues of complicated feature engineering and time-consuming model training (Fan et al., 2019).

Deep learning (DL) approaches have now emerged to be standard tools in the field of IoT anomaly detection (Ullah and Mahmoud, 2021). DL models with strong feature learning capacity have been extensively applied in anomaly detection activities including autoencoder (AE), convolutional neural network (CNN), and recurrent neural network (RNN) (Sewak et al., 2020). Nevertheless, the training and inference process of DL models typically requires a lot of computational resources and time; hence, the models are less interpretable, thus how to balance the performance and computational complexity of the models remains a major difficulty in present research.

More and more studies based on layered architectures are investigating ways to enhance IoT anomaly detection system performance even more. With each level of layered architecture executing certain functional processing, the anomaly detection choreography can be split into several tiers. For instance, hierarchical processing of the phases of data collecting, pre-processing, feature extraction and anomaly detection helps to enable focused optimisation of every module (Habeeb et al., 2019). Given the variety of sensor data and real-time needs in the IoT environment, layered architecture can significantly raise the general accuracy and efficiency of the system. With an eye toward lowering computing complexity and real-time needs while guaranteeing detection accuracy, several research have started to try to mix DL with layered architectures.

Aiming to overcome the issues of accuracy, efficiency, and computational resource consumption of anomaly detection in the IoT environment, we propose DeepIoT-HAD, an IoT anomaly detection approach combining DL and layered architecture, in this work. This work mostly makes two contributions: first, it addresses the following two aspects:

1 Combining AE with multi-level hierarchical architecture to increase the detection accuracy and computational efficiency, a DL and hierarchical architecture-based anomaly detection framework for IoT is suggested.

2 Three experiments are planned and carried out to validate the performance of the proposed DeepIoT-HAD framework compared to conventional methods and other DL models, respectively, and the experimental results reveal that the framework has major advantages in terms of detection accuracy and response time.

This work offers a quick and useful method for IoT anomaly identification with great possibility for pragmatic uses.

2 Relevant technologies

2.1 DL in IoT anomaly detection

By automatically extracting deep features from raw data, DL avoids the difficulty of hand feature engineering and detects intricate patterns impossible with conventional approaches.

Encoding and decoding the incoming data helps the AE to detect anomalies in the data (Chevrot et al., 2022). Assuming $x \in \mathbb{R}^n$ as the input, the autoencoder first transfers the data to the latent space via an encoder:

$$z = f_\theta(x) \tag{1}$$

where θ is a model parameter; z is the latent space representation; f_θ is the encoder function. The decoder then interprets the latent space representation z back into the original data space:

$$\hat{x} = g_\theta(z) \tag{2}$$

where the decoder function is g_θ and the reconstruction output of the self-encoder is $\hat{x}$. While the reconstruction error of abnormal data will be bigger, which can effectively identify normal from abnormal data, if the input data is normal the reconstruction error is usually minor. One computes the reconstruction error as:

$$L(x, \hat{x}) = \|x - \hat{x}\|_2^2 \tag{3}$$

The data is said to be abnormal when the reconstruction error above a specific level.

CNN can extract local characteristics via convolutional processes and fits data with spatial dependency (Chen et al., 2016). CNN's fundamental architecture is one of a convolutional layer, a pooling layer, and a fully linked layer. Through convolution of a convolutional kernel W with an input data x, the convolutional layer generates a feature map y. After that, the convolutional kernel W generates a feature map y via convolution with the input data x:

$$y = W * x + b \tag{4}$$

where b is the bias term; $*$ signifies the convolution operation. By means of several convolution kernels, the convolution process captures local features in the data; following down sampling by the pooling layer, the dimensionality of the features can be further lowered to capture more advanced abstract characteristics.

Apart from CNN, RNN and their variant LSTM have major benefits for processing time-series data. RNN can capture temporal aspects in the time series by means of its

cyclic structure; many IoT data are time-dependent, including sensor data, network traffic, etc., and RNN may therefore reflect them (Liu et al., 2023). With an update formula as follows, LSTM –a type of enhanced RNN –can address gradient disappearance of the conventional RNN in the training of extended sequences:

$$h_t = f\left(x_t, h_{t-1}\right) \tag{5}$$

where h_t is the hidden state at the present; x_t is the LSTM's current input; f is the activation function. Particularly crucial for time-series data in IoT, the LSTM regulates the information flow through memory gates, therefore enabling the model to properly capture long-term dependencies (Wu et al., 2021).

LSTM may learn long-term dependencies in time-series data to capture abrupt shifts or aberrant patterns in network traffic or device statuses, therefore enabling anomaly detection in IoT. LSTM can thus efficiently find and highlight changes in sensor data that differ greatly from historical data as such.

