# Adaptive Federated Learning Defense Against Byzantine Attacks and Concept Drift in IIoT

Assem Alhelou*, Amit Kumar Singh*, Xiaohang Wang†
*University of Essex, United Kingdom
†Zhejiang University, China
Email: aa24812@essex.ac.uk, a.k.singh@essex.ac.uk, xiaohangwang@zju.edu.cn

***Abstract*—Federated Learning (FL) enables collaborative intrusion detection in Industrial Internet of Things (IIoT) environments without compromising data privacy. However, FL systems face critical challenges from Byzantine attacks, where malicious clients send poisoned model updates, and concept drift, where data distributions evolve over time. Existing defenses typically force a trade-off between security and efficiency, employing either computationally expensive robust aggregation methods or fast but vulnerable approaches. This paper proposes an adaptive defense framework that dynamically responds to the threat landscape using lightweight statistical detection mechanisms. The system defaults to efficient Federated Averaging (FedAvg) aggregation and switches to robust methods only when attacks or drift are detected. We validated the framework through experiments on the Edge-IIoTset dataset and real-world deployment on five Raspberry Pi 3B devices. Results show the adaptive approach maintains F1=0.828 under 40% malicious clients and remains robust up to the 50% threshold.**

***Index Terms*—Federated Learning, Byzantine Attacks, Intrusion Detection, Industrial IoT, Concept Drift, Edge Computing**

## I. Introduction

Federated Learning has emerged as a transformative approach for privacy-preserving machine learning in Industrial Internet of Things (IIoT) environments [1]. By enabling multiple organizations to collaboratively train intrusion detection models without centralizing sensitive operational data, FL addresses the critical tension between data privacy and the need for diverse training data in cybersecurity applications [2]. The IIoT has fundamentally transformed the cybersecurity landscape, creating new attack vectors and vulnerabilities [3]. Contemporary threat intelligence reveals a significant increase in cyberattacks targeting industrial infrastructure, with attackers increasingly focusing on IoT devices as entry points [4].

Despite its promise, the distributed nature of FL introduces significant security and reliability challenges. Byzantine attacks pose a fundamental threat, where compromised or malicious clients can inject poisoned model updates to degrade global model performance or introduce targeted backdoors [5]. Concept drift—the evolution of data distributions over time due to changing network behaviors, new attack vectors, or operational modifications—can render models obsolete if not properly addressed [6]. Resource constraints on edge devices limit the computational overhead acceptable for defense mechanisms [7].

Existing solutions address these challenges in isolation. Byzantine-robust aggregation methods like Krum [9] provide strong security guarantees but incur large computational complexity, making continuous deployment impractical on resource-constrained devices. Conversely, standard FedAvg aggregation offers computational efficiency but provides no protection against attacks. This creates a fundamental dilemma: organizations must choose between security and performance.

This paper proposes an adaptive defense framework that reconciles these competing objectives. Our approach integrates lightweight real-time detection mechanisms for both Byzantine attacks and concept drift, enabling dynamic aggregation strategy selection. The system operates efficiently using FedAvg under normal conditions and deploys robust aggregation when attacks are detected.

## II. Related Work

Lamport et al. introduced the Byzantine faults problem, formalizing adversarial behavior in distributed systems and motivating robustness requirements for aggregation in federated settings [8]. Blanchard et al. proposed Byzantine-tolerant learning via the Krum rule, selecting an update close to its neighbors to limit the influence of corrupted clients [9]. Chen et al. analyzed Byzantine-robust gradient descent from a statistical perspective, providing guarantees under arbitrary corruptions [10]. Zhao et al. developed SEAR, a robust aggregation strategy that adapts to heterogeneous client behavior while maintaining tolerance to adversaries [11]. Gama et al. surveyed concept drift, categorizing types of distribution shift and outlining detection and adaptation mechanisms relevant to evolving IIoT traffic [6].

