SF-AE: Split Federated Autoencoder for Unsupervised IoT Intrusion Detection

Proceedings

Accepted version

Augello, A. De Paola, D. Giosuè, G. Lo Re.

In Proceedings of the 2025 International Congress on Information and Communication Technology (ICICT)

DOI: https://doi.org/10.1007/978-981-96-6432-0_28

Publisher: Springer Nature Singapore

SF-AE: Split Federated Autoencoder for Unsupervised IoT Intrusion Detection

Andrea Augello, Alessandra De Paola, Domenico Giosuè, and Giuseppe Lo Re

University of Palermo, Italy
{andrea.augello01,alessandra.depaola,domenico.giosue,giuseppe.lore}@unipa.it

Abstract. Smart systems have become increasingly popular in recent years, widening the attack surface of cyber threats. Machine learning algorithms have been successfully integrated into modern security mechanisms to detect such attacks. Internet of Things (IoT) systems often have limited computational resources and are unable to execute entire machine learning pipelines. However, these systems often produce and manage sensitive data. Thus, it is preferable to avoid exposing their data to external analysis, e.g., on cloud systems. This work introduces SF-AE: a novel architecture that enables the distributed training of an anomaly-based intrusion detection system on devices with limited computational resources without exposing sensitive data. Experimental results on multiple datasets show that SF-AE outperforms state-of-the-art methods in terms of attack detection performance, at lower computation and communication costs for the participating devices.

Keywords: Network-level security and protection; Distributed Machine learning; Anomaly Detection; Internet of Things; Federated Learning.

1 Introduction

Smart devices are becoming increasingly integrated into our daily lives, and are expected to become even more pervasive in the future [Jejdling et al., 2024]. As technology advances, the growing scale and complexity of cyberattacks have caused traditional security measures to become insufficient, highlighting the need for new strategies to address emerging vulnerabilities effectively [Adat and Gupta, 2018]. IoT devices, in particular are especially prone to cyberattacks due to their limited computational capabilities [Fazzolari et al., 2023] which make them unable to run complex security mechanisms. The increasing attractiveness of Internet of Things (IoT) devices as targets for cyberattacks is evidenced by industry reports documenting how the number of malware attacks targeting IoT devices increases by over 60% each year [Kaspersky, 2022]. Thus, it is essential to develop security mechanisms that can detect and mitigate these attacks.

Although IoT devices do not usually have sufficient computational capabilities to locally train and execute complex machine learning models, relying on external servers to perform the training is unacceptable. Doing so would require the devices to upload their sensitive data to the cloud, which is a significant privacy concern:

the traffic generated by IoT devices is highly sensitive and can be exploited to infer information about the users, such as their habits, preferences, and physical location [Zou et al., 2023]. Collaborative learning paradigms, such as federated learning [McMahan et al., 2017] and split learning [Gupta and Raskar, 2018], are thus a promising approach to enable resource-constrained devices with sensitive data to benefit from the power of machine learning algorithms without directly exposing their data to external analysis [Alazab et al., 2021].

To address these pressing challenges, this work introduces SF-AE: a distributed learning architecture that enables the training of an autoencoder-based intrusion detection system on IoT devices with limited computational resources. The main contributions of this work are:

- A lightweight autoencoder-based intrusion detection system that can be trained on IoT devices with limited computational resources.
- A collaborative distributed learning architecture for unsupervised learning that lowers communication and computational costs on IoT devices compared to traditional split and federated learning.

The effectiveness of SF-AE is assessed through extensive experiments on the Bot IoT [Koroniotis et al., 2019] and TON IoT [Alsaedi et al., 2020] datasets, showing that it outperforms state-of-the-art methods in terms of attack detection performance. The remainder of this paper is organized as follows: Section 2 reviews the related works, Section 3 describes the SF-AE architecture, Section 4 presents the experimental evaluation, and Section 5 concludes the paper.

2 Related Works

Artificial intelligence, in particular machine learning, has become an essential tool in the fight against cyberattacks, with a wide range of applications in intrusion detection, malware analysis, and threat intelligence [Imran et al., 2023]. However, the sensitive nature of cybersecurity data, such as network traffic logs, and the difficulty of collecting labeled data make it challenging to apply traditional centralized machine learning approaches, necessitating the development of novel resilient and privacy-preserving collaborative distributed learning mechanisms [Idrissi et al., 2023]. This section provides an overview of existing intrusion detection approaches and distributed learning mechanisms that have been proposed to address these training challenges.