Usually based on back-propagation methods that compute the gradient of the loss function, the training of DL updates the model parameters. Cross-entropy loss is the widely used loss function in classification problems computed as:

$$L_{CE} = -\sum_{i=1}^{N}\left(y_i \log\left(\hat{y}_i\right) + \left(1-y_i\right)\log\left(1-\hat{y}_i\right)\right) \tag{6}$$

where $\hat{y}_i$ is the model's prediction; y_i is the actual label of the sample. Often adding a regularisation term, such L2 regularisation, helps one prevent overfitting:

$$L_{reg} = \lambda\sum_{j=1}^{M}\theta_j^2 \tag{7}$$

where λ is a regularisation coefficient and θ_j is a model parameter. Through optimising the loss function, DL can progressively change the parameters to reduce the prediction error (Shrestha and Mahmood, 2019).

DL has certain difficulties even if it offers great benefits in IoT anomaly detection. First of all, since anomaly data is often rare in IoT systems, the training of DL models often depends on a lot of labelled data. Second, the training process of DL models has a significant computing overhead. Furthermore influencing the efficacy of anomaly detection is the data imbalance issue perhaps biassing the model toward normal data prediction.

2.2 Layered architecture

Layerered architectures are extensively applied for data collecting, transmission, and processing in IoT systems (Mrabet et al., 2020). Particularly in cases of large-scale data, layered design serves primarily to increase the scalability, adaptability, and efficiency of the system. An IoT system usually consists of a perception layer, a network layer, and an application layer whereby each layer performs various functions and cooperatively completes tasks including data processing and anomaly detection (Erhan et al., 2021).

First, raw data gathered by sensors is mostly dependent on the perception layer (Khattak et al., 2019). These data, which could be gathered by several kinds of sensors,

comprise environmental factors including temperature, humidity, air pressure, light intensity, etc.). Specify x_{raw} as the data of the perception layer, so:

$$x_{raw} = Sensor(S) \tag{8}$$

The sensing layer aims to pass gathered raw data to the network layer (Kobo et al., 2017). Some pre-processing tasks such data cleansing, noise reduction, normalisation, etc. could be carried out throughout this procedure. The perception layer might additionally compress the data to lower data redundancy and maximise transmission (Rehman et al., 2016). Here, it is believed that following transmission, the data at the sensing layer undergoes x_{trans}, i.e.,

$$x_{trans} = Preprocess(x_{raw}) \tag{9}$$

Data then moves for routing and aggregation into the network layer. Guaranteeing the security and efficiency of data flow inside the IoT system falls to the network layer. Devices including gateways, routers, hubs, etc. might all fit the network layer. As data moves across the network layer it can be encrypted, compressed, and encapsulated. Following through this layer, the data transforms into $x_{network}$, essentially:

$$x_{network} = Compress\left(Encrypt\left(x_{trans}\right)\right) \tag{10}$$

The network layer mostly serves to guarantee stable transmission of data to the application layer (Lin et al., 2017). Some procedures, such data fusion or path optimisation of transmission, may be carried out as necessary during this process.

Usually tasked with anomaly detection and fault diagnosis depending on the provided data, the application layer is in charge of the last data processing and decision making. The application layer of IoT systems processes and analyses the data using statistical techniques, DL, etc. Following first entry into the application layer, the data will be further handled in the anomaly detection task. At this point the application layer's data processing flow is:

$$x_{processed} = Model\left(x_{network}\right) \tag{11}$$

where *Model* indicates another analysis technique or a deep learning model.

Furthermore, the data in IoT systems are often multidimensional, hence at the application level fusion processing of multi-sensor data is essential. By means of data fusion, the system may enhance anomaly detection accuracy and extract features from several angles. Assume the data gathered from several sensors are $x_1, x_2, \ldots, x_m$; these can be combined with x_{fuse} by weighted averaging or another technique.

$$x_{fuse} = \sum_{i=1}^{m} w_i x_i \tag{12}$$

where w_i is the weight of every sensor data. The fused data will be forwarded to the DL for chores including anomaly detection.

