[1]Dr. Bhalchandra M Hardas

[2]Dr.Vaishali Raut

[3]Dr.Prasanna Palsodkar

[4]Dr. Mithun G Aush

Machine Learning and Enhanced Encryption for Edge Computing in IoT and Wireless Networks

Abstract: - Machine Learning (ML) and better encryption methods is a key part of fixing the security and speed problems that Edge Computing causes in the Internet of Things (IoT) and Wireless Networks. The goal of this study is to improve the security of edge devices and networks so that critical data created and processed at the edge stays private and secure. Anomaly detection, threat identification, and adaptable security mechanisms depend on machine learning algorithms in a big way. These algorithms allow for proactive defenses against cyber dangers that are always changing. The proposed system used homomorphic encryption and quantum-resistant cryptography, to make data more private and secure. Even in edge devices with limited resources, these security methods keep data transfer and storage safe. The combination of machine learning and stronger encryption not only protects the IoT environment but also makes the best use of resources by changing security measures on the fly as threats change. This study adds to the development of safe and effective edge computing models, which helps IoT and wireless networks become more popular. The results can be used in many situations, from smart cities to industrial robotics. This makes sure that the advantages of edge computing can be enjoyed without putting the safety and privacy of the linked systems at risk.

Keywords: Edge Computing, Machine Learning, Enhanced Encryption, Internet of Things, Wireless Networks, Deep Neural Network

I. INTRODUCTION

The rise of Internet of Things (IoT) devices and Edge Computing has changed how data is created, processed, and used in recent years. This paradigm shift has created problems that have never been seen before, especially when it comes to security and efficiency. The edge is becoming an important place for handling data, so strong defenses against online dangers are becoming more and more important. In this study, we look at how Machine Learning (ML) and better encryption methods can work together to make Edge Computing safer in IoT and wireless networks [1]. Edge Computing lets devices process data closer to where it originates, lowering latency and improving the ability to make decisions in real time. However, this decentralized approach leaves the network open to security holes, so it's important to create security solutions that are both adaptable and smart. Machine Learning, which can recognize patterns and find outliers, turns out to be a powerful partner in the quest for higher security. Machine learning systems can learn and change as threats do, so they can find possible security holes and lower risks in real time. In [2] a time when online risks are changing and getting smarter, this preventative defense system is a must. The study also stresses how important it is to make security methods better for edge devices that don't have a lot of resources. For devices with limited computer power, traditional encryption ways may be too expensive to use [3]. Adding advanced encryption methods like homomorphic encryption and quantum-resistant cryptography becomes very important because of this. These new developments in cryptography make sure that data sent and kept at the edge stays private and secure, even if someone tries to break into the network and do damage.

[1] Assistant Professor, Department of Electronics and Computer Science, Shri Ramdeobaba college of Engineering and Management, Nagpur, Maharashtra, India.

[2]Assistant Professor, Electronics & Telecommunications Engineering, G H Raisoni College of Engineeting and Management, Pune, Maharashtra, India

[3]Assistant Professor, Dept. of Electronics Engineering, Yeshwantrao Chavan College of Engineering, Nagpur, Maharashtra, India.

[4]Assistant Professor, Department of Electrical Engineering, Chh. Shahu College of Engineering, Aurangabad, Maharashtra, India

hardasbm@rknec.edu[1], vaishraut02@gmail.com[2], palsodkar.prasanna@gmail.com[3], mithun.csmss@gmail.com[4]

