Journal of Network Intelligence ©2025 ISSN 2414-8105 (Online)
Taiwan Ubiquitous Information Volume **10**, Number **1**, February **2025**

Secure Data Aggregation Model for Cyberattack Prediction in Smart Grids Based on Clustering Federated Learning

Hongsheng Yang, Jianghua Liang*, Ming Li, Qiang Wang, Zhuojun Huang, Ying Zhang

Guangdong Zhongshan Power Supply Bureau of China Southern Power Grid, Zhongshan 528400, China.
yanghongsheng@zs.gd.csg.cn, 15289628536@163.com, liming@zs.gd.csg.cn
wangqiang@zs.gd.csg.cn, 1764124839@qq.com, zhangying@zs.gd.csg.cn

*Corresponding author: Jianghua Liang

Received July 3, 2024, revised October 18, 2024, accepted December 26, 2024.

Abstract. *With the deployment of smart grids, the Internet of Things (IoT) has gained substantial popularity. The integration and utilization of data from IoT monitoring systems in smart grids offer immense potential value. However, various entities store and manage this data, rendering the analysis of these distributed systems both privacy-sensitive and voluminous. Federated learning (FL) offers a promising solution by enabling the development of a shared deep-learning model while maintaining local data privacy within each monitoring system. However, real-world FL scenarios face significant challenges due to the imbalance in training data distribution, both in category and quantity. To align with our research focus, this study uses cyberattack data as a representative example. Different IoT monitoring systems may record varying types and quantities of cyberattacks, leading to divergent optimization directions for local models. Consequently, creating an effective FL framework to learn from these imbalanced distributed data to detect cyberattacks becomes a complex task. In this paper, we propose an enhanced FL framework that incorporates local dataset clustering to improve performance. The distributed local datasets are clustered before conducting a classical FL, with homomorphic encryption employed to protect the privacy of the local datasets during clustering. To evaluate the proposed framework, we utilized the NSL-KDD dataset from the Network Security Laboratory, including all major IoT computing attacks. Experimental results demonstrated that the proposed framework achieves fewer communication rounds than traditional FL models by 5.88% overall.*

Keywords: Federated Learning; Secure Aggregation; Smart Grid; Cyberattack Prediction

1. Introduction.

Aggregating and utilizing data from smart grids has become an increasingly prevalent trend in recent years [1, 2]. Real-time power usage data represent a vital public resource that should be extensively leveraged for business and governmental decision-making. One of the most critical applications of data aggregation in smart grids is the integration of operational data from various monitoring systems to detect potential cyberattacks [3, 4]. However, these data are closely tied to sensitive information, raising significant privacy concerns. Meanwhile, as the Internet of Things (IoT) technology is widely utilized in smart grids [5], the security of IoT systems has garnered significant critical attention [6, 7]. Numerous cyberattacks, such as Distributed Denial of Service (DDoS) and man-in-the-middle attacks, pose significant threats to both service providers and end users within IoT networks [8]. Consequently, anomaly detection-based techniques and systems have been developed and integrated into IoT infrastructures [9]. In this work, we focus on the topic of cyberattack detection in smart grids through secure data aggregation. This approach enables the examination of schemes for securely aggregating data in distributed smart grid systems and provides an efficient method to address the challenge of cyberattack detection in IoT networks. Several key features are crucial for cyberattack detection in IoT networks: firstly, IoT devices generate an enormous amount of data; secondly, IoT monitoring systems are inherently decentralized; and thirdly, the size of IoT networks is continually expanding [10, 11]. The feature of a substantial volume of data presents valuable opportunities for utilizing deep learning models [12, 13]. Therefore, centralized deep learning schemes can be employed to detect cyberattacks. However, these techniques face significant challenges related to data privacy, as they require the transfer of local data from each monitoring system to a centralized third-party server for training. Additionally, centralized deep learning becomes impractical when dealing with large datasets distributed across multiple locations [14]. To address the challenge of large, distributed datasets, federated learning (FL) methods have been employed to distribute the deep learning workload across multiple clients. The FL framework inherently enhances security and privacy compared to centralized learning frameworks, as data generated by end clients remains local. This approach allows the useful data to train the learning model locally on each client in a distributed manner. Only the updated parameters, rather than the raw data, are exchanged between the end clients and the cloud server [15]. While FL enables parallel computation of models for distributed cyberattack detection, it does not adequately address the practical challenges posed by data heterogeneity [16]. This heterogeneity refers to the non-independent and identically distributed (non-IID) nature of the data [17]. Different IoT monitoring systems may capture diverse types and quantities of cyberattacks, resulting in divergent optimization directions for local models. In this paper, we propose a secure data aggregation model for cyberattack prediction in smart grids, utilizing clustering federated learning. Our approach employs a heterogeneity-aware clustering algorithm integrated with homomorphic encryption to securely categorize the distributed datasets. Subsequently, we apply the widely recognized average federated learning model to enhance prediction accuracy and maintain data privacy [16]. Specifically, our contributions are summarized as follows:

- The proposed enhanced federated learning model offers a secure method for aggregating and utilizing distributed data from smart grids. This approach enables more robust predictions for cyberattacks within the IoT network of smart grids, thereby improving overall security and efficiency.
- By employing the proposed approach, we can utilize cyberattack data from smart grids to make accurate predictions while also integrating other data from distributed systems for deep learning. This model effectively addresses the challenges of secure data aggregation and data heterogeneity in federated learning, providing a comprehensive solution for enhancing IoT network security in smart grids.
- Compared to the well-known average federated learning model, which is trained directly on distributed datasets for cyberattack prediction, the proposed model demonstrates faster convergence.

The remaining sections are organized as follows. In Section 2, we review related research on secure data aggregation and cyberattack prediction. Section 3 provides an overview of the proposed federate learning framework and describes the clustering algorithm in detail. Section 4 gives the evaluation of the proposed model compared with the baseline model. Finally, Section 5 offers a summary of the proposed system.