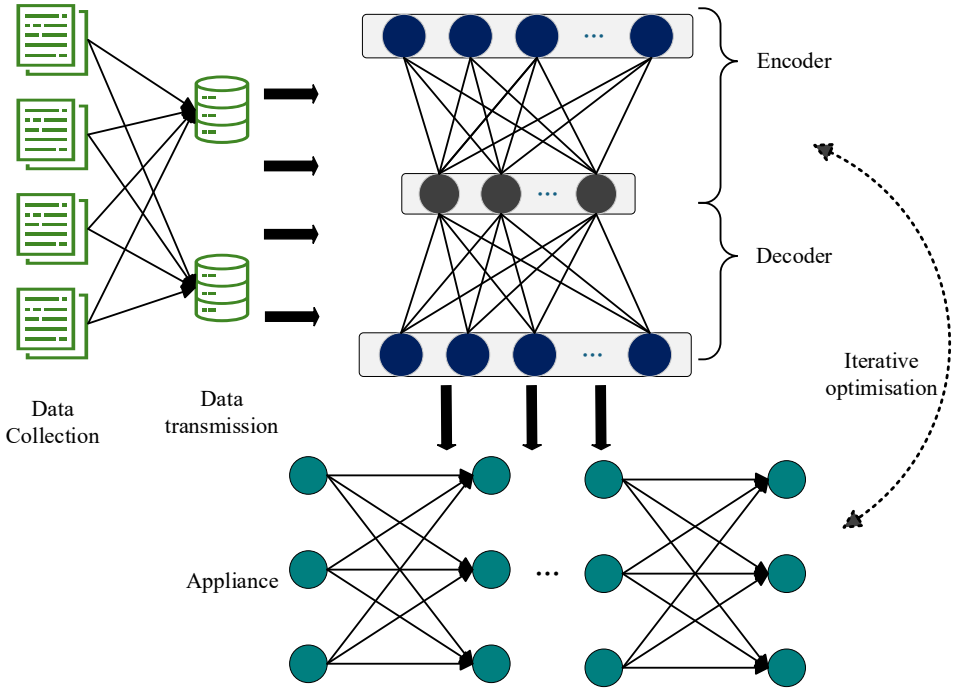
Generally, IoT depends much on layered architecture. Decomposing data processing chores into several levels helps each layer to concentrate on its particular purpose, hence improving the scalability, adaptability, and fault tolerance of the system. Data from the sensing layer to the network layer and subsequently to the application layer is transmitted

and processed such that the system can effectively gather, transmit, and analyse data, therefore laying a strong basis for next uses like anomaly detection.

3 IoT anomaly detection system framework

This IoT anomaly detection system's structure comprises essentially in five modules: data gathering module, data transmission module, deep learning detection module, application layer module, and evaluation and optimisation module. As illustrated in Figure 1, these modules cooperate to finish the whole process from data collecting to anomaly detection to final decision making and assess and maximise the performance of the system. The specifics are as follows:

Figure 1 IoT anomaly detection system (see online version for colours)



3.1 Data acquisition module

The primary process in the module on data acquisition is gathering data from IoT devices and first processing of it. Data normalisation is common to guarantee that data from several sensors are handled at the same scale. The formula for standardising is:

$$X_{norm} = \frac{X - \mu}{\sigma} \tag{13}$$

where X is the original data; μ is the mean; σ is the data's standard deviation; X_{norm} is the normalised data.

One often occurring issue in data collecting is noise. Weighted average allows the data to be smoothed therefore removing the noise. The weighted average formula is:

$$\hat{X}_t = \frac{\sum_{i=0}^{N} w_i X_{t-i}}{\sum_{i=0}^{N} w_i} \tag{14}$$

where $\hat{X}_t$ is the smoothed data; X_{t-i} is the original data at instant $t - i$; w_i is the weighting factor; N is the window size.

Linear interpolation might lastly help to fill missing values in the missing data problem. The interpolation recipe is:

$$X_{fill} = X_{prev} + \frac{\left(X_{next} - X_{prev}\right) \cdot \left(t - t_{prev}\right)}{\left(t_{next} - t_{prev}\right)} \tag{15}$$

where X_{fill} is the data following interpolation and filling; X_{prev} and X_{next} are the data before and after the missing values; t_{prev} and t_{next} are the relevant timestamps; t is the timestamp of the current missing value.

3.2 Data transmission module

First of importance in the data transmission module should be data transmission security and efficiency. Data reduction is achieved for this reason using Huffman coding's compression technique. Its computation is:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i) \tag{16}$$

where $H(X)$ is the entropy of the source data X; $p(x_i)$ is the data element x_i's probability; n is the total count of data items.

Usually, an encryption technique helps to guarantee that the data is not altered during transit. One often used symmetric encryption method is AES, whose encryption process may be expressed as:

$$C = E(K, P) \tag{17}$$

where K is the key; C is the encrypted data; E is the encryption technique; P is the original data.

Using a TCP/IP-based transmission control method helps to guarantee that data is not lost during transfer and to get effective transmission. Usually, a sliding window protocol is utilised for this aim to regulate the data flow using a formula:

$$W = \min\left(W_{\max}, \frac{R_{available}}{RTT}\right) \tag{18}$$

where W is the sliding window size; $W_{\max}$ is the maximum window size; $R_{available}$ is the accessible bandwidth; RTT is the round trip delay.