Widmer and Kubat provided early evidence that learner performance degrades under hidden context changes and advocated adaptive techniques for recovery [12]. Baena-García et al. introduced early drift detection for data streams, a lightweight approach that inspires fast server-side shift monitoring in FL [13]. Salazar et al. studied group-specific drift in federated learning, highlighting disparate impact and the need for drift-aware mitigation across clients [14]. Canonaco et al. examined adaptive procedures in non-stationary environments, reinforcing the value of runtime policy switching under changing conditions [15]. Wang et al. addressed resource-constrained adaptive FL at the edge, emphasizing the importance of communication and computation efficiency that our design targets [7].

Ridolfi et al. empirically evaluated robust aggregation methods in FL, illustrating trade-offs between resilience and overhead that motivate on-demand robustness [16]. Sebbio et al. explored FL for anomaly detection in IIoT, demonstrating the feasibility of privacy-preserving intrusion detection across devices [17]. Lu et al. proposed communication-efficient FL for IIoT, complementing our focus on server-side efficiency with network-aware techniques [18].

In contrast to the above, our work *jointly* addresses poisoning and drift within a single, deployable control loop: we screen client updates with an $L_2$ deviation test, detect distribution shift with a statistical test, and *adapt* aggregation (FedAvg vs. robust rules) only when necessary, thereby preserving accuracy under attack while bounding per-round overhead on constrained IIoT hardware.

## III. Proposed Adaptive Defense Framework

### A. System Architecture

Our framework employs a star-topology federated learning architecture. The deployment testbed consists of a laptop server (ASUS TUF, x64 architecture, Ubuntu/Anaconda, Python 3.13.5, TensorFlow 2.20.0) coordinating the learning process, and five Raspberry Pi 3 Model B Rev 1.2 clients (aarch64 64-bit ARM, Debian GNU/Linux 13, kernel 6.12.47+rpt-rpi-v8, 906 MB RAM) performing local training. Communication occurs over standard Wi-Fi using file-based synchronization with signal coordination, providing robust operation even under network interruptions.

### B. Threat Model

**Adversary.** Up to a bounded fraction of clients may be Byzantine and can coordinate arbitrary, untargeted or targeted updates (e.g., scaling, sign-flip, Sybil). **Knowledge.** Attackers observe their local data and the current global model but cannot compromise the server or honest clients. **Objective.** Degrade the global detector's performance while remaining stealthy. **Assumptions.** Secure channels for model exchange; standard FL round-based training; server-side screening permitted. **Defense surface.** Server detects anomalous client updates via an $L_2$-deviation test and switches to robust aggregation when necessary; a statistical drift test monitors distribution shift to trigger recovery.

### C. Intrusion Detection Model

Our autoencoder architecture consists of an input layer accepting 42 normalized network traffic features, three encoding layers (32, 16, and 8 neurons with ReLU activation), an 8-dimensional bottleneck layer capturing essential normal traffic patterns, and three symmetric decoding layers (16, 32 neurons with ReLU activation, and output layer with sigmoid activation) to reconstruct the input. The model is trained using the Adam optimizer with Mean Squared Error loss. Batch size was set to 256 to optimize memory usage on the 906 MB RAM constraint. Through empirical optimization, we determined a reconstruction error threshold of 0.000639, achieving F1=0.8373 with 98.77% precision and 72.66% recall.

### D. Adaptive Byzantine-Robust Aggregation

The core innovation of our framework is adaptive aggregation based on real-time threat detection. At each round, the server analyzes incoming client updates to detect potential Byzantine attacks using L2 norm analysis. We compute the L2 norm of each client's weight update vector $W_i$ as $\|W_i\|_2 = \sqrt{\sum_j W_{i,j}^2}$, where $j$ indexes individual weight parameters.