2.1 Anomaly-based intrusion detection

An intrusion detection system (IDS) is a security mechanism that monitors network traffic for malicious activities and alerts the system administrator when an attack is detected. Typically, intrusion detection systems can be classified into two categories: signature-based and anomaly-based. Signature-based IDSs have high accuracy in detecting known attacks, but usually rely on closed-world assumptions that make them ineffective against zero-day attacks not available in the dataset used for their creation, requiring frequent updates to maintain

their effectiveness [Zoppi et al., 2023]. By contrast, anomaly-based IDS compare the current behavior of the system with a model of normal behavior, and alert the administrator when the current behavior deviates significantly from the expected behavior. Anomaly-based IDS are well-suited for being trained through unsupervised learning approaches by only using normal traffic data, which is often more readily available than labeled data.

Many neural network architectures have been proposed for anomaly-based intrusion detection, including recurrent neural networks [Augello et al., 2024b] and transformers [Sana et al., 2024], but autoencoders are the most popular deep learning model for unsupervised anomaly-based intrusion detection. Autoencoders are neural networks trained to project the input data into a usually lower-dimensional latent space and then reconstruct the input from the latent representation. The dimensionality reduction forces the autoencoder to learn a lossy compressed representation of the input patterns. Therefore, when presented with data that is significantly different from the training data, autoencoders will on average have a higher reconstruction error than when presented with data that is similar to the training data, making them suitable for detecting anomalous behavior [Gong et al., 2019]. Autoencoder-based IDSs have been shown to be particularly effective in detecting anomalous traffic and extracting meaningful features from network traffic data [Meidan et al., 2018].

2.2 Distributed collaborative learning

In recent years, distributed collaborative learning has gained popularity to train machine learning models on large, geographically distributed, datasets without requiring to upload the data to a central, possibly untrusted, server. Two main key motivations can be identified for the development of these approaches: data privacy and computational resource constraints. The data privacy concerns stem from regulations, such as the General Data Protection Regulation (GDPR) [Truong et al., 2021], and the sensitive nature of the data being such that it is not desirable to share it with third parties. The computational resource constraints are particularly relevant in the context of IoT devices, which often have limited computational capabilities and are unable to run complex machine learning algorithms, requiring the support of a more powerful server to train and execute the model.

Federated learning Federated Learning (FL) [McMahan et al., 2017], illustrated in Fig. 1a, is a distributed learning approach that enables multiple clients to collaboratively train a shared model without sharing their data with each other. In federated learning, each client trains a copy of the model on its local data for a set number of epochs and then uploads the model to a central server. The federated server aggregates the models weights, typically through averaging, and redistributes the updated weights to the clients for further training. This iterative process continues for multiple communication rounds until convergence. Since the data remains on the clients' devices, FL well-suited to scenarios where

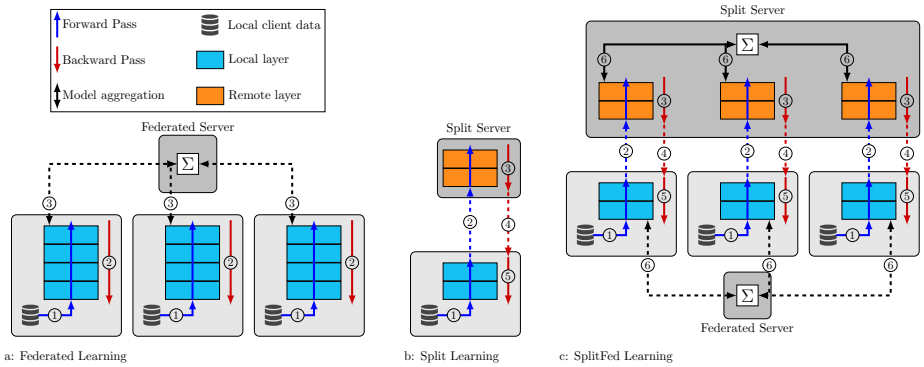

Fig. 1: Comparison of different distributed learning approaches.

data privacy is a concern. However, the lack of direct access to the data poses challenges in detecting clients that provide malicious data, which may degrade the performance of the shared model [Bhagoji et al., 2019] or otherwise compromise the learning process [Augello et al., 2024a]. Thus, it is a fundamental challenge to ensure that robust and reliable mechanisms are in place to detect and mitigate the impact of such malicious clients.