3.3 DL detection module

DL detection module is really important in IoT anomaly detection system. In this framework, AE is applied as a DL method for anomaly identification. Based on unsupervised learning, self-encoder is a network architecture able to learn the normal features of the data, compress the input data into a low-dimensional space, and detect the anomalies in the data by reconstruction error.

There are two sections to AE: encoder and decoder. Whereas the decoder rebuilds the low-dimensional representation Z back to the original space to obtain the reconstruction result $\hat{X}$, the encoder translates the input data X to a low-dimensional space representation Z. One may determine the reconstruction error with the following formula:

$$L_{AE} = \frac{1}{N}\sum_{i=1}^{N}\left\|X_i - \hat{X}_i\right\|^2 \tag{19}$$

where X_i is the i^{th} sample; $\hat{X}_i$ is its reconstruction result; N is the total number of samples; LAE is the reconstruction error, therefore indicating the variation between the input and rebuilt data.

Through optimising this reconstruction error, AE learns the usual pattern of the data. Sample with high reconstruction error can be regarded as anomalous data for anomaly identification. By means of a reconstruction error threshold θ, one can ascertain if the data is aberrant or not, therefore enabling more precisely detection of aberrant data. Should the reconstruction error surpass this threshold, the data is deemed aberrant:

$$\hat{y} = \begin{cases} 1 & \text{if } L_{AE}(X) > \theta \\ 0 & \text{if } L_{AE}(X) \le \theta \end{cases} \tag{20}$$

where $L_{AE}(X)$ is the reconstruction error of the current input sample X, where $\hat{y}$ is the prediction result; 1 denotes aberrant, 0 denotes normal.

This method allows the deep learning model to handle different unlabelled data and efficiently identify aberrant events from the data streams of IoT devices with high adaptability.

3.4 Application layer module

The core of the IoT anomaly detection system and in charge of additional decision making and processing depending on deep learning model output is the application layer module. The application layer module of the IoT environment uses a rule-based decision-making framework to properly manage aberrant occurrences. Under such a system, a rule engine uses historical data, device status and current environmental elements in addition to anomaly detection results as inputs to reach a final conclusion.

When a sensor data anomaly is found, for instance, the application layer responds in line with several anomaly kinds. A basic decision function can be built assuming y_t is the prediction outcome of DL at instant t (1 denotes abnormal, 0 implies normal).

$$y_{decision} = \mathbb{I}(y_t = 1)$$

where $\mathbb{I}(y_t = 1)$ is an indicator function and $y_t = 1$ indicates that the data is found to be anomalous and $y_{decision}$ is 1, indicating that anomalous processing is necessary; if $y_t = 0$ the system does not respond and $y_{decision}$ is 0.

To further make more precise decisions, the application layer also must combine anomaly information from other data sources. To lower some chance mistakes, the system may have to weight and average the anomalous information from several devices when it detects anomalies from many devices. The weighted decision output can be stated as y_i, the anomaly prediction result of the i^{th} device, assuming that y_i is the outcome of weight w_i:

$$y_{fusion} = \frac{\sum_{i=1}^{N} w_i y_i}{\sum_{i=1}^{N} w_i} \tag{22}$$

where N is the overall number of devices; w_i is the weight of device i; y_{fusion} is the decision output following weighted fusion.

These decision outcomes also enable the application layer module to initiate other responses, e.g., alert generation, device parameter adjustment, fault automatically repair, etc., so improving the adaptability and resilience of the system. In some complicated situations, the application layer can additionally aggregate previous data for pattern recognition and trend prediction, therefore anticipating and responding to possible anomaly risks in advance.

This framework can efficiently improve the response speed and processing capability to abnormal events by combining the anomaly detection of the deep learning model, so guaranteeing the continuous and stable running of the system.

3.5 Evaluation of the optimisation module

Monitoring and optimising DL to guarantee the accuracy and efficiency of the IoT anomaly detection system falls to the evaluation and optimisation module. The module functions mostly through the following actions:

Metrics including accuracy, precision, and recall first help to assess the model performance. Assuming $\hat{y}$ as the model output and y as the true label, the formula determines accuracy:

$$\text{Accuracy} = \frac{\sum_{i=1}^{N} \mathbb{I}(\hat{y}_i = y_i)}{N} \tag{23}$$

Meanwhile, precision and recall can be defined separately:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{24}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{25}$$

where *TP*, *FP* and *FN* stand for true positives, false positives and false negatives accordingly. Combining the F1 values allowed one to assess memory and accuracy with equation:

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{26}$$

These optimisation techniques guarantee that the evaluation optimisation module delivers regular performance reports to enable system managers monitor model health and guarantees effective anomaly identification over the long run.