2. **Related work.** In smart grids, secure data aggregation methods have been developed to efficiently gather data while ensuring the preservation of data privacy [18, 19, 20]. These methods usually employ a cryptography scheme to encrypt and aggregate the data. In the end, the aggregated data can be used by the authorized entities. However, traditional secure data aggregation methods encounter significant challenges when applied to cyberattack detection in IoT networks of smart grids due to the increased size and distribution of the data. Aggregating these distributed datasets incurs substantial costs. As distributed datasets continue to grow in size, a new paradigm known as federated learning has emerged [16, 21, 14]. Federated learning enables the training of a global deep-learning model to predict cyberattacks in smart grids while preserving the local storage of datasets. Although federated learning has become promising in this topic, key challenges caused by the heterogeneity still have to be addressed for practical application [22]. Although federated learning shows great promise in this domain, key challenges posed by data heterogeneity still need to be addressed for practical application. Consequently, research aimed at enhancing federated learning methods to manage heterogeneity, which includes significant variability in data categorization and non-identically distributed data, has garnered considerable interest.

For cyberattack detection using traditional deep learning methods, Song et al. [23] treated network attacks as malicious traffic and utilized an integrated algorithm combining LSTM and XGBoost for malicious traffic classification. Despite its innovative approach, the detection accuracy was relatively low. By employing the SRS activation function at the top layer of the model, it achieved better generalization capability and learning speed, with a recognition rate of 97%, though its detection precision was relatively low. Yue et al. [24] proposed an integrated detection algorithm based on CNN and RNN for DoS attacks, achieving an accuracy rate of 99.1%. However, this method incurs significant computational overhead. Yu et al. [25] developed a high-precision intrusion detection system based on a multi-scale convolutional neural network (MSCNN). Using convolutional kernels of different scales effectively improved the detection precision, increasing the average accuracy rate for various attacks. Wu et al. [26] introduced the LuNet model, which achieved a high detection rate for Probe and DoS attacks but showed lower recognition rates.

For cyberattack detection using federated learning methods, Nguyen et al. [27] proposed the Dïot system, which can be deployed in IoT environments for automated intrusion detection after being trained via federated learning. Liu et al. [28] designed a federated learning-based intrusion detection system framework that can be used for anomaly detection in temporal attack data. Li et al. [29] presented a federated learning-based intrusion detection system for industrial environments, demonstrating improved communication efficiency. Wang et al. [30] proposed a network anomaly detection method based on deep neural networks and federated learning. This method utilizes federated learning to train models using dispersed local device data within the Internet of Things (IoT), sharing only the trained weight parameters with a central server. By deploying detection models across different regions, this approach enhances model accuracy and reduces false positive rates compared to other deep detection models. However, the method only employs the federated averaging algorithm (Fed Avg) for model aggregation, without considering model performance and communication efficiency. Rahman et al. [31] presented an intrusion detection model that trains local models using federated learning with local data, thereby protecting data privacy. Experiments on the NSLKDD dataset showed that, after the final round of federated learning, the accuracy of the aggregated model fluctuated around 83.09%. Nevertheless, this method has several issues: prolonged response times for model uploading and updating during training, poor performance of selected client detection models, and low detection accuracy. Sawsan et al. [32] addressed the heterogeneity of devices and limited computational and communication resources in IoT by proposing FedMCCS. This approach considers factors such as CPU, storage, and time for each client device to predict their participation in the next training round. Experiments demonstrated that FedMCCS could train the maximum number of local models per communication round, but it did not optimize client detection model performance and selection methods to reduce communication overhead.

3. Proposed Scheme.

3.1. Overview. As is shown in Figure 1, the entire framework consists of four key parts:

- Secure Data Transmission: Each client employs a homomorphic encryption scheme to securely transmit the summary information of its local dataset to the server..
- Similarity Calculation: The server calculates a Jaccard similarity matrix [33] based on the received summary information.
- Client Clustering: Utilizing the Jaccard similarity matrix, the server divides the clients into several groups, ensuring that clients with more similar records are placed in the same group.
- Federated Model Training: Each group then trains a federated learning model, leveraging the grouped data to enhance prediction accuracy and maintain data privacy.

We will describe these parts in detail next.