Split learning Complementary to the federated learning approach, where each client is assumed to have sufficient computational resources to train a model, Split Learning (SL) [Gupta and Raskar, 2018] is a distributed learning approach to train deep learning models on edge devices with limited computational resources by offloading most of the layers of the model to a server with more computational power. In SL, as depicted in Fig. 1b, a single client possesses training data, which is assumed to be sufficient to train a deep learning model, and the layers of the model are split between the client and the server, with the client only training the first few layers. For each training sample, the client performs the forward pass up to the layer where the split occurs, and the intermediate representation given by the split layer is sent to the server. The split server then completes the forward pass and backpropagates the gradient up to the split layer, updating the model parameters, and transmitting the accumulated gradients to the client so that it can compute the gradients for the remaining layers. If the labels also need to be kept private, the last layers of the model can also be stored on the client. This training process is mathematically equivalent to training the model in a centralized manner, but with the computational load distributed between the client and the split server. Since only the intermediate representation is shared between the client and the server, the original data is not directly exposed to the split server.

Federated Split learning Merging the federated and split learning approaches, SplitFed learning [Thapa et al., 2022] is a distributed learning approach that

combines the advantages of both federated and split learning. All the SplitFed clients train copies of the same model in parallel through split learning, with some layers on the client, and other layers on the server, and after a set number of epochs, the federated server aggregates the layers on the clients to create a global model and redistributes it to the clients. Two variants of SplitFed exist: SFLV1 and SFLV2. In SFLV1, for each client the split server maintains a separate copy of the split model, and these copies are also periodically averaged to create a global model. SFLV2 removes the server-side aggregation step: the split server maintains a single copy of the model, and the forward/backward pass is performed on the same model, randomizing the order the clients are processed. Fig. 1c shows the SFLV1 SplitFed learning variant.

3 SF-AE Architecture

In order to detect malicious traffic in IoT systems, this work presents a distributed learning architecture that enables the training of an autoencoder-based intrusion detection system on IoT devices with limited computational resources. The autoencoder is trained on the clients' logs of network traffic data corresponding to normal behavior. The autoencoder will, on average, be able to reconstruct the traffic generated by normal behavior of the system with a lower error than the anomalous behavior which was not seen during training, allowing the detection of attacks through the reconstruction error.

The SF-AE architecture, shown in Fig. 2, comprises three types of entities: the clients, a split server, and a federated server. The clients are IoT device with private datasets containing traffic data and limited computational capabilities, while the servers have less stringent computational constraints but are not trusted with direct access to the data. The split server is tasked with training the split part of the model. The federated server is responsible for coordinating the training process of the clients and computing the shared client model. The deep autoencoder model θ is split into three parts:

- θ_e: an encoder that maps the input data to a latent representation, located on the client side;
- θ_{S}: the split part of the model, which is located on the split server side and is responsible for reconstructing the latent representation;
- θ_d: a decoder that reconstructs the input data from the latent representation, also located on the client side.

The three sets of layers correspond to the original autoencoder model so that $\theta(\cdot) = \theta_d(\theta_{\mathrm{S}}(\theta_e(\cdot)))$. The layers belonging to θ_d are located on the client side because, since the autoencoder is trained to reconstruct the input data, computing the reconstruction error on the server side would require sharing the input data, which should be kept private. The training process is divided into two main steps: the federated training of the local autoencoder $\theta_C(\cdot) = \theta_d(\theta_e(\cdot))$ on the clients and the training of the split part of the model on the split server.

The SF-AE training process is divided in the five steps. The steps 1–3 are iterated for a T communication rounds, steps 5 and 6 are performed once.

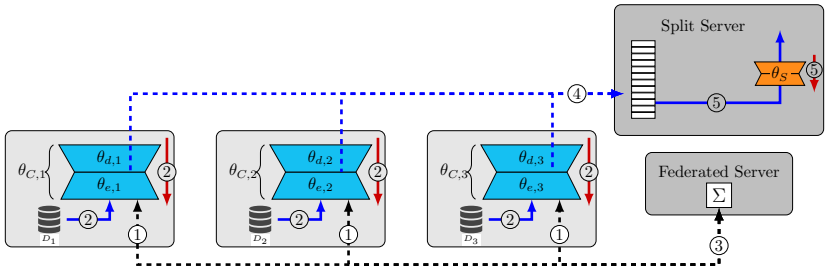

Fig. 2: The SF-AE architecture overview: each client obtains a small autoencoder from the federated server (1), and trains it on their local data (2). The federated server combines the models to create a global client model θ_C (3). The latent representation of the local datasets are shared with the split server (4), which uses them to train the split part of the model (5).

