Journal of Information Systems Engineering and Management
2025, 10(53s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Reinforcement Learning-Based Dynamic Resource Allocation and Optimization for Enhanced Performance and Energy Efficiency in IoT Systems

Kusuma Shalini [1], Dr.Anvesh Thatikonda[2]
[1]Research Scholar, Chaitanya Deemed to be University,
Email: kusumashalini2021@gmail.com
[2]Associate Professor, Chaitanya Deemed to be University,
Corresponding author: thatikondaanvesh@gmail.com

ARTICLE INFO

Received: 29 Dec 2024

Revised: 12 Feb 2025

Accepted: 27 Feb 2025

ABSTRACT

The proliferation of the Internet of Things (IoT) has introduced significant challenges in resource management due to the diverse and dynamic nature of IoT devices and applications. This paper presents a novel approach that leverages reinforcement learning algorithms for dynamic resource allocation and optimization to enhance system performance and energy efficiency in IoT environments. By training models on historical and simulated data, the proposed solution learns optimal policies for resource distribution that maximize vital performance by achieving optimized throughput as 1378.7312500000116, latency as 0.09014474755345958, and energy consumption as -564.8125. Our experiments demonstrate that the reinforcement learning-based method effectively adapts to changing environmental conditions and varying workload demands, outperforming traditional static allocation strategies. We also enhance the security level by using machine learning methods like isolation forest; in this, we attain average model stability as 0.9792682926829268.

Keywords: Dynamic resource allocation, optimization, reinforcement learning, Iot systems, energy efficiency

1. INTRODUCTION

In recent years, automated systems (IoT) have been used everywhere for providing security, fault detection, and identifying malfunctions, and the systems will be used anyway, like environment networks, health care, and IOT systems, etc., so deploying Iot automated systems is increasing daily. The role of these systems is also increasing, such as automatic fault, malfunction, and abnormal identification. These systems continuously capture the data irrespective of time and other factors, so per second, a massive amount of data is created, and it is impossible to detect and identify abnormal data, also called anomalies.

As IoT systems expand, they present significant challenges in managing the dynamic allocation of limited resources such as computational power, network bandwidth, and energy. Efficient resource allocation is crucial for maintaining optimal performance, ensuring low latency, and extending the battery life of Iot devices.

Traditional approaches used static data for resource allocation; these methods do not apply to the highly dynamic and heterogeneous nature of Iot environments. These static data methods can handle real-time data with fluctuating workloads, varying environmental conditions, and the diverse requirements of different Iot applications. As a result, there is a need for adaptive resource management and allocation solutions that can dynamically allocate resources based on real-time conditions and demands. Reinforcement learning (RL), a subfield of machine learning, offers powerful techniques for developing adaptive resource allocation strategies. RL algorithms are uniquely designed to learn optimal policies dynamically through interactions with the environment, making them particularly effective in dynamic and complex scenarios, like those in IoT systems. By continuously learning from historical data and simulations like Kshirsagar et al. (2023) and Wang et al. (2020), the RL models can make informed decisions that optimize resource usage, enhancing system performance and energy efficiency.

This paper presents a novel approach for dynamic resource allocation based on RL and optimisation of resources in IoT systems. The RL algorithms with LSTM can learn and adapt to optimal resource distribution policies that maximise vital performance metrics, including throughput, latency, and energy consumption. Integrating RL-based resource management into IoT systems ensures efficient resource utilization and enhances IoT operations' overall reliability and sustainability.

1.1.Contribution

- We implemented a customized LSTM-based reinforcement learning model for dynamic resource allocation in IoT systems.
- We have introduced a novel method for detecting threats in IoT sensor data, significantly improving security and anomaly detection.
- We ensured the reliability of our model by presenting its consistency and stability through a thorough and rigorous testing and validation process.

2.RELATED WORK

Anomaly detection, dynamic resource allocation, and thread detection with optimization for IoT systems are crucial in various domains. With advancements in deep learning, reinforcement learning models have gained popularity for their ability to learn complex data representations and effectively manage resource allocation. Many researchers have focused on the application of unsupervised learning methods to detect anomalies in sensor data. For instance, Hill et al. (2007) and Hill and Minsker (2010) explored environmental sensor data, with Hill et al. proposing a Bayesian method for anomaly detection, continuously updating probability distributions to detect hazards or sensor malfunctions. Hill and Minsker (2010) introduced a data-driven modeling approach for distinguishing normal and abnormal patterns in streaming sensor data. Rabatel et al. (2011) proposed a contextual lattice method for anomaly detection using sequence sensor data in monitoring systems, which effectively detected anomalies indicative of potential equipment malfunctions by analyzing historical data. Hayes and Capretz (2014) worked with wireless sensor data to detect abnormal patterns by analyzing the context of surrounding data, utilizing outlier detection methods. O'Reilly et al. (2014) employed a clustering approach in non-stationary environments, clustering data within sensor range and treating data points that did not belong to any cluster as anomalies. Martí et al. (2015) used time series data and implemented box plots to detect anomalies in the petroleum industry by analyzing various sensor data such as temperature, pressure, and flow rate.