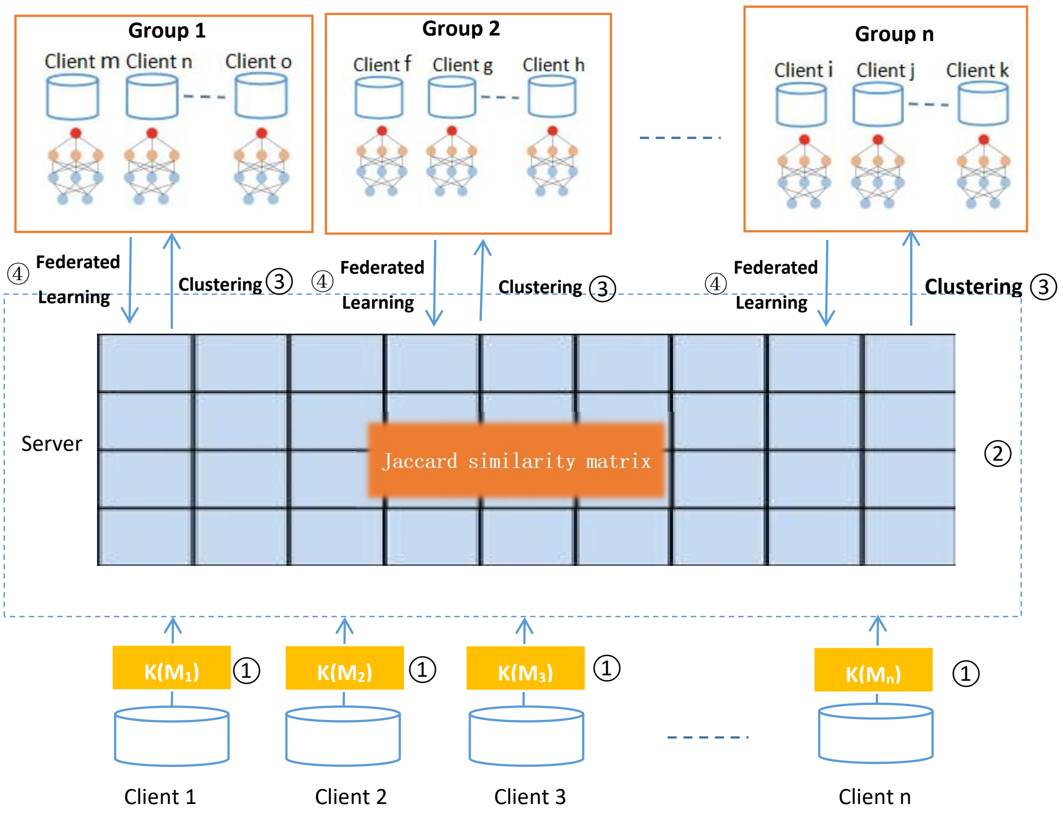

FIGURE 1. The Proposed Framework

3.2. **Dataset.** The proposed framework is based on the premise that training federated models with datasets containing more similar records can achieve higher performance compared to those with less similarity. To validate this hypothesis, we divide the overall dataset into several local subsets, ensuring that similar types of records are grouped together as much as possible. For instance, in an extreme scenario, the dataset with 40 types of records is divided into 10 local subsets, each containing 4 distinct types of records, with no overlap of record types between different clients. Thus, a DDoS attack record, for example, will be found in only one client dataset. We will train the federated model on this extreme scenario and compare the performance with the same model which is trained on the randomly split local datasets. The result can be found in the Section 4.

3.3. **Secure Data Transmission.** After validating the aforementioned hypothesis, we proceed by clustering local datasets with higher similarity before initiating the training process within the proposed framework. Here, similarity is defined by the proportion of records of the same type; local datasets containing a greater number of identical record types are considered to be more similar. Therefore, to calculate the similarity among different local datasets, we should get the information of how many records for each type in a local dataset. To protect the privacy of each client, we employ a homomorphic encryption scheme [34] in the proposed framework. Homomorphic encryption is a form of encryption that allows computations to be carried out on ciphertexts, so that when decrypted, the result matches the outcome of operations performed on the plaintext. This means that data can be encrypted and processed without ever exposing the plaintext, ensuring privacy and security. With this scheme, the local information can be transmitted to a server securely. We sets up a context for CKKS (Cheon-Kim-Kim-Song) scheme [34] , which is a type of leveled fully homomorphic encryption. The principles of homomorphic encryption are as follows:

1. Encryption: The summary information in each client is encrypted using a public key, resulting in ciphertexts. In the CKKS scheme, this involves encoding real or complex numbers into polynomials and then encrypting these polynomials. The formula is as follows:

$$\text{Enc}(m) = ([m] + \Delta \cdot r) \mod q$$

where $[m]$ is the encoding of the information m, Δ is a scaling factor, r is a random polynomial, and q is the coefficient modulus.

2. Computation: Operations such as addition and multiplication are performed directly on the ciphertexts. The CKKS scheme allows for these operations to be performed approximately due to its encoding and encryption methods.

3. Decryption : The modified ciphertexts are decrypted using a private key, returning the result of the computation on the original plaintext data. The formula is as follows:

$$\text{Dec}(c) = \left(\frac{c \mod q}{\Delta} \right)$$

where c is the ciphertext.

As is shown in Algrithm 1, each client uses the encryption to calculate the numbers of each type of record in its local dataset and send the encrypted vector to the server. For each local dataset, Line 1 of the algorithm calculates the total count of each record type, where “TargetColumn” indicates the specific type of record. Then the frequency of each record type is encrypted by the homomorphic encryption, resulting in an encrypted frequency vector of each record type.

Algorithm 1 Calculate Local Information with Homomorphic Encryption

```
Require: $datasets$
1: $frequency \leftarrow dataset.$TargetColumn.ValueCounts
2: $frequency\_vector \leftarrow$ Array([$frequency$.get(cla, 0) for cla in range(num_classes)])
3: $encrypted\_vector \leftarrow$ ckks_tensor(context, [str(value) for value in $frequency\_vector$])

4: return $encrypted\_vector$
```

3.4. **Similarity Calculation.** After getting the encrypted frequency vectors from the local clients, the server tries to calculate a similarity matrix called Jaccard Similarity Matrix [33]. The Jaccard similarity matrix, also known as the Jaccard coefficient, is a statistic used for gauging the similarity and diversity of sample sets. It measures similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets. Given two sets A and B, the Jaccard similarity matrix is given by:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

As is shown in Algrithm 2, the Jaccard similarity matrix is being calculated for a number of datasets, where each dataset is represented by a frequency vector of encrypted attack types. The intersection of two sets is represented by the element-wise multiplication of their frequency vectors. This is the equivalent of counting the number of elements that are common between the two sets. The union of two sets is represented by the sum of the element-wise multiplication of each set with itself, minus the intersection. This is equivalent to counting all the unique elements in both sets. We perform these operations using the homomorphic computation, which allows the operations to be performed on encrypted data without decrypting it first. This is particularly useful for preserving privacy when dealing with sensitive data.

Algorithm 2 Calculate Intersection and Union

```
Require: $encrypted\_vectors$, $num\_clients$
1: $intersection\_vectors = []$
2: $union\_vectors = []$
3: for $i \leftarrow 0$ to $num\_clients - 1$ do
4:     for $j \leftarrow i + 1$ to $num\_clients$ do
5:         $intersection \leftarrow encrypted\_vectors[i] * encrypted\_vectors[j]$
6:         $union \leftarrow encrypted\_vectors[i] * encrypted\_vectors[i] + encrypted\_vectors[j] * encrypted\_vectors[j] - intersection$
7:         $intersection\_vectors \leftarrow intersection$
8:         $union\_vectors \leftarrow union$
9:     end for
10: end for
11: return ($intersection\_vectors$, $union\_vectors$)
```

Upon completing the computation of the intersection and union vectors, the server transmits these values to each client. As illustrated in Algorithm 3, each client decrypts these values and calculates the Jaccard similarity by dividing the sum of the decrypted intersection by the sum of the decrypted union. Consequently, all clients obtain the same Jaccard similarity matrix, which they then return to the server.