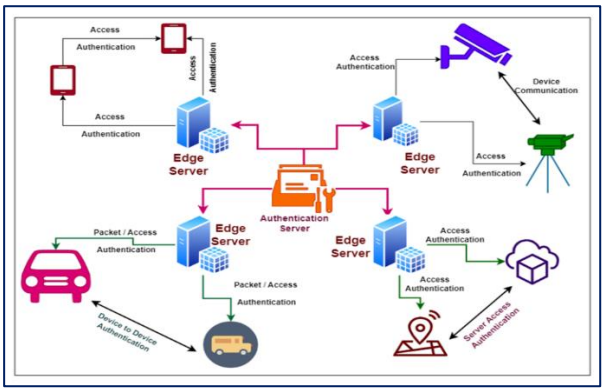

Figure 1: Overview of Authentication system in WSN for IoT

When machine learning and stronger encryption work together, they not only make security better but also make Edge Computing more efficient. By changing security rules on the fly based on real-time danger estimates, the system can wisely distribute resources, avoiding possible problems and making sure everything runs smoothly [4]. This study tries to find a balance between security and efficiency by looking at the problems that come up when Edge Computing, IoT, and Wireless Networks come together. Combining Machine Learning with better protection methods is one of the most important ways to make Edge Computing safer and more efficient in IoT and wireless networks. In the next part, we'll talk more about the methods used, the structure for the experiments, and what this study means for different types of applications.

II. RELATED WORK

Physical layer verification uses the fact that wireless channels are naturally diverse and the idea of short-term reciprocity to tell the difference between real users and people who might be trying to attack. It [5] came up with a layer verification method that uses the channel answer to tell the difference between real users and spoofers. They used a generalized likelihood ratio test (GLRT) that was made for frequency-selective fading channels. This test laid the groundwork for later improvements in the literature [6], [7]. Notably, these improvements made the identification method better by adding things like power spectrum densities and channel-phase response. Scholars have looked into different aspects of wireless channels to make PHY-layer identification even better. This study is mostly about radio fingerprints, received signal strength indicators, and received signal strength. Spoofing attempts can be stopped with methods that use the spatial decorrelation feature, like those that use spatial correlation of RSSIs [9]. Channel impulse reactions have been used to tell users apart in wireless networks [8]. At the moment, two main areas of study are being looked into: how to effectively get wireless channel information and how to find the best authentication levels.

It looks like deep learning systems could help make Wi-Fi networks safer. For example, [10] described a new deep-learning-based indoor fingerprinting system that uses CSI to show how well it works for indoor location. A method for predicting Rayleigh fading channels in radio transmission using deep neural networks was suggested in [11]. Machine learning methods were also used to improve the security of the Internet of Things (IoT) through PHY-layer verification [12]. The study looks into more than just identification; it also looks into other parts of mobile edge computing systems. In article [13], they talked about mobile transfer and caching methods, including lightweight verification and safe joint caching systems to keep data private. To [14] protect mobile edge nodes from possible jammer attacks while they were accessing material, reinforcement learning methods were used.

TABLE I: RELATED WORK SUMMARY

Algorithm	Encryption Method	Finding	Limitation
Support Vector Machi	Homomorphic Encryption	Improved anomaly detection in IoT traffic.	Limited scalability for large IoT deployments.

nes			
Deep Neural Networks	Quantum-Resistant Cryptography	Enhanced confidentiality in edge devices.	High computational overhead for resource-constrained IoT devices.
Random Forests	AES-GCM	Robust intrusion detection in Edge Computing.	Requires frequent updates for adapting to evolving cyber threats.
Decision Trees	Lattice-Based Cryptography	Effective protection against data tampering.	Limited support for real-time data processing, impacting latency-sensitive IoT applications.
K-Nearest Neighbors	Elliptic Curve Cryptography	Improved authentication accuracy.	May suffer from reduced performance in scenarios with high levels of noise in communication channels.
Ensemble Methods	Post-Quantum Cryptography	Resilience against quantum attacks.	Increased computational demands on edge devices for implementing post-quantum cryptographic algorithms.
Convolutional Neural Networks	Lightweight Cryptography	Efficient image-based authentication.	Limited application in scenarios where detailed image analysis is required.
Reinforcement Learning	Code-Based Cryptography	Dynamic adaptation to evolving threats.	Complexity in tuning reinforcement learning parameters for optimal

			performance.
Clustering Algorithms	Identity-Based Encryption	Scalable key management for IoT devices.	Vulnerability to key exposure risks in identity-based encryption, impacting overall security.
Anomaly Detection Models	Homomorphic Encryption	Early detection of anomalous behavior in IoT.	Challenge in distinguishing between legitimate variations and malicious anomalies, leading to false positives.
Genetic Algorithms	Attribute-Based Encryption	Enhanced access control in Edge Computing.	Potential complexity in defining and managing attribute policies for secure access control.
Time Series Analysis	Fully Homomorphic Encryption	Improved privacy-preserving analytics.	Computational intensity associated with fully homomorphic encryption may limit its applicability in real-time analytics scenarios.