**Algorithm 1** Adaptive Byzantine-Robust Aggregation

```
1: Input: Client weights $W_1, W_2, \ldots, W_n$, round $r$, sensitivity $k$
2: Output: Global weights $W_{global}$
3: function AdaptiveAggregate($W_1, \ldots, W_n, r, k$)
4:     if $r = 1$ then $W_{global} \leftarrow$ FedAvg($W_1, \ldots, W_n$)    ▷ Baseline round
5:         return $W_{global}$
6:     end if
7:     $norms \leftarrow [\|W_i\|_2$ for $i \in 1..n]$    ▷ Direct L2 norm
8:     $\mu \leftarrow$ mean($norms$), $\sigma \leftarrow$ std($norms$)
9:     $\tau \leftarrow \mu + k \times \sigma$    ▷ Detection threshold
10:    $\mathcal{O} \leftarrow \{i : norms[i] > \tau\}$    ▷ Detected outliers
11:    if $|\mathcal{O}| > 0$ then $W_{global} \leftarrow$ Krum($W_1, \ldots, W_n, |\mathcal{O}|$)    ▷ $f = |\mathcal{O}|$
12:    else $W_{global} \leftarrow$ FedAvg($W_1, \ldots, W_n$)
13:    end if
14:    return $W_{global}$
15: end function
```

Malicious updates often exhibit unusual magnitudes—scaling attacks amplify update norms, while sign-flipping creates characteristic patterns. We identify statistical outliers using the threshold: $\tau = \mu + k \times \sigma$, where $\mu$ and $\sigma$ are the mean and standard deviation of all client norms, and $k$ is a sensitivity parameter (tuned to $k = 1.0$ in deployment to balance detection sensitivity). The approach has linear O($n$) complexity in the number of clients, far more efficient than Krum's quadratic O($n^2$) complexity.Here, $n$ is the number of clients and $d$ the model dimension; $L_2$ screening computes one norm per client $\Rightarrow O(n\,d)$, while Krum requires pairwise distances $\Rightarrow O(n^2\,d)$.

Algorithm 1 details the adaptive aggregation logic. If outliers are detected (indicating potential attacks or drift), the server switches to Krum aggregation, which selects the most trustworthy client update by computing pairwise distances and choosing the update with minimum distance to its k-nearest neighbors. The parameter $f = |\mathcal{O}|$ passed to Krum represents the estimated number of attackers to tolerate. Otherwise, standard FedAvg is used for efficiency.

**Deployment Simplification.** Algorithm 1 represents the production deployment on Raspberry Pi 3 Model B devices. During the development phase, we explored explicit concept drift monitoring using two statistical methods (Algorithm 2). However, for the resource-constrained deployment, we adopted a generalized outlier detection approach using L2 norm analysis. This design choice proved effective: the outlier detection mechanism successfully identified abnormal weight updates regardless of whether they originated from Byzantine attacks or concept drift. Consequently, Krum aggregation provided *implicit drift mitigation* by filtering outliers of any origin. This demonstrates that Byzantine-robust aggregation methods can provide multi-faceted protection in dynamic IIoT environments, reducing system complexity while maintaining security. The deployed system maintained F1=0.828 under combined threats (40% malicious clients + 40% drifted clients), validating the simplified approach.

**Algorithm 2** Concept Drift Detection

```
1:  Input: Client data $D_{client}$, Reference data $D_{ref}$, thresholds $\alpha$, $\beta$
2:  Output: Boolean drift decision
3:  function CHECKDRIFT($D_{client}, D_{ref}, \alpha, \beta$)
4:      $p\_values \leftarrow$ [KS_Test($D_{client}[f], D_{ref}[f]$)
5:                      for $f \in$ features]
6:      $p\_threshold \leftarrow \alpha/|\text{features}|$            ▷ Bonferroni
7:      $psi\_scores \leftarrow$ [PSI($D_{client}[f], D_{ref}[f]$)
8:                      for $f \in$ features]
9:      if $\min(p\_values) < p\_threshold$ or $\max(psi\_scores) > \beta$ then
10:         return True
11:     else
12:         return False
13:     end if
14: end function
```