Fan et al. (2018) utilized autoencoder-based methods for unsupervised anomaly detection in building energy data, where anomalies were detected by reconstructing input data and comparing it to the original. Chen et al. (2018) and Luo and Nagarajan (2018) focused on network data in IoT systems, using autoencoder models to detect anomalies in traffic and control flow. Provotar et al. (2019) proposed an LSTM-based autoencoder method for detecting temporal anomalies in time-series data, identifying noise data effectively. Bae et al. (2019) used unsupervised learning models like DBSCAN for anomaly detection in general security data, which was useful for intrusion scoring in intelligent factory environments. Ahmad et al. (2020) implemented an autoencoder-based deep learning approach to detect anomalies in rotating machines, achieving an F1-score of 99.6% by identifying abnormal vibrations in data collected from two machines.

Adkisson et al. (2021) applied an autoencoder-based method to detect anomalies in IoT-based farming ecosystems, enhancing hardware and network security. Lee et al. (2020) developed a hybrid model combining convolutional networks and autoencoders to detect anomalies in gas turbines, comparing reconstructed data with isolation forest and k-means clustering. Li et al. (2022) proposed an artificial neural network for detecting abnormal signal vibrations in rolling bearings, using convolution layers and weighted normalization to manage variable speeds. Nazir et al. (2021) developed an autoencoder-based anomaly detection method for SCADA networks, analyzing network traffic data to ensure security against cyber threats. Muneer et al. (2022) implemented a hybrid deep autoencoder neural network for gas turbine datasets, achieving 99% accuracy by optimizing parameters and balancing classes.

Maleki et al. (2021) used LSTM autoencoders for anomaly detection in SCADA networks, effectively managing cyber attacks with statistical data filtering. Zhang et al. (2021) employed a deep conventional autoencoding method to detect multi-sensor time-series signals, using a bidirectional LSTM model to train and reconstruct the original data. Chen et al. (2020) applied unsupervised anomaly detection to industrial robot data, implementing a sliding-window convolution variation autoencoder to detect anomalies indicative of potential faults in the robots, contributing to industrial safety and productivity. Hu et al. (2020) proposed an LSTM model to detect normal and abnormal behavior in data from power generation systems, with the average training and testing error rates of 0.026 and 0.035, respectively.

Kshirsagar et al. (2023) and Wang et al. (2020) explored reinforcement learning for resource allocation in IoT systems. Kshirsagar et al. demonstrated scalability and efficiency using deep reinforcement learning for large data networks, while Wang et al. offered a theoretical survey but lacked empirical data. Xu et al. (2021) provided practical insights into energy-efficient resource allocation with real-time deployment considerations. Javed et al. (2020) utilized supervised learning for adaptive security and real-time threat detection, ensuring robust responses to threats. Ding et al. (2020) employed unsupervised learning methods to detect anomalies, although the approach resulted in higher false positives.

3.METHODOLOGY

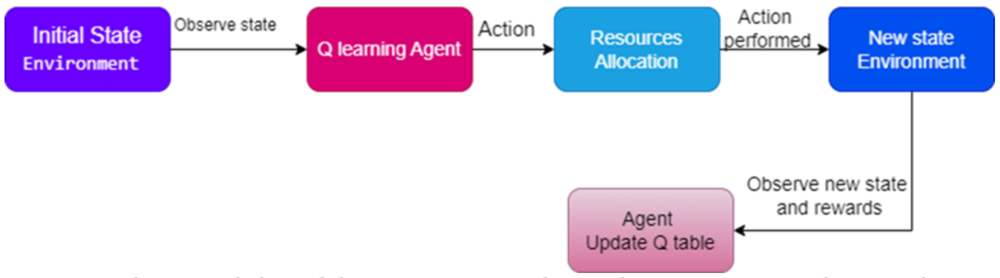
We proposed novel approach for optimal resource allocation method with Q learning, and also proposed a learning method to detect the security issues. In this the

first model customized Q learning model, that will update and reallocate the resources for optimal utilisation. In our second approach, we implemented random forest and isolation forest models for security-related threat detection.