1. At a given iteration t, each client $i \in \{1, \ldots, N\}$ downloads the weights for θ_C^t from the federated server and sets its local copies of the model as $\theta_{C,i}^t = \theta_C^t$.
2. Using its local dataset D_i, each client i updates its local autoencoder model $\theta_{C,i}$ by solving the following optimization problem through e epochs of stochastic gradient descent:

$$\arg\min_{\theta_{C,i}^{t+1}} \frac{1}{|D_i|} \sum_{x \in D_i} \mathcal{L}(x, \theta_{C,i}^{t+1}(x)), \tag{1}$$

   with $\mathcal{L}$ being the reconstruction loss of the input data, e.g., the mean squared error.
3. After e local training epochs, the clients upload their model parameters to the federated server, which averages them to create the shared models θ_C through model averaging as in the FedAvg algorithm [McMahan et al., 2017]:

$$\theta_C^{t+1} = \frac{1}{N} \sum_{i=1}^{N} \theta_{C,i}^{t+1}. \tag{2}$$

   Steps 1–3 are repeated for T communication rounds, after which the clients have finished training the local autoencoder.
4. Each client separates their copy of the autoencoder in the encoder and decoder parts, θ_e and θ_d, respectively. The encoder θ_e is used by the clients to generate the latent representations of their local data, $\theta_e(D_i)$, which are then uploaded to the split server.
5. The split server trains the split part of the model θ_S to minimize the reconstruction error on the latent representations of the clients' data:

$$\arg\min_{\theta_S} \frac{1}{N} \sum_{i=1}^{N} \frac{1}{|D_i|} \sum_{x \in D_i} \mathcal{L}(\theta_e(x), \theta_S(\theta_e(x))). \tag{3}$$

This optimization problem is solved through multiple epochs of stochastic gradient descent.

At inference time, given a data point x, the clients use their local encoder to generate a latent representation of the input data $\theta_e(x)$, which is then uploaded to the split server. The split server uses its autoencoder to process the received data and transmits its output $\theta_S(\theta_e(x))$ back to the client. The client then utilizes $\theta_S(\theta_e(x))$ instead of $\theta_e(x)$ as input for reconstruction to the local decoder, obtaining $\hat{x} = \theta_d(\theta_S(\theta_e(x)))$. The output of the local decoder $\hat{x}$ is then compared to the original data x. If the reconstruction error is above a threshold τ, the input data is classified as anomalous, otherwise it is classified as normal.

4 Experimental Evaluation

In order to assess the effectiveness of SF-AE, a series of experiments were conducted on the publicly available Bot IoT [Koroniotis et al., 2019] and TON IoT [Alsaedi et al., 2020] datasets. The Bot IoT dataset contains realistic network traffic data from a simulated IoT environment. It includes both normal traffic and attacks typically associated with botnets. These attacks belong to categories such as service scanning, OS fingerprinting, Denial of Service (DoS), Distributed DoS (DDoS), keylogging, and data theft. The TON IoT dataset contains network traffic logs from an Industrial IoT testbed with normal traffic and attacks belonging to the scanning, DoS, DDoS, man in the middle, ransomware, backdoor, data injection, cross site scripting, and password cracking categories. In these experiments, 70% of the data was used for training and the remaining 30% for testing. 50 clients were used in the experiments, each with an equal amount of data. The following hyperparameters were used in the experiments: $e = 5$, $T = 20$, learning rate of 0.0001, and a batch size of 32.

In order to evaluate the effectiveness of the SF-AE architecture to differentiate regular traffic from attacks, the Receiver Operating Characteristic curve is used with different values for the threshold τ on the reconstruction error. The results for the two datasets at various concentrations of infected clients are reported in Fig. 3. The reported curves show that the model is extremely effective: even with stringent false positive rates as low as 5%, the model can correctly flag 87% of the attacks in the Bot IoT dataset and 81% in the TON IoT dataset.

The performance of SF-AE is also compared in terms of F1–score to the model trained through other distributed learning strategies: SFLV1, SFLV2, and FL. Additionally, a centralized setting is considered as a baseline. For the centralized approach, all the client data is uploaded to the server, which trains the model using the entire dataset. The results of this analysis varying the concentration of infected clients are shown in Table 1.