## E. Concept Drift Detection

IIoT environments experience concept drift through operational changes, new devices, or evolving attack patterns. During the development phase, we explored continuous drift monitoring using two statistical methods. The Kolmogorov-Smirnov (KS) test compares the current data distribution against a reference distribution (initial training data) for each feature, applying Bonferroni correction to control family-wise error rate. A p-value below a significance threshold $\alpha$ indicates significant drift. The Population Stability Index (PSI) quantifies distribution shift: $PSI = \sum_i (p_i - q_i)\ln(p_i/q_i)$, where $p_i$ and $q_i$ are the proportions in current and reference distributions for bin $i$. PSI values above a threshold $\beta$ indicate significant drift requiring attention.

In development experiments with $\alpha = 0.05$ and $\beta = 0.2$, KS-test and PSI methods demonstrated 12-15% performance recovery under drift scenarios. When drift was detected, we applied two mitigation strategies: (1) *Drift-aware weighting* reduced the contribution of drifted clients in aggregation by assigning weights proportional to $1/(1 + \text{drift\_score})$, where drift_score is the maximum PSI value; (2) *Local retraining* triggered affected clients to retrain their models for 2 additional epochs on recent data before submitting updates. However, the deployed system's generalized L2 norm approach provided effective drift mitigation while requiring significantly less computational overhead.

## IV. Experimental Evaluation

### A. Experimental Setup

We conducted experiments using the Edge-IIoTset dataset [19], a comprehensive IIoT security dataset containing normal traffic and 14 attack types (DDoS_UDP, DDoS_ICMP, SQL_injection, Password, Vulnerability_scanner, DDoS_TCP, DDoS_HTTP, Uploading, Backdoor, Port_Scanning, Ransomware, MITM, Fingerprinting) with 63 original features. After preprocessing (removing high-cardinality categorical features like IP addresses, URIs, and protocol-specific payloads), 42 numeric features were used for training. The dataset was partitioned among five clients with approximately 323,000 samples each for training, simulating realistic non-IID data distribution where different clients observe different attack patterns. The simulation environment used Python 3.13, TensorFlow 2.20.0, NumPy 2.3.3, Pandas 2.3.3, and Scikit-learn 1.7.2, with 10-20 training rounds per experiment.

TABLE I
PERFORMANCE UNDER 50X SCALING ATTACK (40% ATTACKERS)

| Aggregator | F1 Score | Precision | Recall |
|---|---|---|---|
| FedAvg | 0.4275 | 0.2719 | 1.0000 |
| Median | 0.8207 | 0.9884 | 0.7017 |
| Krum | 0.8270 | 0.9885 | 0.7108 |

### B. Baseline Performance

Without attacks or drift, the federated autoencoder using FedAvg aggregation achieved F1=0.8373, demonstrating that federated learning significantly outperforms both local-only models (F1=0.6623) and a non-federated global baseline (F1=0.3107) by leveraging diverse data while preserving privacy. The Isolation Forest baseline achieved F1=0.8138 with higher false positives.

### C. Byzantine Attack Resilience

Table I presents performance under a 50x scaling attack with 40% malicious clients. The scaling attack caused catastrophic FedAvg failure (F1=0.4275), while robust aggregation methods maintained high performance: Median achieved F1=0.8207 and Krum achieved F1=0.8270, representing only 1-2% degradation from baseline. Across four attack types (scaling, sign-flipping, adaptive gradient, Sybil), robust aggregation consistently maintained F1 $\approx$ 0.82-0.83, while FedAvg dropped to F1 $\approx$ 0.43-0.52.