Algorithm 3 Calculate Jaccard Similarity Matrix

```
Require: $intersection\_vectors$, $union\_vectors$
 1: $jaccard\_matrix \leftarrow$ Zeros($num\_clients$, $num\_clients$)
 2: $k = 0$
 3: for $i \leftarrow 0$ to $num\_clients - 1$ do
 4:    for $j \leftarrow i + 1$ to $num\_clients$ do
 5:       $jaccard \leftarrow$ sum(decrypt($intersection[k]$)) / sum(decrypt($union[k]$))
 6:       $jaccard\_matrix[i, j] \leftarrow jaccard$
 7:       $jaccard\_matrix[j, i] \leftarrow jaccard$
 8:    end for
 9: end for
10: return ($jaccard\_matrix$)
```

3.5. **Client Clustering.** Upon getting the Jaccard similarity matrix, the server uses hierarchical clustering to group datasets into clusters based on their similarity. Hierarchical clustering [35] is a method of cluster analysis which seeks to build a hierarchy of clusters. Initially, each dataset is considered as a separate cluster. The Algorithm 4 then repeatedly executes the following steps:

- Identify the two clusters that are closest together based on the complete linkage criterion.
- Merge these two clusters into a single cluster.
- This process continues until the specified number of clusters ($num\_cluster$) is reached.

The result, stored in the variable "clusters", is an array of cluster labels. Each dataset is assigned a label corresponding to the cluster it belongs to. By using the Jaccard similarity matrix as input, the clustering is based on the similarity between datasets, with more similar datasets being grouped together. This approach is particularly useful when dealing with complex data structures where the relationships between data points are not easily quantifiable in a traditional Euclidean space.

Algorithm 4 Hierarchical Clustering

```
Require: $jaccard\_matrix, num\_cluster$
 1: $clustering \leftarrow$ AgglomerativeClustering($n\_clusters = num\_cluster$)
 2: $clusters \leftarrow clustering$.fit_predict($jaccard\_matrix$)
 3: $groups \leftarrow$ list of empty lists with size $num\_cluster$
 4: for $cluster\_id \leftarrow 0$ to $num\_cluster - 1$ do
 5:    for $dataset\_idx \leftarrow 0$ to len($clusters$) $- 1$ do
 6:       if $clusters[dataset\_idx] = cluster\_id$ then
 7:          $groups[cluster\_id].append(dataset\_idx)$
 8:       end if
 9:    end for
10: end for
11: return $groups$
```

3.6. **Federated Model Training.** In this part, the framework executes federated model training [16], where each client in the same group trains a local model on its own data and then sends the model updates to a central server. The server aggregates these updates to improve the global model. The "ClientUpdate" function represents the local training process on the client's data. The global model parameters are updated by taking a weighted average of the local model parameters, where the weights are proportional to the number of data points each client has.

As is shown in Algorithm 5 executed by the server and Algorithm 6 executed by the clients, the server initializes the global model parameters, denoted as θ_0, which will be shared with all clients in the same group. For a predefined number of rounds T, each client k receives the current global model parameters θ_t from the server. The client performs local updates to the model parameters based on its own data, resulting in updated parameters θ^k_{t+1}. This is done through the "ClientUpdate" function, which typically involves training for several local epochs E using stochastic gradient descent (SGD) [36] or a similar optimization method.

The "ClientUpdate" function takes the current global parameters θ and performs updates based on the client's local dataset. For each batch b, the parameters are updated by subtracting the product of the learning rate η and the gradient of the loss function $\nabla\mathcal{L}(\theta; b)$. After all selected clients in a group have completed their local updates, the server aggregates these updates to form the new global model parameters θ_{t+1}. This is done by taking a weighted average of the updated parameters from each client, where the weights are proportional to the number of data points n_k that each client contributes to the total data points n. The aggregated parameters θ_{t+1} become the new global model parameters for the next round of training or for final evaluation. This iterative process allows for the creation of a robust global model that benefits from the diverse data across all clients without requiring the actual data to be centralized, thus preserving privacy and reducing communication costs.

Algorithm 5 Federated Averaging (FedAvg)

```
Require: A set of $m$ clients in a group
1: Initialize global model parameters $\theta_0$
2: for each round $t = 1, 2, \dots, T$ do
3:     for each client $k$ do
4:         $\theta^k_{t+1} \leftarrow \text{ClientUpdate}(k, \theta_t)$
5:     end for
6:     $\theta_{t+1} \leftarrow \sum_{k=1}^{m} \frac{n_k}{n} \theta^k_{t+1}$
7: end for
```

Algorithm 6 ClientUpdate(k, θ)

```
Require: Client index $k$, current global model parameters $\theta$
1: Updated local model parameters $\theta'$
2: Initialize local model with global parameters $\theta$
3: for each local epoch $i = 1, 2, \dots, E$ do
4:     for each batch $b$ in local dataset of client $k$ do
5:         $\theta' \leftarrow \theta - \eta\nabla\mathcal{L}(\theta; b)$
6:     end for
7: end for
8: return $\theta'$
```

4. **Experiments and Analysis.**

4.1. **Experimental Environment.** The experimental environment for this study was meticulously designed to ensure availability and optimal performance for the proposed framework. The hardware setup included a high-performance computing server equipped with the following specifications:

- Processor: Intel Xeon Silver 4310, 48 cores, 2.10 GHz
- Memory: 256 GB DDR4 RAM
- Graphics Processing Unit (GPU): Two NVIDIA A30 GPUs, each with 24 GB memory

To evaluate the proposed framework, we utilized the NSL-KDD dataset [37], a benchmark in the field of cyberattack detection system assessment. This dataset comprises $148,517$ records, with more than 48% representing cyberattack instances. Each record contains 42 features and 1 identification indicating the classification of the record. There are 40 types of the records, including 1 normal type of record and 39 types of attack records. Then we randomly split this dataset into several shares and take each share as a local dataset for a client. Ultimately, this process yields multiple local datasets, each varying in size and containing different quantities and types of attack records. All experiments were conducted using a uniform random seed to ensure consistency across different runs. We use LSTM [38] as the test model in the federated learning. The learning rate was set to 0.001 and the hidden units are 64. The software environment was configured as follows:

- Operating System: Linux login.cluster.cn 3.10.0-1160.el7.x86_64
- Deep Learning Framework: PyTorch 2.3.0+cu121
- Python Version: 3.10.14, with necessary libraries such as NumPy, Tenseal, and Pandas for data preprocessing and homomorphic encryption.