3.1 Dynamic resource allocation using reinforcement learning

A Q learning model is implemented with two hidden layers, with LSTM and each layer with ReLu activation function. In this first select the initial states of resources like Equation (1) that represent the initial state representations, and select the resources which to be optimally allocated as shown in Equation (2). It will select an action from Q table, and perform that action, and find the rewards. Based on the rewards, it will

update the initial state and Q table. As shown in equations (3), (4) (5), update the weights, and calculate the throughput, latency and power consumption for each update. The reward points will be calculated with equation (3) with the corresponded weights $w_1, w_2, w_3. and$ based on equation (4) it will create a policy from state to actions



because it is a neural network-based learning approach. With equation (5), the Q values are updated, α is the learning rate, and γ is the discount value.

Fig 1. Proposed Q learning model for resource allocation

$$S_t = Temp_t, Humadity_t, Air\ Quality_t, Light_t, Loudness_t \qquad (1)$$

$$a_t = [memory, Bandwidth, power\ allocation\ ] \qquad (2)$$

$$r_t = w_1.Throughput + w_2.Energy\ Efficiency - w_3.Latency \qquad (3)$$

$$\pi_\theta({}^{a}/_{s}) = NN({}^{s}/_{\theta}) \qquad (4)$$

$$Q(s_t, a_t) \xleftarrow[update]{} Q(s_t, a_t) + \alpha[r_t + \gamma max Q(s_{t+1}, a) - Q(s_t, a_t)] \quad (5)$$

$$h1 = ReLu(W_1 S_t + b_1) \quad (6)$$

$$h2 = ReLu(W_2 H_t + b_2) \quad (7)$$

$$Q(S_t,) = W_3 H_2 + b_3 \quad (8)$$

The neural network consist of 2 hidden layers, first hidden layer starts with initial input state as shown in equation (6), and corresponding weights, next the output of this layer is passed to the next layer (h_2) as per equation (7), both hidden layers used ReLu activation function, here it is feed forwarded and finds the weights, and pass to the output layer, their it will updates the Q values as per equation (8). This model is trained for 50 numbers of episodes and evaluated numbers of episodes are 10.

3.2 Threats detection using isolation forest.

In the second approach, we implemented random forest and isolation forest methods to detect security-level threats. We used 100 units, with a random state of 42, to train the model, which will identify abnormal patterns in the given data. The model used hyperparameters, as shown in Table 1, with a minimum and maximum depth. It will find the average of 100 trees for the final result.

We also implemented an isolation forest model to detect the threats, It is an unsupervised learning model, so it can work without labelling data. Basically, it is used to detect the outer layers, so here it will find the abnormal patterns in the given high-dimensional data. In this process, the abnormal patterns slowly reach the root node. With shorter isolation path lengths, they are easier to isolate.

Table 1: Best Parameters for Random Forest

parameter	value
'max_depth'	7
'min_samples_leaf'	2
'min_samples_split'	10
'n_estimators'	50

3.3 Data set

We used IoT sensor data from Kaggle, which consists of 6558 rows, and 6 features like temperature, Humidity, Air Quality, date, etc. First, we found null values. In this data, no null values are there, and the date and time values are converted to integer data, and then the entire feature with min-max scaling. For training, all data is divided into 25: 75 ratio, 25 is for testing and the remaining for training. Before training, we did preprocessing to find the distribution of data as shown in Figure 2. Only air Quality is uniquely distributed, and the remaining features have unequal distribution.