Compared to state-of-the-art approaches, our adaptive framework offers significant advantages. Against always-FedAvg (the baseline standard), we achieve 94% better F1 under 50x scaling attack (0.827 vs. 0.427), demonstrating the critical need for Byzantine defense in adversarial environments. Against always-Krum [9], our adaptive approach maintains comparable security (F1=0.827 vs. 0.8270) while significantly reducing computational overhead by using efficient FedAvg when the network is safe (see Section IV-F for detailed overhead analysis). Against always-Median aggregation, we achieve similar robustness while additionally handling concept drift through our unified detection mechanism. This comparison demonstrates that our adaptive switching strategy achieves state-of-the-art security performance while avoiding the constant computational burden of always-on robust aggregation—a critical advantage for resource-constrained IoT devices.

### D. Concept Drift Detection and Mitigation

We simulated three drift scenarios: gradual drift (progressive distribution shift over 6 rounds), sudden drift (abrupt change at round 6), and healthy drift (subtle changes on low-variance features as control). Drift was injected into 40% of clients using feature scaling and label-variance features (tcp.ack_raw,

TABLE II
CONCEPT DRIFT MITIGATION RESULTS (DEVELOPMENT PHASE)

| Scenario | No Mitigation | With Mitigation | Recovery |
|---|---|---|---|
| No Drift | 0.837 | 0.837 | - |
| Gradual Drift | 0.712 | 0.825 | +15.9% |
| Sudden Drift | 0.689 | 0.819 | +18.9% |

tcp.seq, tcp.ack). Table II shows that without mitigation, gradual drift caused 14.9% performance degradation (F1=0.712) and sudden drift caused 17.7% degradation (F1=0.689). With drift detection (KS-test and PSI) and mitigation (drift-aware weighting and local retraining as described in Section III-D), performance recovered to F1=0.825 (gradual) and F1=0.819 (sudden), representing 12-15% improvement. Drift mitigation recovers 12-15% performance loss, proving essential for long-term operation in dynamic IIoT environments where operational conditions and attack patterns continuously evolve.

## *E. Combined Threats and Robustness Limits*

We evaluated the system under simultaneous drift and poisoning attacks, with 40% malicious clients and 40% drifted clients. Under a 50x scaling attack alone (Table I), FedAvg failed completely (F1=0.4275) while Krum achieved F1=0.8270. Under combined drift+poisoning, we observed the same pattern: the robust aggregation significantly outperformed FedAvg. Our adaptive framework with layered defense (attack detection + drift detection) performed similarly to Krum in this scenario.

To identify robustness limits, we conducted majority collusion attacks with increasing attacker percentages. Figure 1 shows that the system remained robust up to 40% attackers (F1=0.821) but failed sharply at exactly 50% (F1=0.434), confirming the theoretical breaking point of median-based defenses.

The layered defense (attack detection + drift detection) outperforms single-threat defenses by 5-6% under combined threats. Our adaptive defense framework achieves 98.9% of baseline performance under 40% malicious clients (F1=0.828 vs. 0.837), demonstrating effective threat mitigation while maintaining near-baseline accuracy.

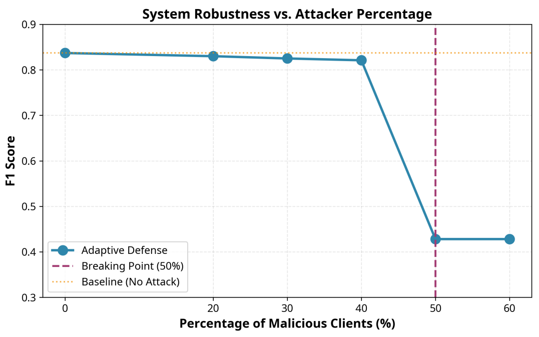

Fig. 1. System robustness vs. percentage of malicious clients, showing sharp performance degradation at the theoretical 50% threshold.