4.2. **Results Analysis.** Firstly, we need to verify our hypothesis that training federated models with datasets containing more similar records can achieve higher performance compared to those with less similarity. To test this, we conducted two experiments. In the first experiment, we divided the entire dataset into 10 local subsets, each containing 4 distinct types of records with no overlap of record types between different clients. This scenario represents the optimal case for our proposed framework. In the second experiment, we randomly divided the dataset into 10 subsets, each representing a different client, to simulate a typical real-world scenario. The performance outcomes of these two cases are illustrated in Figure 2. The results demonstrate that the baseline case, with more homogeneous record types within each client, achieves faster convergence compared to the random division, thereby supporting our hypothesis.

After validating our hypothesis, we proceeded to compare the performance of our proposed framework, referred to as “fedCluster” with the average learning model “fedAvg” as described by McMahan et al. [16]. This comparison was carried out across four different scenarios, each involving a distinct set of local clients. As is shown in Figures 3 to 6, the proposed framework shows progress. In scenarios involving 10 or 25 clients, the proposed framework exhibited only marginally better performance than the ”fedAvg” model. However, for configurations with 15 or 20 clients, the proposed framework showed a significant improvement in performance. This enhances our hypothesis that local datasets containing more similar records can achieve higher performance in federated learning. And the number of clients is not a key factor for the convergence performance.

Overall, the proposed framework exhibits superior performance compared to ”fedAvg.” It’s crucial to note that a randomly divided local dataset might inherently contain enough similar records, potentially limiting the advantage of ”fedCluster.” The degree to which

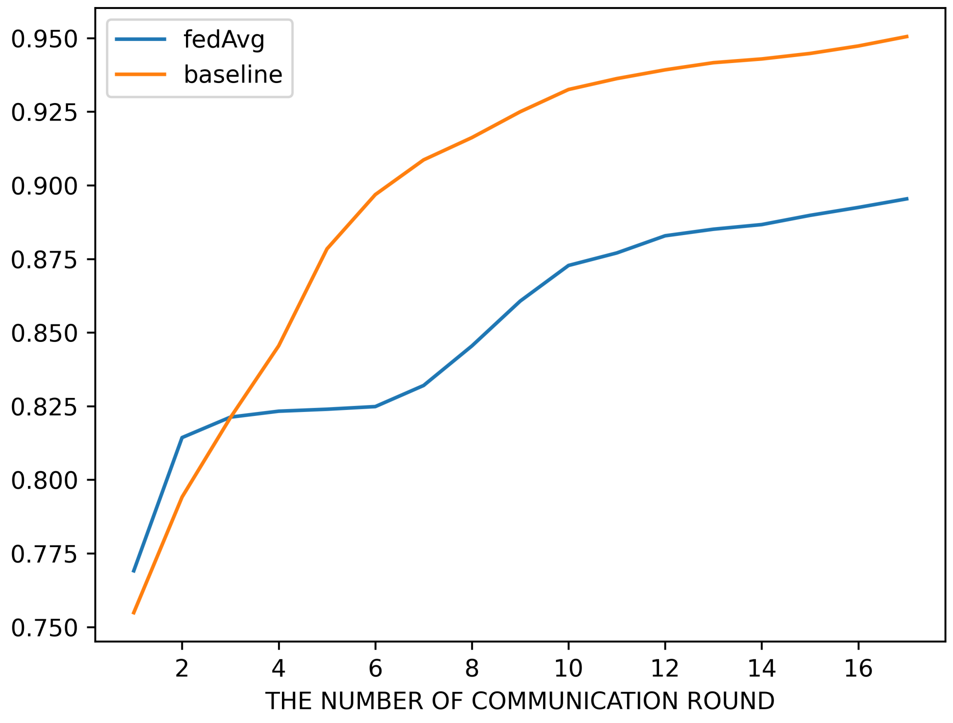

Figure 2. Hypothesis Verification

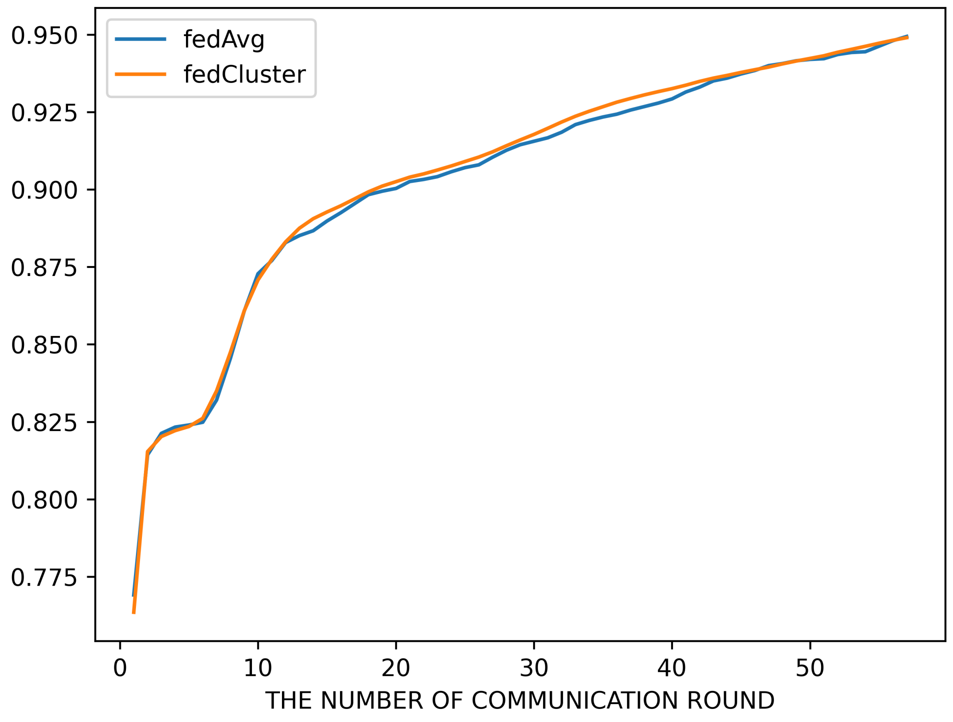

Figure 3. Comparison in the Case of 10 Clients. Slight Improvement.