III. METHODOLOGY

Figure 2 shows a suggested multi-layer authentication framework for Mobile Edge Computing (MEC) systems. It aims to improve security by combining common login methods on different levels. The process starts when node X sends details about its name to the MEC server. The MEC server then asks the authentication center for upper-layer protocol authentication. This is the first layer of authentication. The authentication center replies with an authentication confirmation, which proves that the procedure is real.

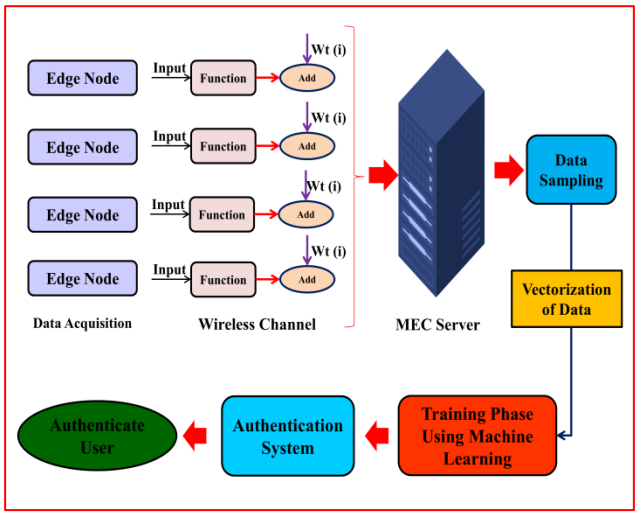

Figure 2: Proposed model User Authentication system

Once the upper-layer protocol authentication goes well, the MEC server moves on to the cable layer authentication. This step makes sure that the data coming from node X follows the rules for security. Once the uplink layer authentication is done, the last step of authentication takes place at the downlink, where node X performs physical layer authentication. This thorough layer access authentication method checks the communication entities' identities and reliability, creating a safe route for communication. Finally, the next layer of data validation makes the MEC system even safer. After access authentication is done between node X and the MEC server, the MEC server verifies every file it receives at the physical layer (PHY-layer). This process makes sure that the data is correct and that it hasn't been changed by hackers while it's being sent. In particular, the PHY-layer data verification is a key part of lowering the risk of data corruption and making sure that the information sent within the MEC structure can be trusted. The figure 2 shows an extra thing to think about in the case where edge nodes are limited in the amount of computing resources they can use. For these kinds of situations, the suggested structure lets the MEC server handle layer data validation. This method makes the best use of resources and makes sure that security steps, especially checking the accuracy of data, are always followed, even in edge nodes that don't have a lot of resources. In addition, the system handles a unique security issue that comes up with the Internet of Things (IoT). Figure 2 shows an example of an IoT mobile edge computing threat that can be stopped by authenticating data at the physical layer. By using multi-layer data verification, the suggested framework lowers the chance that data will be accessed, changed, or compromised without permission [15]. This protects the privacy and security of data in the IoT environment, which is always changing and linked. The multi-layer identification scheme in MEC systems provides a complete and flexible way to keep information safe. The proposed framework builds a strong defense against possible threats by adding upper-layer protocol, uplink layer, and downlink physical layer authentications one after the other, along with data integrity checks. It does this by taking into account both limited resources and specific attack scenarios that could happen in the IoT world.

IV. MACHINE LEARNING BASED AUTHENTICATION SYSTEM

A Machine Learning (ML) method for an identification system as DNN with advanced encryption is chosen based on a number of factors, such as the needs of the system, the type of data, and the performance qualities that are wanted. ML methods are good at different things, so picking the best one for the identification system means thinking about its specifics. Defining the key parts of the encryption process is part of creating a step-by-step mathematical model for Enhanced Encryption in Edge Computing using Deep Neural Networks (DNN). Let's look at a simple case where a DNN is used for encryption in an IoT and wireless network setting:

A. Deep Neural Networks for Encryption:

1. Input Data:

- Let X represents the input data, which could be a vector of raw sensor readings from IoT devices.