In this analysis, we also consider what happens when some of the clients are infected during training. This occurrence is particularly relevant in the context of IoT systems, where the devices are often exposed to the internet and can be infected by malware, and the FL paradigm does not allow to examine their data directly. The model performance is expected to worsen as the concentration of

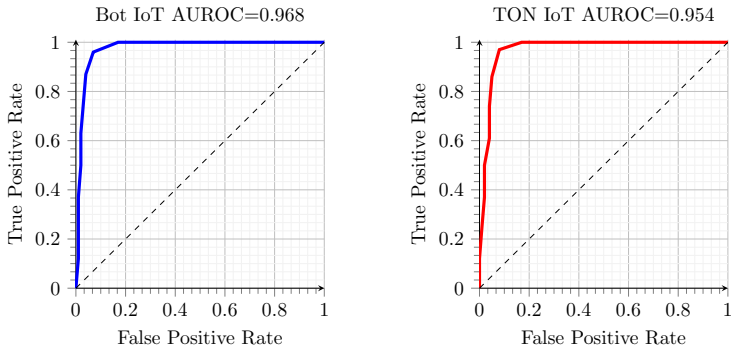

Fig. 3: SF-AE obtains excellent performance in detecting attacks on both datasets.

Table 1: F1–scores for the different distributed training strategies on the Bot IoT and TON IoT datasets at various concentrations of infected clients. The best performing method is highlighted in bold, while the second best is underlined.

Dataset	Infected %	Algorithm				
		SF-AE	SFLV1	SFLV2	FL	Centralized
Bot IoT	0	**0.968**	0.827	0.697	0.876	<u>0.951</u>
	10	**0.912**	0.892	0.768	<u>0.892</u>	0.891
	20	**0.907**	0.832	0.768	0.849	<u>0.865</u>
TON IoT	0	<u>0.954</u>	0.955	0.875	0.915	**0.983**
	10	<u>0.943</u>	0.926	0.869	0.903	**0.969**
	20	**0.928**	0.915	0.875	0.881	<u>0.917</u>

infected clients increases, as the model is trained on data that is not representative of the normal behavior of the system.

When no clients are infected, SF-AE is competitive with all the other methods as only the centralized approach outperforms it in the TON IoT dataset, and none can outperform it in the Bot IoT dataset. Thus, the reduced communication overhead does not come at the cost of detection performance. It is worth noting that, in general, SFLV2 appears to be the worst performing method at low concentrations of infected clients and, in the Bot IoT dataset, there is a sizable gap between the performance of SF-AE and centralized learning with other methods. As the concentration of infected clients increases, the traditional approaches are less able to discriminate normal traffic from attacks, dropping by up to 0.09 in the F1–score, while SF-AE maintains a high performance, with a maximum drop of 0.06 in the Bot IoT dataset. and 0.026 in the TON IoT dataset. These results show that SF-AE is able to maintain a high detection performance even when a significant portion of the clients are infected, outperforming the other distributed learning strategies, demonstrating the feasibility of the proposed architecture for training intrusion detection systems on IoT devices.

5 Conclusions

Unsupervised intrusion detection systems are an essential component of modern security mechanisms, especially in the context of IoT systems. However, the limited computational resources of IoT devices make it challenging to train such systems on these devices. The SF-AE architecture introduced in this work addresses this challenge by enabling the distributed training of an autoencoder-based intrusion detection system on IoT devices with limited computational resources through a split federated collaborative learning approach. The experimental evaluation demonstrated that SF-AE outperforms state-of-the-art methods in terms of attack detection performance, maintaining good detection capabilities even when some of the participants are already infected when the model is being trained. Future work will focus on further improving the performance of SF-AE by exploring alternative training strategies to reduce the communication and computational costs, as well as extending the evaluation to other datasets and scenarios.

Acknowledgments: This work was partially supported by the AMELIS project, within the project FAIR (PE0000013), and by the ADELE project, within the project SERICS (PE00000014), both under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU

References

[Adat and Gupta, 2018] Adat, V. and Gupta, B. B. (2018). Security in internet of things: issues, challenges, taxonomy, and architecture. *Telecommunication Systems*, 67:423–441.

[Alazab et al., 2021] Alazab, M., RM, S. P., Parimala, M., Maddikunta, P. K. R., Gadekallu, T. R., and Pham, Q.-V. (2021). Federated learning for cybersecurity: Concepts, challenges, and future directions. *IEEE Transactions on Industrial Informatics*, 18(5):3501–3509.