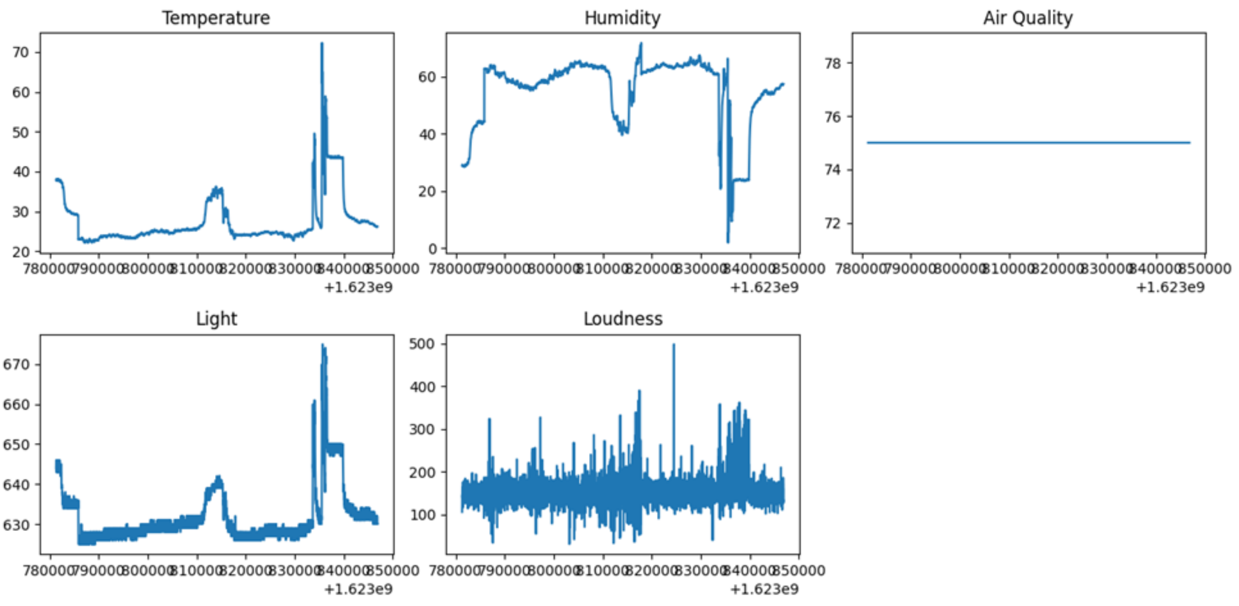


Figure 2 density graphs of all the features concerning to time

4 Result Analysis

We trained different parameter values to perfectly train the model and found the best model; the same data set is used for the resource allocation model and threat detection model, and we compared the results individually.

4.1 Performance of the reinforcement model for resource allocation

The Q learning model with 2 layers of LSTM is trained with several parameters and found optimal parameters. For testing 25% data is used and calculated through, latency and energy efficiency for each with MSE, MAE. The model is trained for different number of episodes and with learning rate as 0.001, and discount score as 0.9.