2. Feature Extraction using DNN:

- Employ a DNN for feature extraction. Let $f(X;\theta)$ denote the DNN model, where θ represents the model parameters. The output of the DNN is denoted as $H = f(X;\theta)$, capturing relevant features.

$$H = f(X;\theta)$$

3. Key Generation:

- Generate encryption keys based on the extracted features H. This can be represented as a key generation function :

$$K = g(H)$$

4. Encryption:

- Utilize the generated key K to encrypt the original data X. Let Y represent the encrypted data.

$$Y = Encrypt(X, K)$$

5. Transmission:

- Transmit the encrypted data Y over the wireless network or within the Edge Computing environment.

6. Decryption:

At the receiving end, use the same DNN-based feature extraction to obtain H' from the received encrypted data Y'.

$$H' = f(Y';\theta)$$

7. Key Extraction:

Extract the decryption key K' from the features H' using the inverse of the key generation function.

$$K' = g\text{^}(-1)(H')$$

8. Decryption:

Use the extracted key K' to decrypt the received data Y'.

$$X' = Decrypt(Y', K')$$

When cross entropy is used, the log probability function gives us the cost function J(ψ).

Let's derive the cost function J(ψ) using cross-entropy from the log likelihood function. Assuming a binary classification problem, the log likelihood function is defined as follows:

a. Hypothesis Function:

Start with the hypothesis function for logistic regression:

$$h\psi(x) = 1 / (1 + e^{-\psi^T x})$$

b. Likelihood Function:

Define the likelihood function for the entire dataset:

$$L(\psi) = \prod_{i=1}^{m} \left[h\psi(x^i)^{y^i} \cdot \left(1 - h\psi(x^i)\right)^{1-y^i} \right]$$

Where, m is the number of training examples, x^i is the feature vector for the i-th example, and y^i is the corresponding label.

c. Log Likelihood Function:

Take the natural logarithm (log) of the likelihood function to simplify computations:

$$\ell(\psi) = \sum_{i=1}^{m} \left[y^i \log\left(h\psi(x^i)\right) + (1 - y^i) \log\left(1 - h\psi(x^i)\right) \right]$$

d. Negative Log Likelihood:

Convert the log likelihood to a cost function by taking the negative log likelihood and averaging over all examples:

$$J(\psi) = -\left(\frac{1}{m}\right) \sum \left[y^i \log\left(h\psi(x^i)\right) + (1 - y^i) \log\left(1 - h\psi(x^i)\right) \right]$$

This is the final cost function $J(\psi)$ for logistic regression with cross-entropy as the loss function. The goal in training the logistic regression model is to minimize this cost function by adjusting the parameters ψ.

B. Gradient Descent for User Authentication

For multi-user verification, the pseudocode describes the Gradient Descent with Momentum (GDM) method. Initializes important factors such as learning rate and momentum, and then updates weights iteratively based on gradients and momentum terms that have been calculated. By looking at past slopes, this dynamic method speeds up convergence.

Algorithm:

$Initialize$:

$-$ $Learning\ rate\ \alpha$

$-$ $Momentum\ parameter\ \beta\ (usually\ between\ 0\ and\ 1)$

$-$ $Initial\ weights\ \psi$

$-$ $Momentum\ term\ v\ (initialize\ to\ zero\ vector)$

$Repeat\ until\ convergence$:

$for\ each\ training\ example\ (X, y)\ in\ the\ dataset$:

1. $Compute\ the\ gradient\ of\ the\ cost\ function\ with\ respect\ to\ the\ weights$:

$$\frac{\partial J(\psi)}{\partial \psi} = ComputeGradient(X, y, \psi)$$

2. $Update\ the\ momentum\ term$:

$$v = \beta * v + (1 - \beta) * \frac{\partial J(\psi)}{\partial \psi}$$

3. $Update\ the\ weights\ using\ the\ momentum\ term$:

$$\psi = \psi - \alpha * v$$

$Compute\ the\ cost\ function\ J(\psi)\ over\ the\ entire\ dataset$:

$$J(\psi) = ComputeCost(X_{dataset}, y_{dataset}, \psi)$$

$Check\ for\ convergence\ criteria\ (e.g., change\ in\ cost\ function,\ number\ of\ iterations, etc.)$.