[Alsaedi et al., 2020] Alsaedi, A., Moustafa, N., Tari, Z., Mahmood, A., and Anwar, A. (2020). TON_IoT telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems. *Ieee Access*, 8:165130–165150.

[Augello et al., 2024a] Augello, A., Gupta, A., Lo Re, G., and Das, S. K. (2024a). Tackling selfish clients in federated learning. In *27th European Conference on Artificial Intelligence (ECAI 2024)*, pages 1888–1895. IOS Press.

[Augello et al., 2024b] Augello, A., Lo Re, G., Peri, D., and Thiyagalingam, P. (2024b). NEP-IDS: a network intrusion detection system based on entropy prediction error. In *2024 IEEE 49th Conference on Local Computer Networks (LCN)*, pages 1–9.

[Bhagoji et al., 2019] Bhagoji, A. N., Chakraborty, S., Mittal, P., and Calo, S. (2019). Analyzing federated learning through an adversarial lens. In *International conference on machine learning*, pages 634–643. PMLR.

[Fazzolari et al., 2023] Fazzolari, M., Ducange, P., and Marcelloni, F. (2023). An explainable intrusion detection system for IoT networks. In *2023 IEEE International Conference on Fuzzy Systems (FUZZ)*, pages 1–6. IEEE.

[Gong et al., 2019] Gong, D., Liu, L., Le, V., Saha, B., Mansour, M. R., Venkatesh, S., and Hengel, A. v. d. (2019). Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1705–1714.

[Gupta and Raskar, 2018] Gupta, O. and Raskar, R. (2018). Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116:1–8.

[Idrissi et al., 2023] Idrissi, M. J., Alami, H., El Mahdaouy, A., El Mekki, A., Oualil, S., Yartaoui, Z., and Berrada, I. (2023). Fed-ANIDS: Federated learning for anomaly-based network intrusion detection systems. *Expert Systems with Applications*, 234:121000.

[Imran et al., 2023] Imran, M., Siddiqui, H. U. R., Raza, A., Raza, M. A., Rustam, F., and Ashraf, I. (2023). A performance overview of machine learning-based defense strategies for advanced persistent threats in industrial control systems. *Computers & Security*, 134:103445.

[Jejdling et al., 2024] Jejdling, F., Davis, S., Barnes, T., Batra, R., Blennerud, G., Englund, L., Ericsson, E., Hammar, H., Husseini, A., Jansson, C., Kazi, S., Koleilat, A., Lindberg, P., Linder, P., Ludwig, R., Malizia, M., Murphy, M., and Shekhar Pandey, R. (2024). Ericsson mobility report june 2024.

[Kaspersky, 2022] Kaspersky (2022). Pushing the limits: How to address specific cybersecurity demands and protect iot. Online.

[Koroniotis et al., 2019] Koroniotis, N., Moustafa, N., Sitnikova, E., and Turnbull, B. (2019). Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset. *Future Generation Computer Systems*, 100:779–796.

[McMahan et al., 2017] McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR.

[Meidan et al., 2018] Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Shabtai, A., Breitenbacher, D., and Elovici, Y. (2018). N-baiot—network-based detection of iot botnet attacks using deep autoencoders. *IEEE Pervasive Computing*, 17(3):12–22.

[Sana et al., 2024] Sana, L., Nazir, M. M., Yang, J., Hussain, L., Chen, Y.-L., Ku, C. S., Alatiyyah, M., and Por, L. Y. (2024). Securing the iot cyber environment: Enhancing intrusion anomaly detection with vision transformers. *IEEE Access*.

[Thapa et al., 2022] Thapa, C., Arachchige, P. C. M., Camtepe, S., and Sun, L. (2022). Splitfed: When federated learning meets split learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8485–8493.

[Truong et al., 2021] Truong, N., Sun, K., Wang, S., Guitton, F., and Guo, Y. (2021). Privacy preservation in federated learning: An insightful survey from the GDPR perspective. *Computers & Security*, 110:102402.

[Zoppi et al., 2023] Zoppi, T., Ceccarelli, A., Puccetti, T., and Bondavalli, A. (2023). Which algorithm can detect unknown attacks? comparison of supervised, unsupervised and meta-learning algorithms for intrusion detection. *Computers & Security*, 127:103107.

[Zou et al., 2023] Zou, Q., Li, Q., Li, R., Huang, Y., Tyson, G., Xiao, J., and Jiang, Y. (2023). IoTBeholder: A privacy snooping attack on user habitual behaviors from smart home wi-fi traffic. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 7(1):1–26.