4 Experimental results and analyses

4.1 Datasets

This work uses publicly accessible IoT Network Intrusion Dataset (IoT-NIDS) dataset on Kaggle to test the efficacy of the suggested IoT anomaly detecting approach. Suitable for anomaly detection activities, this dataset comprises IoT device traffic data for several cyber attack kinds. Table 1 exhibits the main characteristics of the dataset:

Table 1 IoT-NIDS information

Feature	*Description*	*Data type*
Packet size	Size of each packet	Numeric
Timestamp	Timestamp of the packet	Numeric
Source IP	Source IP address of the packet	Categorical
Destination IP	Destination IP address of the packet	Categorical
Protocol type	Network protocol used (e.g., TCP, UDP)	Categorical
Packet type	Type of packet (e.g., request, response)	Categorical
Label	Data label (normal or attack)	Categorical (0 or 1)

The collection includes network traffic from several IoT devices tagged with both normal and several kinds of attack events. Data pretreatment for the trials consists in elimination of missing values, numerical feature normalisation, and uniquely hot coding of classification features. The dataset was eventually split into a test set to assess the model's performance and a training set utilised for model development.

Particularly for various attack kinds in IoT environment, the dataset is fit for assessing the performance of the anomaly detection technique suggested in this research.

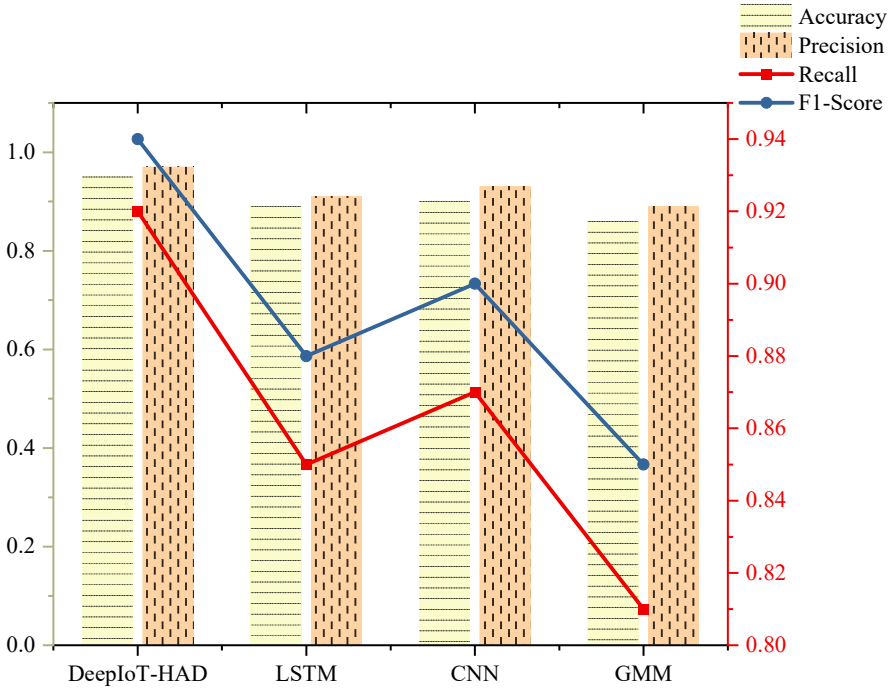
4.2 Experimental procedure

The experimental setup comprised of the same dataset for comparison, a hardware configuration of a computer with 16 GB of RAM and an NVIDIA GPU, a Python language software environment, LSTM model training using TensorFlow/Keras framework, and SVM training using the Scikit-learn library.

Comparative research between DL models and conventional techniques forms the first experiment. This work aims to validate the performance of the IoT anomaly

detection method based on DL and layered architecture, i.e., DeepIoT-HAD, compared with other traditional methods and DL, including Gaussian mixture models (GMMs), LSTMs, CNNs, etc., in order to evaluate the advantages of this framework in IoT data anomaly detection. Figure 2 displays the experimental outcomes.