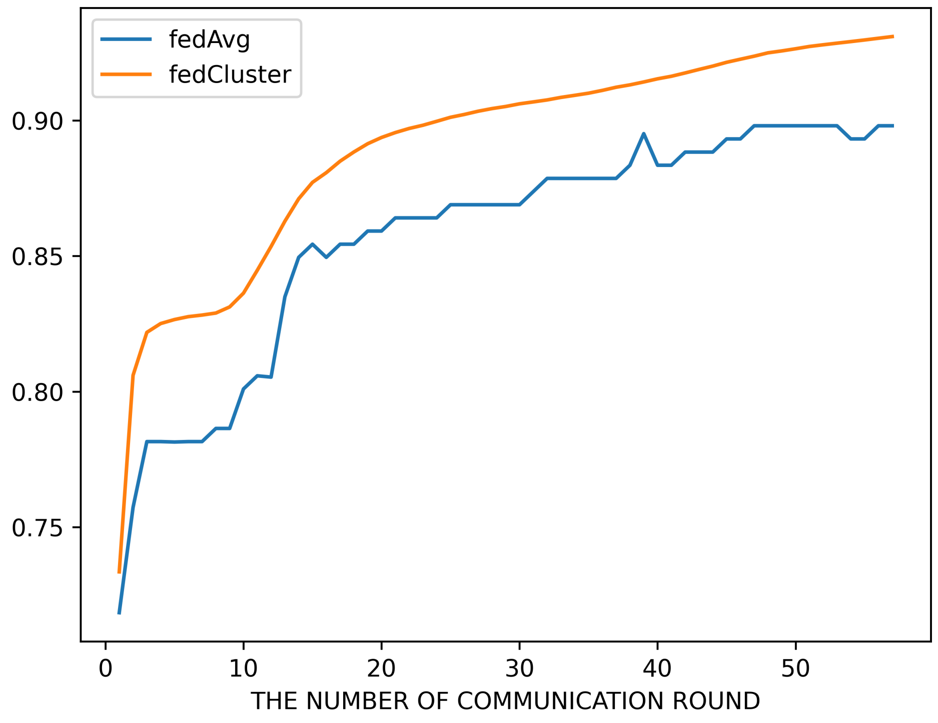

Figure 4. Comparison in the Case of 15 Clients. Faster Convergence and Higher Overall Accuracy.

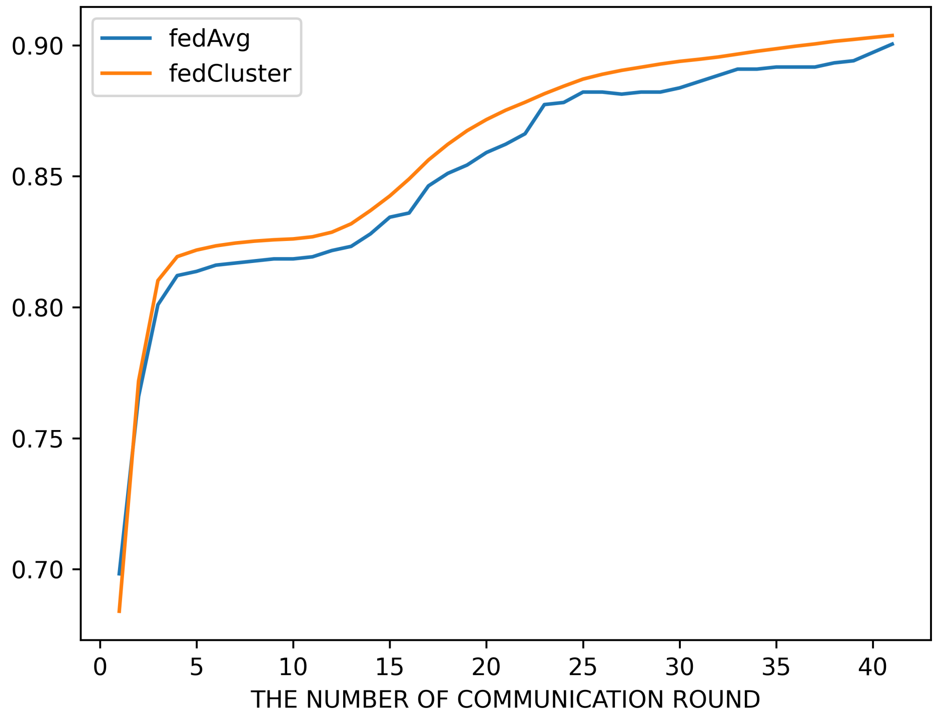

Figure 5. Comparison in the Case of 20 Clients. Faster Convergence and Higher Overall Accuracy.

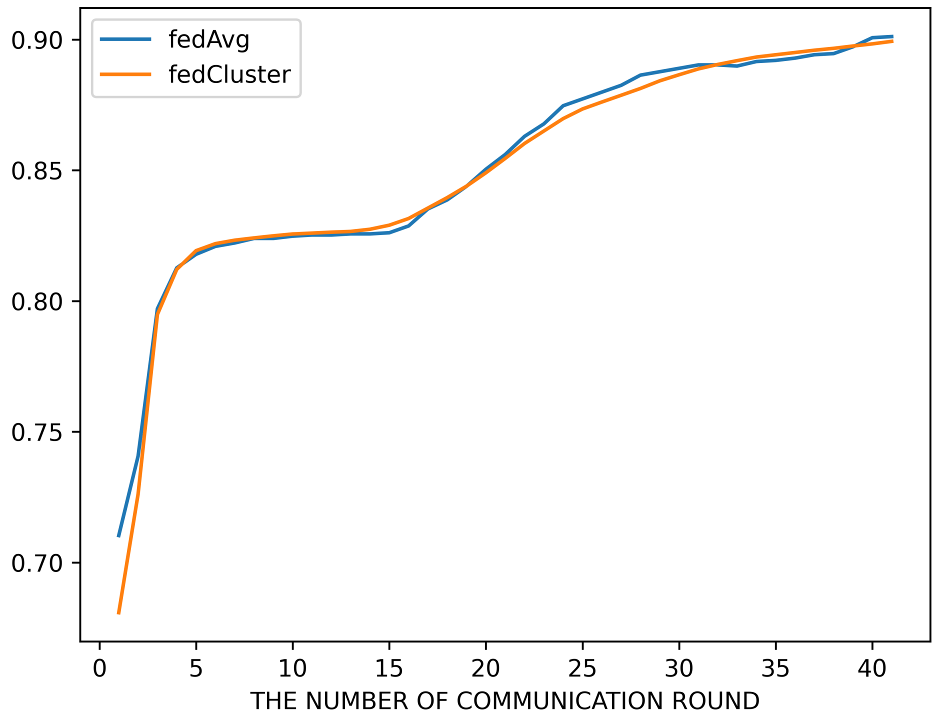

FIGURE 6. Comparison in the Case of 25 Clients. Slight Improvement.