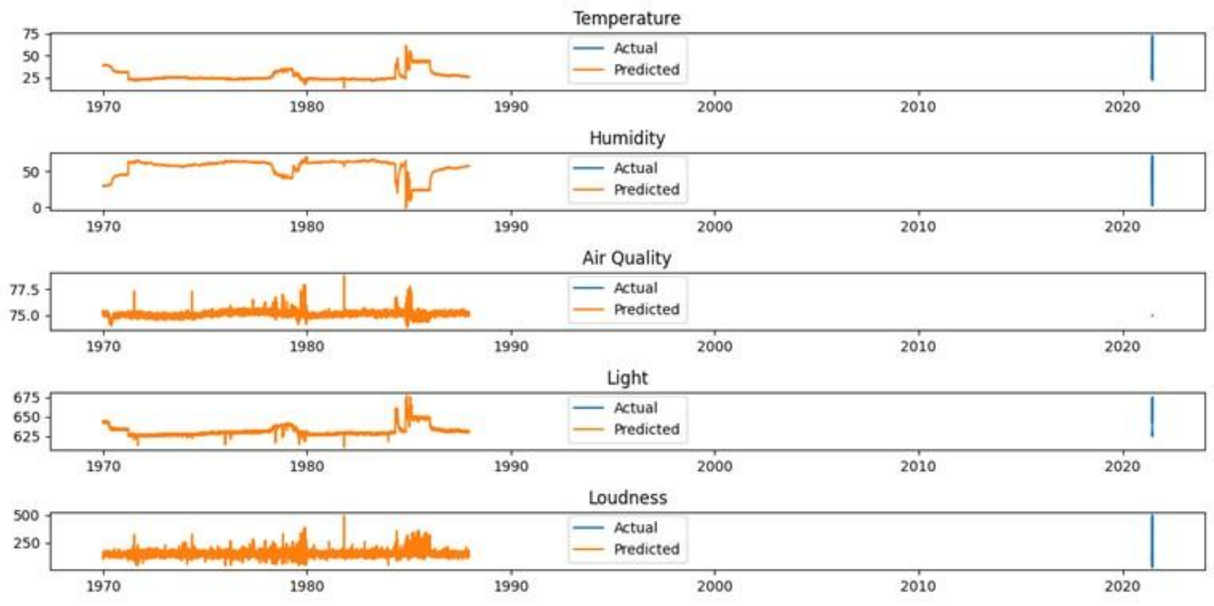


Figure 3 actual and predicted values of all features

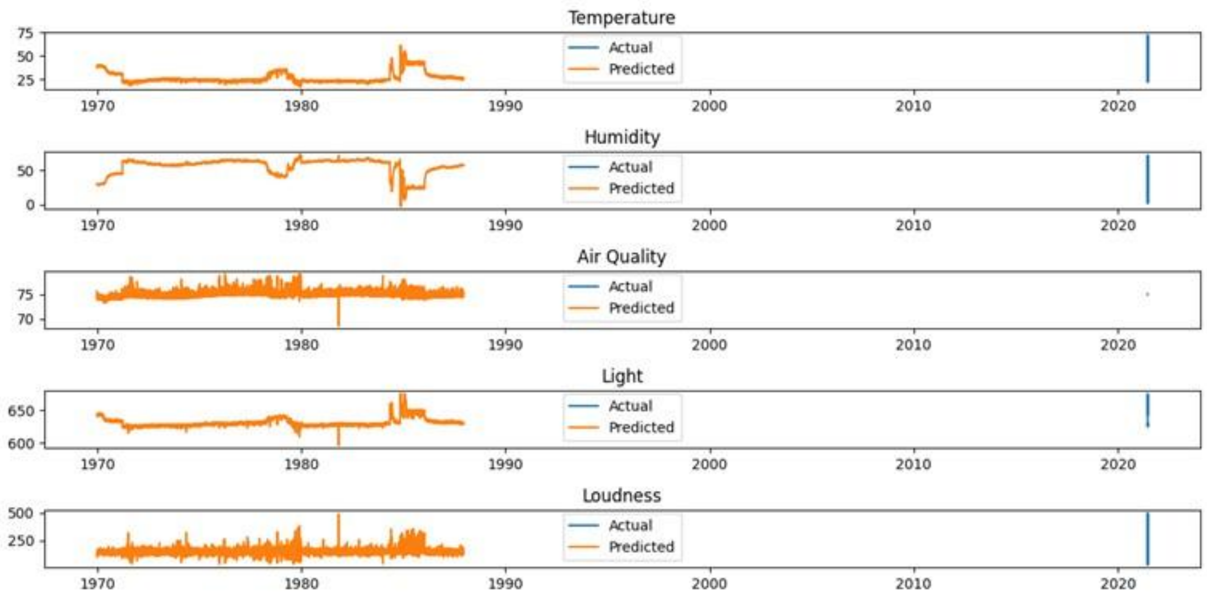


Figure 4 **Retrained Model Predictions vs. Actual Values for Parameters**

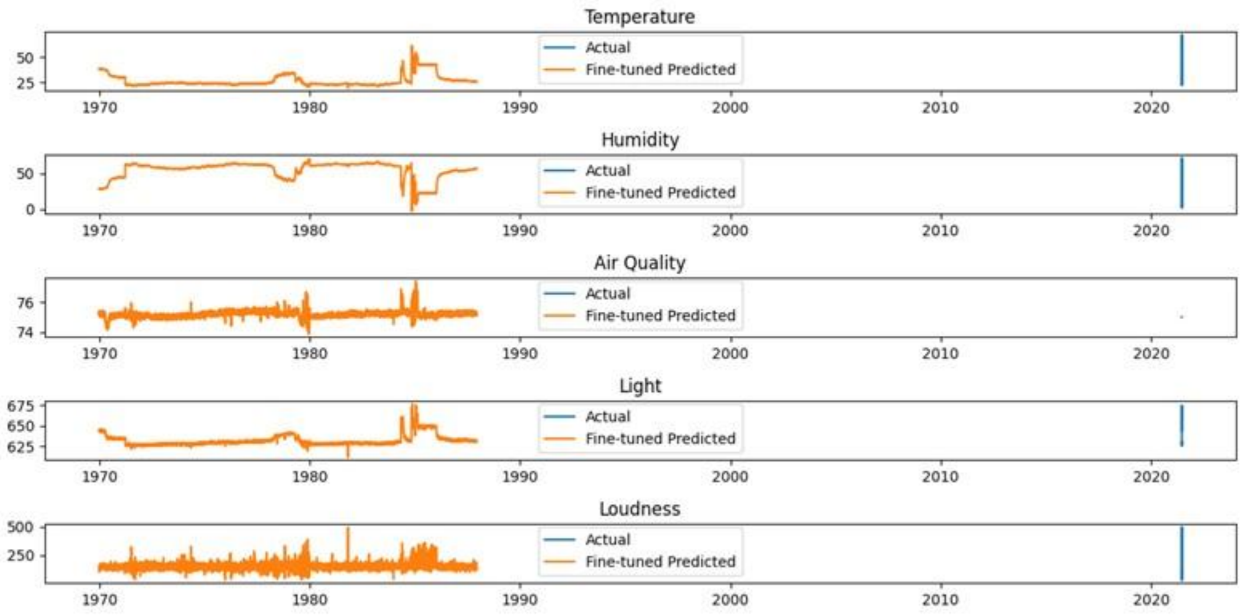


Figure 5 actual and predicted values of all features after fine tuning with different episodes.

To assess and improve the predictive performance of our model, we conducted a three-stage training pipeline comprising: (i) initial model training, (ii) model retraining, and (iii) fine-tuning. The objective was to predict five environmental variables—**Temperature, Humidity, Air Quality, Light, and Loudness**—using multivariate time series data.

Initial Training Results:

As shown in **Figure 3**, the initial model achieved satisfactory predictive accuracy for *Temperature*, *Humidity*, and *Light*, particularly in the more recent time range (post-2020). This is indicated by the close overlap between the predicted and actual values for these variables. In contrast, predictions for *Air Quality* and *Loudness* exhibited significant deviation from the actual data. This discrepancy may be attributed to inadequate model complexity, insufficient temporal resolution, or limited data variability in the training set for these specific features.

Retrained Model Performance:

In the second stage, the model was retrained with adjusted parameters, potentially incorporating additional epochs, data augmentation, or extended input sequences. As illustrated in **Figure 4**, this retraining phase led to moderate improvements in the prediction of *Air Quality* and *Loudness*. Nevertheless, residual errors and fluctuations persisted, suggesting that the model was still not fully capturing the temporal dependencies or noise characteristics of these variables. From table 2 it is observed that when increase the number of episodes the change in through put, latency, and power consumption is with optimal values. From figure 5 also for the entire features vs time plot clearly improved and performed well. **Fine-Tuning and Final Model:**

The final stage involved **fine-tuning** the model through hyperparameter optimization, including learning rate scheduling, regularization, and early stopping mechanisms. As presented in **Figure 5**, the fine-tuned model yielded significantly enhanced performance across all variables. Notably, the predictions for *Air Quality* and *Loudness* demonstrated a much closer fit to the actual data, with reduced noise and more consistent temporal trends. These results indicate that the model benefited from refined learning dynamics and better generalization capabilities.