Figure 2 Performance evaluation of DeepIoT-HAD and comparison with other models (see online version for colours)

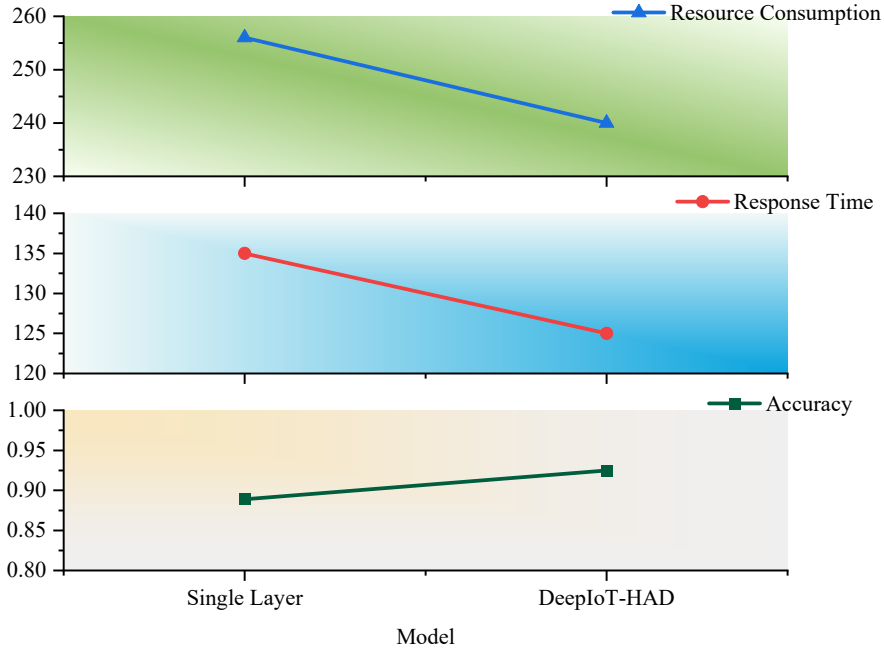


DeepIoT-HAD shows good performance in all the assessment metrics from the experimental data, particularly in terms of accuracy, recall, and F1 score, which are much higher than those of other conventional techniques and D. DeepIoT-HAD's accuracy of 0.95 is notably greater than that of LSTM (0.89), CNN (0.90), and GMM (0.86), therefore indicating that DeepIoT-HAD is able to more correctly categorise normal and aberrant data. DeepIoT-HAD's recall of 0.92 is likewise much higher than LSTM (0.85), CNN (0.87), and GMM (0.81), implying that the technique is more suited to detect aberrant data and more able to catch most of the anomalies. With a 0.97 accuracy level far greater than that of the other models, almost all of the data indicate are actual anomalies with a minimal false alarm rate when anomalies are found. At last, DeepIoT-HAD has an F1 score of 0.94, which is likewise better than other models, particularly GMM (0.85), so demonstrating the advantage of the framework in balancing the accuracy and recall of anomaly detection.

Overall, the experimental results reveal that DeepIoT-HAD has great performance in IoT anomaly detection, particularly in terms of recall and accuracy, thereby supporting the efficacy of the method in managing challenging data and high-dimensional anomaly detection chores.

Comparative experiments between single-layer and multi-layer layered architecture constitute the second one. This work aims to validate, particularly in terms of detection accuracy, computational complexity, and reaction time, the benefits of multilevel layered architecture in IoT anomaly detection system. We will thus compare single-tier architecture and multi-layer layered architecture for training independently and assess their effectiveness on certain assessment criteria. Figure 3 displays the experimental findings.

Figure 3 Comparison of performance between single-layer and DeepIoT-HAD (see online version for colours)