"fedCluster" outperforms "fedAvg" is significantly influenced by the level of dispersion among the divided datasets.

4.3. **Comparison.** To facilitate comparison, we selected the same deep learning model (LSTM) with identical batch sizes and epochs across all federated learning frameworks. The scenario where the number of clients equals 0 represents the baseline case, which serves as the optimal scenario within the proposed framework. We conducted a comparative analysis based on the number of communication rounds required between the server and clients to achieve specific target accuracies. Table 1 presents the total number of communication rounds needed for the model accuracy to exceed 85%, while Table 2 illustrates the number of communication rounds required for the accuracy to surpass 95%. The last columns of the tables summarize the percentage improvement of "fedCluster" over "fedAvg" for communication rounds. The results clearly indicate that the proposed framework achieves the target accuracies with fewer communication rounds, demonstrating its efficiency in the experimental setup.

5. **Conclusion.** The integration of operational data from various monitoring systems is critical for detecting potential cyberattacks in smart grids, yet this process must navigate significant privacy concerns due to the sensitivity of the data involved. In this work, we address the complex challenge of cyberattack detection in smart grids through a secure data aggregation approach, leveraging federated learning (FL) to distribute the deep learning workload across multiple clients while maintaining data privacy. The proposed enhanced FL model incorporates a heterogeneity-aware clustering algorithm and homomorphic encryption to securely categorize distributed datasets, followed by the application of the average federated learning model. The proposed model provides a secure and efficient

Table 1. COMPARISON WITH %85 ACCURACY

Framework	**Clients**	**Batch Size**	**Epoch**	**Comm. Rounds**	**Improvement (%)**
Baseline	0	16	5	50	N/A
fedCluster	10	16	5	90	0
fedAvg	10	16	5	90	N/A
fedCluster	15	16	5	180	20
fedAvg	15	16	5	225	N/A
fedCluster	20	16	5	340	5
fedAvg	20	16	5	360	N/A
fedCluster	25	16	5	500	4.76
fedAvg	25	16	5	525	N/A

Table 2. COMPARISON WITH %95 ACCURACY

Framework	**Clients**	**Batch Size**	**Epoch**	**Comm. Rounds**	**Improvement (%)**
Baseline	0	16	5	170	N/A
fedCluster	10	16	5	580	1.69
fedAvg	10	16	5	590	N/A
fedCluster	15	16	5	1335	13.87
fedAvg	15	16	5	1500	N/A
fedCluster	20	16	5	2280	1.72
fedAvg	20	16	5	2320	N/A
fedCluster	25	16	5	3550	0
fedAvg	25	16	5	3550	N/A

method for aggregating and utilizing distributed data from smart grids, resulting in improved security and predictive accuracy. Additionally, the model shows faster convergence compared to traditional federated learning models, highlighting its practical advantages in real-world applications. This research offers a comprehensive solution to enhance IoT network security within smart grids, addressing both the technical and privacy-related challenges inherent in distributed data environments.

In the future, several areas can be built on the findings of this study. For example, investigating the scalability of the proposed framework in larger and more complex smart grid environments would be beneficial. This includes testing with a greater number of clients and more varied datasets to evaluate performance and security. Another promising avenue is the development of adaptive clustering techniques. An adaptive online clustering method can analyze data distribution and dynamically update cluster centroids to reflect real-time changes in the data stream. This enables the formation of flexible clusters and the assignment of data based not only on inherent features but also on its distribution at specific moments. By leveraging such adaptive techniques, which adjust to the evolving nature of cyber threats and the increasing diversity of data, the robustness and flexibility of federated learning models can be significantly enhanced.

Acknowledgment. This research was supported by the Guangdong Provincial Key Laboratory of Power System Network Security (No.GPKLPSNS-2023-KF-04).

REFERENCES

[1] J. Song, Y. Liu, J. Shao, and C. Tang, "A dynamic membership data aggregation (dmda) protocol for smart grid," *IEEE Systems Journal*, vol. 14, no. 1, pp. 900–908, 2019.

[2] A. Saleem, A. Khan, S. U. R. Malik, H. Pervaiz, H. Malik, M. Alam, and A. Jindal, "Fesda: Fog-enabled secure data aggregation in smart grid iot network," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6132–6142, 2019.

[3] Y. Yilmaz and S. Uludag, "Mitigating iot-based cyberattacks on the smart grid," in *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2017, pp. 517–522.

[4] Y. Yılmaz and S. Uludag, "Timely detection and mitigation of iot-based cyberattacks in the smart grid," *Journal of the Franklin Institute*, vol. 358, no. 1, pp. 172–192, 2021.

[5] D. Bandyopadhyay and J. Sen, "Internet of things: Applications and challenges in technology and standardization," *Wireless personal communications*, vol. 58, pp. 49–69, 2011.

[6] T.-Y. Wu, F. Kong, Q. Meng, S. Kumari, and C.-M. Chen, "Rotating behind security: an enhanced authentication protocol for iot-enabled devices in distributed cloud computing architecture," *EURASIP Journal on Wireless Communications and Networking*, vol. 2023, no. 1, p. 36, 2023.

[7] T.-Y. Wu, L. Wang, and C.-M. Chen, "Enhancing the security: A lightweight authentication and key agreement protocol for smart medical services in the ioht," *Mathematics*, vol. 11, no. 17, p. 3701, 2023.