C. RMS Optimization Authentication Algorithm

For Multi-User Authentication, a well-known optimization method called RMSprop is used to improve training speed and resolution. The program changes the learning rates for each parameter on the fly, which makes it easier for users whose gradients are different to solve problems. In this case, RMSprop adjusts to the specifics of each person, making the security model parameters work better. This helps to speed up convergence and improve the accuracy of authentication. The fact that the algorithm can change to different user patterns shows how important it is for making strong and adaptable security systems for a wide range of user settings.

1. Initialization:

- Learning rate α
- Exponential decay parameter β (typically close to 1, e.g., 0.9)
- Small constant ϵ (to avoid division by zero)
- Initial weights θ
- Initialize squared gradient accumulator E[g^2] to zero vector.

2. Repeat until convergence:

- For each training example (X_i, y_i) in the dataset:

2.1. Compute the gradient of the cost function with respect to the weights:

$$g = \nabla_\theta J(\theta; X_i, y_i)$$

2.2. Update the squared gradient accumulator:

$$E[g^2] = \beta * E[g^2] + (1 - \beta) * g^2$$

2.3. Update the weights using the RMSprop update rule:

$$\theta = \theta - \frac{\alpha}{\left(\sqrt{E[g^2] + \epsilon}\right) \odot g}$$

Compute the cost function over the entire dataset:

$$J(\theta) = \left(\frac{1}{m}\right) \sum_{i=1}^{m} J\,(\theta; X_i, y_i)$$

Check for convergence criteria (e.g., change in cost function, number of iterations, etc.). In the above equations, $J(\theta; X\_i, y\_i)$ represents the cost function for a single training example, and m is the number of training examples. The operator ⊙ denotes element-wise multiplication.

V. RESULT AND DISCUSSION

Key measures, such as the cost function (J) and the authentication rate (Pa), are used to judge how well deep learning-based multi-user authentication works. The cost function J, which is found in equation (6), shows how well the model can tell the difference between real emitters and spoofers. This measure is very important for figuring out how well the deep neural network (DNN) works. On the other hand, the identification rate (Pa) is the chance of correctly telling the difference between real users and fake ones. P is written in math terms as the ratio of the number of successful authentications to the total number of samples in a group.

$$Pa = \frac{1}{N\sum iNyiTi}, k$$

The power delays of the real emitter and the spoofer are shown in Figure 3. These delays are important for simulating wireless channels. For a standardized Doppler shift (fd) of 0.125, six lines with different power delays are picked to make channels. To make things more realistic, the first five tracks for both the real emitter and the spoofer are the same.

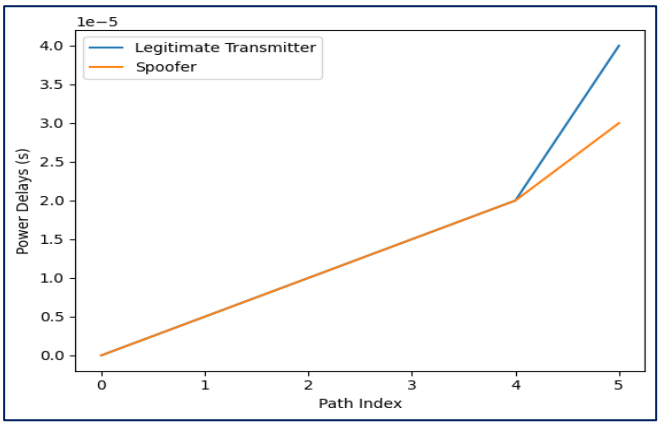

Figure 3: Power Delays of Legitimate Transmitter and Spoofer

They go through 0s, 5×10^6s, 1×10^5s, 1.5×10^5s, and 2×10^5s of delays. The sixth path splits, with delays of 4×10^5s for the real transmitter, 3×10^5s for the spoofer, 2.6×10^5s for the valid transmitter, and 2.2×10^5s for the spoofer. A sample period (tsampling) of 5×10^6s is used to keep the signal-to-noise ratio (SNR) at 4 dB. There are 256 subcarriers in the exercise, with pilot intervals of 15 kHz and a cycle prefix length (lcp_length) of 30. This setup makes sure that the behaviors of the wireless channel are fully modeled, giving useful information about the power delays of both real and fake channels when things are really like they are. Figure 4 shows how time complexity and convergence speed affect each other. This helps us understand how the multi-user authentication system works in terms of both computing efficiency and training dynamics. Time complexity, which is shown on the y-axis, shows how many computers are needed to finish one identification process.