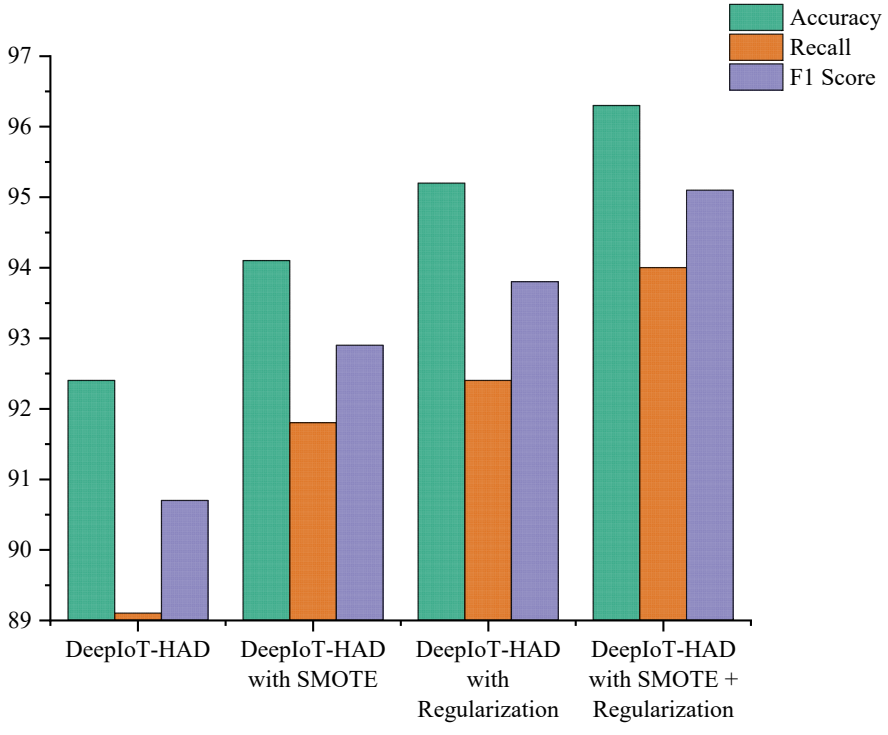
In terms of accuracy, response time, and computing resource consumption, the comparison experiments reveal that the multilevel layered architecture (DeepIoT-HAD) shows appreciable benefits over the single-layer architecture. DeepIoT-HAD surpasses the 88.9% of the single-layer design with a detection accuracy of 92.5%, which is more accurate. DeepIoT-HAD responds 125 seconds to process every piece of data, while single-layer architecture responds 135 seconds. DeepIoT-HAD still demonstrates improved response time, especially when processing big-scale datasets, even if the two differ in not much terms; the advantage of layered architecture is more clear. DeepIoT-HAD shows the capacity of the layered architecture in optimising computational resource consumption by using less memory than the single-layer architecture, 240 MB rather than 256 MB.

The third experiment is the model optimisation and data enhancement one. This experiment aims to investigate how methods of data augmentation and model optimisation could help to raise IoT anomaly detection performance. We assess the influence of several data enhancement techniques (e.g., SMote, data resampling, etc.) on

model performance by combining them with optimisation strategies like regularisation and learning rate adjustment and so increase the diversity of training data.

In this experiment, we first compare the datasets using data enhancement with those without data enhancement, and train alternative models depending on this in combination with optimisation strategies such regularisation and learning rate modification. We want to confirm the efficiency of data augmentation and optimisation strategies in raising anomaly detection performance by means of comparison of the experimental outcomes. Figure 4 shows the experimental outcomes.

Figure 4 Comparison of performance with without data augmentation and optimisation in DeepIoT-HAD (see online version for colours)



The trials reveal that boosting the performance of the DeepIoT-HAD model depends much on data enhancement (SMote) and model optimisation (e.g., regularisation). The accuracy, recall, and F1 value all rise during SMote data enhancement; the enhancement performance is particularly notable in recall and F1 value. This suggests that improving data helps the model to identify several kinds of abnormalities with efficiency. Regularisation helps to solve the overfitting issue of the model and raises general performance with an accuracy of 95.2%. With accuracy, recall, and F1 values of 96.3%, 94.0%, and 95.1%, respectively, the model displays the best results when both SMote and regularisation are applied, so proving the power of DeepIoT-HAD with the mix of data enhancement and optimisation techniques.

Particularly in practical applications, where enhancement and optimisation means are of great relevance, the results of this experiment show that methods of data enhancement and model optimisation not only improve the detection accuracy of the model but also essentially increase its ability to generalise to anomalous data.

Taken together, DeepIoT-HAD not only innovates in the model structure and improves the accuracy and efficiency of anomaly detection through a multi-level layered architecture, but also combines the data enhancing and model optimising techniques to make the system more robust and efficient in handling the task of anomaly detection in complex IoT environments.

5 Conclusions

In this work, we offer DeepIoT-HAD, an IoT anomaly detection technique grounded on DL and layered architecture. We essentially increase the accuracy and computational efficiency of the anomaly detection system in the IoT environment by creatively merging layered architecture with DL. This work not only provides effective detection of abnormal activities of IoT devices but also considers the real-time and scalability of the system to guarantee good performance in large-scale data transmission and complex situations.

Still, there are certain limits to this study. First of all, the utilised dataset is somewhat basic; hence, it is advisable to consider in the future more complicated and varied IoT datasets to test the performance of the model even further. Second, although the layered design works better in this study, how to choose the suitable layers and model parameters depending on the particular case when applied in practice still has to be investigated more. Furthermore, especially for applications in large-scale IoT contexts, future optimisation still revolves on the computational complexity and resource consumption of the model.

Future studies can delve further in the following spheres:

1 further improve the layered architecture's design to fit the data transmission and processing needs in more pragmatic settings

2 add more sophisticated DL as graph neural networks (GNN) and transformer to increase model generalisation and detection even more

3 edge computing along with distributed computing can be investigated to solve the contradiction between resource use and real-time in IoT systems.