[8] Z. Bakhshi, A. Balador, and J. Mustafa, "Industrial iot security threats and concerns by considering cisco and microsoft iot reference models," in *2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*. IEEE, 2018, pp. 173–178.

[9] X.-X. Lin, P. Lin, and E.-H. Yeh, "Anomaly detection/prediction for the internet of things: State of the art and the future," *IEEE Network*, vol. 35, no. 1, pp. 212–218, 2020.

[10] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE communications surveys & tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[11] (2017) Number of connected iot devices will surge to 125 billion by 2030. Accessed: 2024-06-12. [Online]. Available: https://sst.semiconductor-digest.com/2017/10/number-of-connected-iot-devices-will-surge-to-125-billion-by-2030/

[12] Y. Zhou, M. Han, L. Liu, J. S. He, and Y. Wang, "Deep learning approach for cyberattack detection," in *IEEE INFOCOM 2018-IEEE conference on computer communications workshops (INFOCOM WKSHPS)*. IEEE, 2018, pp. 262–267.

[13] K. Dushyant, G. Muskan, Annu, A. Gupta, and S. Pramanik, "Utilizing machine learning and deep learning in cybesecurity: An innovative approach," *Cyber Security and Digital Forensics*, pp. 271–293, 2022.

[14] B. Ghimire and D. B. Rawat, "Recent advances on federated learning for cybersecurity and cybersecurity for federated learning for internet of things," *IEEE Internet of Things Journal*, vol. 9, no. 11, pp. 8229–8249, 2022.

[15] M. Alazab, S. P. RM, M. Parimala, P. K. R. Maddikunta, T. R. Gadekallu, and Q.-V. Pham, "Federated learning for cybersecurity: Concepts, challenges, and future directions," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 5, pp. 3501–3509, 2021.

[16] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[17] C. Yang, M. Xu, Q. Wang, Z. Chen, K. Huang, Y. Ma, K. Bian, G. Huang, Y. Liu, X. Jin *et al.*, "Flash: Heterogeneity-aware federated learning at scale," *IEEE Transactions on Mobile Computing*, 2022.

[18] R. Lu, X. Liang, X. Li, X. Lin, and X. Shen, "Eppa: An efficient and privacy-preserving aggregation scheme for secure smart grid communications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1621–1631, 2012.

[19] T. W. Chim, S.-M. Yiu, V. O. Li, L. C. Hui, and J. Zhong, "Prga: Privacy-preserving recording & gateway-assisted authentication of power usage information for smart grid," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 1, pp. 85–97, 2014.

[20] Y. Liu, W. Guo, C.-I. Fan, L. Chang, and C. Cheng, "A practical privacy-preserving data aggregation (3pda) scheme for smart grid," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 3, pp. 1767–1774, 2018.

[21] T. T. Huong, T. P. Bac, D. M. Long, T. D. Luong, N. M. Dan, B. D. Thang, K. P. Tran *et al.*, "Detecting cyberattacks using anomaly detection in industrial control systems: A federated learning approach," *Computers in Industry*, vol. 132, p. 103509, 2021.

[22] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[23] C. Song, Y. Sun, G. Han, and J. J. Rodrigues, "Intrusion detection based on hybrid classifiers for smart grid," *Computers & Electrical Engineering*, vol. 93, p. 107212, 2021.

[24] C. Yue, L. Wang, D. Wang, R. Duo, and X. Nie, "An ensemble intrusion detection method for train ethernet consist network based on cnn and rnn," *IEEE Access*, vol. 9, pp. 59 527–59 539, 2021.

[25] J. Yu, X. Ye, and H. Li, "A high precision intrusion detection system for network security communication based on multi-scale convolutional neural network," *Future Generation Computer Systems*, vol. 129, pp. 399–406, 2022.

[26] P. Wu and H. Guo, "Lunet: a deep neural network for network intrusion detection," in *2019 IEEE symposium series on computational intelligence (SSCI)*. IEEE, 2019, pp. 617–624.

[27] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "Dïot: A federated self-learning anomaly detection system for iot," in *2019 IEEE 39th International conference on distributed computing systems (ICDCS)*. IEEE, 2019, pp. 756–767.

[28] Y. Liu, S. Garg, J. Nie, Y. Zhang, Z. Xiong, J. Kang, and M. S. Hossain, "Deep anomaly detection for time-series data in industrial iot: A communication-efficient on-device federated learning approach," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6348–6358, 2020.

[29] B. Li, Y. Wu, J. Song, R. Lu, T. Li, and L. Zhao, "Deepfed: Federated deep learning for intrusion detection in industrial cyber–physical systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5615–5624, 2020.

[30] X. Wang, Y. Wang, Z. Javaheri, L. Almutairi, N. Moghadamnejad, and O. S. Younes, "Federated deep learning for anomaly detection in the internet of things," *Computers and Electrical Engineering*, vol. 108, p. 108651, 2023.

[31] S. A. Rahman, H. Tout, C. Talhi, and A. Mourad, "Internet of things intrusion detection: Centralized, on-device, or federated learning?" *IEEE Network*, vol. 34, no. 6, pp. 310–317, 2020.

[32] S. AbdulRahman, H. Tout, A. Mourad, and C. Talhi, "Fedmccs: Multicriteria client selection model for optimal iot federated learning," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4723–4735, 2020.

[33] J. Irani, N. Pise, and M. Phatak, "Clustering techniques and the similarity measures used in clustering: A survey," *International journal of computer applications*, vol. 134, no. 7, pp. 9–14, 2016.

[34] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23*. Springer, 2017, pp. 409–437.

[35] F. Nielsen and F. Nielsen, "Hierarchical clustering," *Introduction to HPC with MPI for Data Science*, pp. 195–211, 2016.

[36] S.-i. Amari, "Backpropagation and stochastic gradient descent method," *Neurocomputing*, vol. 5, no. 4-5, pp. 185–196, 1993.

[37] M. Hassan, "Nsl-kdd dataset," https://www.kaggle.com/datasets/hassan06/nslkdd, 2020, accessed: 2024-03-25.

[38] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: Lstm cells and network architectures," *Neural computation*, vol. 31, no. 7, pp. 1235–1270, 2019.