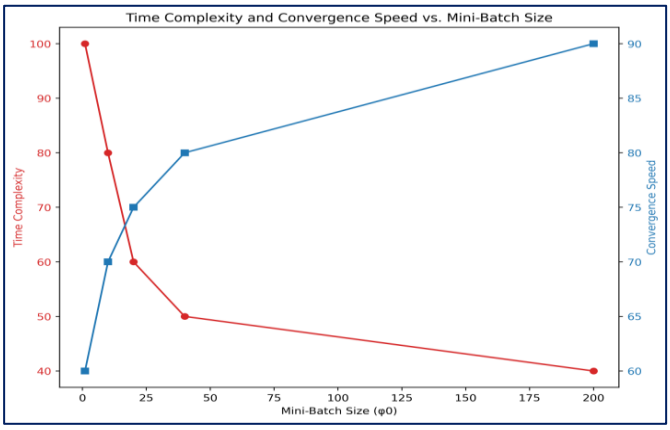

Figure 4: Time Complexity and Convergence Speed

The x-axis shows convergence speed, which shows how fast the system becomes stable during training. The graph shows how different mini-batch sizes (φ0) affect the amount of time needed and how fast the solution converges.

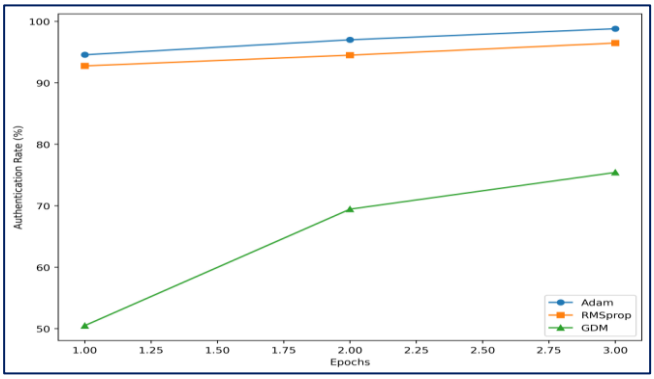

Figure 5: Authentication Rate vs Epochs for Different Optimizers

As φ0 goes up, floating-point operations take less time, which suggests that bigger mini-batch sizes might be better for computing. Notably, a mini-batch size of φ0=200 has the lowest time complexity, which suggests that resources are being used efficiently. However, the benefit of φ0=200 is not very strong when compared to smaller mini-batch sizes like φ0=10, φ0=20, and φ0=40. It's also clear from the graph that using stochastic gradient descent (φ0=1) takes the most time and slows down convergence the most. The epochs, which show how many full passes through the whole training dataset were made during the model training process, are shown on the x-axis. The identification rate, which shows how well the system can tell the difference between legal and rogue nodes, is shown on the y-axis. The figure 5 identifies rates of various optimization methods change over time and across different epochs. Notably, the Adam algorithm always does better than the others, showing a higher rate of confirmation. Adam's identification rate hits 97.0% after three epochs when a learning rate (±) of 5×10^-4 is used. The rates of identification for RMSprop are 92.75%, while the rates for GDM are 50.5%. It's helpful to know how the performance of optimization methods changes over time, and this graph helps with choosing the best way to train the multi-user login system.

VI. CONCLUSION

Incorporating machine learning and stronger protection for edge computing in IoT and wireless networks looks like a good way to make these connected systems safer and more efficient. The study of physical layer identification that takes advantage of different wireless channel features shows a proactive way to stop bad things from happening. Researchers have looked into new methods, such as deep learning algorithms, to make wireless networks safer. They have made big strides in identification and tracking. The comparison of optimization algorithms shows that Adam is better at getting higher recognition rates. This shows how important algorithm selection is to system performance. The difficulties of optimizing the mini-batch size show how difficult it is to find the right balance between speed of convergence and time complexity. A mini-batch size of 10 turns out to be the best solution. The computer complexity study also shows how the post-training identification process has been simplified, which shows that the system can be used in real life. As technology moves closer to the edge, it's more important than ever to keep the Internet of Things and wifi networks safe.