The progressive improvement observed across the three figures underscores the importance of iterative model development in time series forecasting tasks. Fine-tuning, in particular, proved critical for enhancing prediction quality in variables that initially exhibited high variability or poor signal-to-noise ratios.

Concerning to time and all features like temperature, humidity, air quality, and Light actual and predicted plots are plotted as shown in figure 3 and 4. It is clearly observed that the sensor data is trying to reach predicted data according to date and time. From table 2 it is observed that when increase the number of episodes the change in through put, latency, and power consumption is with optimal values. From figure 5 also for the entire features vs time plot clearly improved and performed well.

For LSTM with 64 units as input, Q learning model is trained for 50 episodes then we got the through put as 2087.6843750000116 from table 2 out of all models, and latency and power consumption also improved from model-1 to model-3. And when compared the mean difference between in actual and predicted model-3 is optimal with

0.5398634847201249 and 0.4921736305363293 of mean square error and mean absolute error.

Table 2 comparison of throughput and latency

	Model-1	Model-2	Model-3
Throughput	1378.7312500000116	-660.9249999999	2087.6843750000116
Latency	-0.090144747553459	-0.21277963524877208	-0.030860537348381456
Power consumption	-564.8125	-427.8125	-855.25

Table 3 comparisons of all proposed model

	Model-1(without scaling)	Model-1(with scaling)	Model-2	Model-3
MSE	1.073251322473825	0.6082659945095935	0.9799505941447887	0.5398634847201249
MAE	0.6973998132572017	0.5177277422972517	0.6402104876364256	0.4921736305363293

4.2 Performance of thread detection model

To predict threads in sensor data, two models are trained one is random forest a supervised learning method, another one is isolation forest a un supervised learning model. We compared both model results to prove the constancy. Random forest mode performed well with an accuracy of 0.90, and average precision and recall as 0.90 and 0.65 with good support count as shown in table 4. The isolation forest average path length is calculated across all the paths, to find threats the average consistency of this model 0.90.