Finally, DeepIoT-HAD offers a realistic and effective way for IoT anomaly detection with great possibility for development prospects and useful applications.

Declarations

All authors declare that they have no conflicts of interest.

References

Alaa, M., Zaidan, A.A., Zaidan, B.B. et al. (2017) 'A review of smart home applications based on internet of things', *Journal of Network and Computer Applications*, Vol. 97, pp.48–65.

Allioui, H. and Mourdi, Y. (2023) 'Exploring the full potentials of IoT for better financial growth and stability: a comprehensive survey', *Sensors*, Vol. 23, No. 19, p.8015.

Chen, Y., Jiang, H., Li, C. et al. (2016) 'Deep feature extraction and classification of hyperspectral images based on convolutional neural networks', *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 54, No. 10, pp.6232–6251.

Chevrot, A., Vernotte, A. and Legeard, B. (2022) 'CAE: contextual auto-encoder for multivariate time-series anomaly detection in air transportation', *Computers & Security*, Vol. 116, p.102652.

Erhan, L., Ndubuaku, M., Di Mauro, M. et al. (2021) 'Smart anomaly detection in sensor systems: a multi-perspective review', *Information Fusion*, Vol. 67, pp.64–79.

Fan, C., Sun, Y., Zhao, Y. et al. (2019) 'Deep learning-based feature engineering methods for improved building energy prediction', *Applied Energy*, Vol. 240, pp.35–45.

Habeeb, R.A.A., Nasaruddin, F., Gani, A. et al. (2019) 'Real-time big data processing for anomaly detection: a survey', *International Journal of Information Management*, Vol. 45, pp.289–307.

Khattak, H.A., Shah, M.A., Khan, S. et al. (2019) 'Perception layer security in internet of things', *Future Generation Computer Systems*, Vol. 100, pp.144–164.

Kobo, H.I., Abu-Mahfouz, A.M. and Hancke, G.P. (2017) 'A survey on software-defined wireless sensor networks: challenges and design requirements', *IEEE Access*, Vol. 5, pp.1872–1899.

Lin, J., Yu, W., Zhang, N. et al. (2017) 'A survey on internet of things: architecture, enabling technologies, security and privacy, and applications', *IEEE Internet of Things Journal*, Vol. 4, No. 5, pp.1125–1142.

Liu, Y., Zhou, Y., Yang, K. et al. (2023) 'Unsupervised deep learning for IoT time series', *IEEE Internet of Things Journal*, Vol. 10, No. 16, pp.14285–14306.

Ma, X., Wu, J., Xue, S. et al. (2021) 'A comprehensive survey on graph anomaly detection with deep learning', *IEEE Transactions on Knowledge and Data Engineering*, Vol. 35, No. 12, pp.12012–12038.

Mrabet, H., Belguith, S., Alhomoud, A. et al. (2020) 'A survey of IoT security based on a layered architecture of sensing and data analysis', *Sensors*, Vol. 20, No. 13, p.3625.

Patcha, A. and Park, J-M. (2007) 'An overview of anomaly detection techniques: existing solutions and latest technological trends', *Computer Networks*, Vol. 51, No. 12, pp.3448–3470.

Rehman, M.H., Liew, C.S., Abbas, A. et al. (2016) 'Big data reduction methods: a survey', *Data Science and Engineering*, Vol. 1, pp.265–284.

Rodriguez-Galiano, V., Sanchez-Castillo, M., Chica-Olmo, M. et al. (2015) 'Machine learning predictive models for mineral prospectivity: an evaluation of neural networks, random forest, regression trees and support vector machines', *Ore Geology Reviews*, Vol. 71, pp.804–818.

Sewak, M., Sahay, S.K. and Rathore, H. (2020) 'An overview of deep learning architecture of deep neural networks and autoencoders', *Journal of Computational and Theoretical Nanoscience*, Vol. 17, No. 1, pp.182–188.

Shrestha, A. and Mahmood, A. (2019) 'Review of deep learning algorithms and architectures', *IEEE Access*, Vol. 7, pp.53040–53065.

Tien, J.M. (2017) 'Internet of things, real-time decision making, and artificial intelligence', *Annals of Data Science*, Vol. 4, pp.149–178.

Ullah, I. and Mahmoud, Q.H. (2021) 'Design and development of a deep learning-based model for anomaly detection in IoT networks', *IEEE Access*, Vol. 9, pp.103906–103926.

Wu, D., Xu, H., Jiang, Z. et al. (2021) 'EdgeLSTM: towards deep and sequential edge computing for IoT applications', *IEEE/ACM Transactions on Networking*, Vol. 29, No. 4, pp.1895–1908.