REFERENCES

[1] L. Ding and M. Ben Salem, "A Novel Architecture for Automatic Document Classification for Effective Security in Edge Computing Environments," 2018 IEEE/ACM Symposium on Edge Computing (SEC), Seattle, WA, USA, 2018, pp. 416-420

[2] Y. Li, H. Ling, X. Ren, C. Yu and T. Liu, "Privacy-Preserving Swarm Learning Based on Lightweight Homomorphic Encryption and Blockchain Technology," 2023 IEEE 5th Eurasia Conference on IOT, Communication and Engineering (ECICE), Yunlin, Taiwan, 2023, pp. 692-697

[3] A. R, K. Renuka D, O. S and S. R, "Secured Data Sharing of Medical Images for Disease diagnosis using Deep Learning Models and Federated Learning Framework," 2023 International Conference on Intelligent Systems for Communication, IoT and Security (ICISCoIS), Coimbatore, India, 2023, pp. 499-504

[4] X. Li, J. He, P. Vijayakumar, X. Zhang and V. Chang, "A Verifiable Privacy-Preserving Machine Learning Prediction Scheme for Edge-Enhanced HCPSs," in IEEE Transactions on Industrial Informatics, vol. 18, no. 8, pp. 5494-5503, Aug. 2022

[5] M. Nijim and H. Albataineh, "Secure-Stor: A Novel Hybrid Storage System Architecture to Enhance Security and Performance in Edge Computing," in IEEE Access, vol. 9, pp. 92446-92459, 2021.

[6] J. Sharma, D. Kim, A. Lee and D. Seo, "On Differential Privacy-Based Framework for Enhancing User Data Privacy in Mobile Edge Computing Environment," in IEEE Access, vol. 9, pp. 38107-38118, 2021.

[7] M. Yang, Y. He and J. Qiao, "Federated Learning-Based Privacy-Preserving and Security: Survey," 2021 Computing, Communications and IoT Applications (ComComAp), Shenzhen, China, 2021, pp. 312-317.

[8] Ajani, S. N. ., Khobragade, P. ., Dhone, M. ., Ganguly, B. ., Shelke, N. ., & Parati, N. . (2023). Advancements in Computing: Emerging Trends in Computational Science with Next-Generation Computing. International Journal of Intelligent Systems and Applications in Engineering, 12(7s), 546–559

[9] R. Dornala, S. Ponnapalli and K. Sai, "Blockchain Security in Edge Applications with Novel Load Balancing Approach," 2023 International Conference on Sustainable Communication Networks and Application (ICSCNA), Theni, India, 2023, pp. 263-269, doi: 10.1109/ICSCNA58489.2023.10370477.

[10] T. S. Vasista, R. P. Singh and P. Kumar, "Advancing Healthcare: Unleashing the Potential of Cryptography, Blockchain, and Machine Learning," 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), Delhi, India, 2023, pp. 1-6.

[11] G. Eibl, K. Bao, P.-W. Grassal, D. Bernau and H. Schmeck, "The influence of differential privacy on short term electric load forecasting", Energy Informat., vol. 1, pp. 48, Oct. 2018.

[12] R. Civino, C. Blondeau and M. Sala, "Differential attacks: Using alternative operations", Des. Codes Cryptogr., vol. 87, no. 2, pp. 225-247, Mar. 2019.
[13] A. R. Chowdhury, C. Wang, C. He, A. Machanajjhala and S. Jha, " Crypt \$epsilon\$: Crypto-assisted differential privacy on untrusted servers ", Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD), pp. 603-619, Jun. 2020.
[14] L. Ni, C. Li, X. Wang, H. Jiang and J. Yu, "DP-MCDBSCAN: Differential privacy preserving multi-core DBSCAN clustering for network user data", IEEE Access, vol. 6, pp. 21053-21063, 2018.
[15] Y. Zhang, Z. Hao and S. Wang, "A differential privacy support vector machine classifier based on dual variable perturbation", IEEE Access, vol. 7, pp. 98238-98251, 2019.