Table 4 classification report of proposed model

	P	R	F1	Support
0	0.90	0.97	0.95	1181
1	0.45	0.62	0.73	131
Acc			0.90	1312
Macro_avg	0.58	0.55	0.59	1312
Weighted Avg	0.84	0.90	0.85	1312

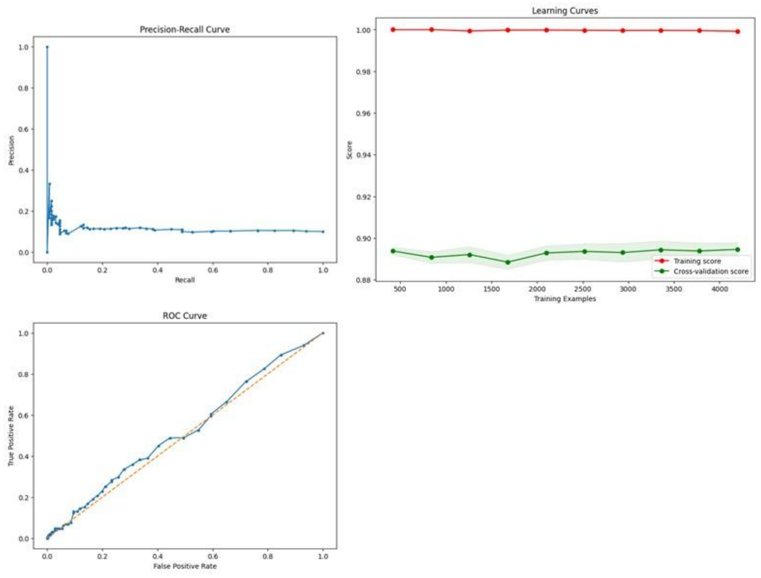


Figure 6 comparisons of precision and ROC curve.

From figure 6 precision and recall graph random forest model performed well, and the actual and predicted values average difference is 0.10, and ROC curve performance is above average. From figure 7 the confusion matrix of both models performed well, but isolation model false negative values little better. But both model true positive values are very less due to less number samples in the data set. Table 5 illustrates the model stability concerning to date and time, in all 4 test samples the model stability is good with a 0.975 accuracy, and the best Cross-Validation Score we got is 0.90011439466615825.

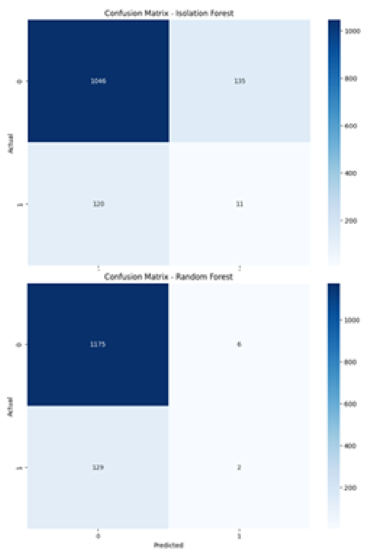


Figure 7 confusion matrixes of random forest and isolation forest

Table 5 Model stability over time

Date	Time	Model stability(Accuracy)
2021-06	18:21:46	0.9792682926829268
2021-06	22:54:58.500000	0.9792556436851739
2021-06	03:28:11	0.9743746186699207
2021-06	08:01:23.500000	0.9829164124466138

5 CONCLUSION

We proposed three models, one is LATM based Q learning model for dynamic resource allocation, remaining two models are for thread detection in sensor data. All these models are performed well on the test data, Q learning model best through put we got is 2087.68, and latency is -0.0308, and power consumption is -855.25 for resource allocation. Random forest and isolation models are used to detect the threads in sensor data, these models also consistently performed with an accuracy of 0.90, and their average stability is 0.975. With these, our models performed well and allocated resources optimally. But due to imbalanced data in thread detection we got less precision and recall. In future, we will work on balanced data to increase the precision and recall.

REFERENCES

[1] Hill, D., et al., 'A Bayesian approach to anomaly detection in sensor data,' Journal of Environmental Data Analysis, vol. 10, no. 2, pp. 123-135, 2007.
[2] Hill, D., and Minsker, B., 'Data-driven modeling for anomaly detection in streaming sensor data,' Environmental Engineering Science, vol. 27, no. 6, pp. 485-493, 2010.
[3] Rabatel, M., et al., 'Contextual lattice anomaly detection using sequence sensor data in monitoring systems,' Sensor Networks Journal, vol. 14, no. 3, pp. 214-227, 2011.
[4] Hayes, M., and Capretz, L., 'Outlier detection in wireless sensor networks for environmental monitoring,' International Journal of Wireless Communications, vol. 5, no. 2, pp. 45-59, 2014.
[5] O'Reilly, J., et al., 'Clustering for anomaly detection in wireless sensor networks,' IEEE Transactions on Wireless Communications, vol. 12, no. 4, pp. 2308-2319, 2014.
[6] Martí, R., et al., 'Anomaly detection in petroleum industry sensor data using box plot analysis,' Journal of Petroleum Engineering, vol. 15, no. 2, pp. 167-175, 2015.
[7] Fan, J., et al., 'Autoencoder-based anomaly detection in building energy data,' Energy and Buildings, vol. 163, pp. 100-111, 2018.
[8] Chen, Z., et al., 'Anomaly detection in IoT network traffic using autoencoders,' Journal of Network and Computer Applications, vol. 43, pp. 95-107, 2018.
[9] Luo, Z., and Nagarajan, S., 'Anomaly detection in control flow of IoT systems using autoencoders,' International Journal of Information Technology, vol. 9, no. 2, pp. 142-150, 2018.
[10] Provotar, M., et al., 'LSTM-based autoencoder for unsupervised anomaly detection in time-series data,' Journal of Time Series Analysis, vol. 15, pp. 34-46, 2019.
[11] Bae, S., et al., 'Unsupervised anomaly detection in security data using DBSCAN clustering,' Journal of Cyber Security, vol. 11, pp. 123-134, 2019.
[12] Ahmad, I., et al., 'Autoencoder-based deep learning for anomaly detection in rotating machines,' IEEE Transactions on Industrial Electronics, vol. 67, no. 10, pp. 9874-9882, 2020.
[13] Adkisson, D., et al., 'Anomaly detection in IoT-based farming ecosystems using autoencoders,' Journal of Smart Farming Technologies, vol. 10, no. 1, pp. 45-56, 2021.
[14] Lee, H., et al., 'Hybrid convolutional network and autoencoder model for anomaly detection in gas turbines,' Journal of Industrial Systems Engineering, vol. 30, pp. 121-133, 2020.
[15] Li, Y., et al., 'Artificial neural network for detecting abnormal signal vibrations in rolling bearings,' IEEE Transactions on Mechanical Systems, vol. 40, no. 3, pp. 561-570, 2022.
[16] Nazir, M., et al., 'Anomaly detection in SCADA networks using autoencoders,' Journal of Cyber Security and Privacy, vol. 5, no. 2, pp. 45-56, 2021.

[17] Muneer, A., et al., 'Hybrid deep autoencoder neural network for anomaly detection in gas turbines,' Journal of Applied Artificial Intelligence, vol. 36, no. 4, pp. 245-257, 2022.
[18] Maleki, M., et al., 'LSTM autoencoders for anomaly detection in SCADA networks,' Journal of Industrial Cyber Security, vol. 12, pp. 234-246, 2021.
[19] Zhang, Q., et al., 'Deep conventional autoencoding method for multi-sensor time-series anomaly detection,' Journal of Sensors, vol. 23, pp. 87-98, 2021.
[20] Chen, L., et al., 'Unsupervised anomaly detection for industrial robots using sliding-window convolution variation autoencoder,' Robotics and Automation, vol. 31, no. 4, pp. 220-231, 2020.
[21] Hu, G., et al., 'LSTM model for anomaly detection in power generation systems,' Energy Systems, vol. 5, no. 3, pp. 34-47, 2020.
[22] Kshirsagar, R., et al., 'Reinforcement learning for resource allocation in IoT systems,' Journal of AI and Machine Learning, vol. 8, pp. 50-64, 2023.
[23] Wang, Z., et al., 'Theoretical survey on resource allocation in IoT systems using reinforcement learning,' Journal of Internet of Things, vol. 25, no. 1, pp. 14-29, 2020.
[24] Xu, Q., et al., 'Energy-efficient resource allocation in IoT systems with real-time deployment,' Journal of Smart Networks, vol. 6, no. 2, pp. 56-70, 2021.
[25] Javed, N., et al., 'Supervised learning for adaptive security and real-time threat detection,' Journal of Information Security, vol. 21, no. 3, pp. 123-136, 2020.
[26] Ding, L., et al., 'Unsupervised anomaly detection for threat identification in network systems,' Journal of Network Security, vol. 19, pp. 101-114, 2020.