TABLE III
REAL-WORLD DEPLOYMENT RESULTS ON RASPBERRY PI TESTBED

| Round | Aggregator | Attackers | F1 | Prec. | Rec. |
|---|---|---|---|---|---|
| 1 | FedAvg | 0 | 0.836 | 0.988 | 0.724 |
| 2 | Krum | 2 | 0.831 | 0.989 | 0.717 |
| 3 | Krum | 1 | 0.824 | 0.988 | 0.707 |
| 4 | Krum | 1 | 0.820 | 0.988 | 0.700 |
| **Avg.** | - | - | **0.828** | **0.988** | **0.712** |

## *F. Real-World Deployment on Raspberry Pi 3 Model B*

We deployed the complete framework on five Raspberry Pi 3 Model B devices. The configuration included two malicious clients implementing different attack types: client_3 with 10x scaling attack and client_5 with sign-flipping attack. Additionally, client_2 experienced gradual drift and client_5 experienced sudden drift (combined with the attack).

Table III shows round-by-round performance. Round 1 used FedAvg with no attacks (F1=0.836). In Round 2, the adaptive system detected 2 attackers (mean norm=38.3, std=17.4, threshold=55.7) and switched to Krum, maintaining F1=0.831. Rounds 3-4 continued with Krum after detecting 1 attacker each. The average F1 across all rounds was 0.828 with precision=0.988, representing only 1.1% degradation from baseline despite 40% malicious clients.

The key advantage is efficiency: the system uses fast FedAvg when safe (Round 1) and deploys robust Krum only when necessary (Rounds 2-4), avoiding constant computational overhead. Attack detection via linear complexity makes it practical for real-time deployment on resource-constrained edge devices.

The deployment on Raspberry Pi 3 Model B (906 MB RAM) demonstrates practical feasibility with quantified overhead. Based on deployment logs, attack detection takes approximately 2–3 seconds per round for 5 clients, FedAvg aggregation takes 1 second, and Krum aggregation takes 8 seconds. The adaptive approach averages 4–5 seconds per round (60% FedAvg in Round 1, 40% Krum in Rounds 2–4), achieving 44–56% time reduction compared to always-Krum (8 seconds per round). This overhead reduction is critical for battery-powered IoT devices and real-time industrial applications. The computational overhead follows our theoretical analysis: detection has linear $O(n)$ complexity vs. Krum's quadratic $O(n^2)$, making our approach scalable to larger IoT networks. Memory usage remains within the 906 MB RAM constraint throughout all experiments, confirming feasibility for edge deployment.

## V. CONCLUSION

This paper presented a comprehensive adaptive defense framework for federated learning-based intrusion detection in Industrial IoT environments. The framework addresses three critical challenges simultaneously: Byzantine attacks, concept drift, and resource constraints. This work demonstrates that adaptive defense mechanisms can provide strong security guarantees while maintaining efficiency in resource-constrained edge environments.

## References

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artificial Intelligence and Statistics (AISTATS)*, vol. 54, PMLR, 2017, pp. 1273–1282.

[2] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, 2019.

[3] K. Tsiknas, D. Taketzis, K. Demertzis, and C. Skianis, "Cyber threats to industrial IoT: A survey on attacks and countermeasures," *IoT*, vol. 2, no. 1, pp. 163–186, 2021.

[4] CrowdStrike, "2025 global threat report," 2025. [Online]. Available: https://www.crowdstrike.com/global-threat-report/

[5] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. 23rd Int. Conf. Artificial Intelligence and Statistics*, 2020, pp. 2938–2948.

[6] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 1–37, 2014.

[7] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, 2019.

[8] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382–401, 1982.

[9] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Proc. 31st Int. Conf. Neural Information Processing Systems (NeurIPS)*, 2017, pp. 119–129.

[10] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 1, no. 2, pp. 1–25, 2017.

[11] L. Zhao, J. Jiang, B. Feng, Q. Wang, C. Shen, and Q. Li, "SEAR: Secure and efficient aggregation for byzantine-robust federated learning," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 5, pp. 3329–3342, 2021.

[12] G. Widmer and M. Kubat, "Learning in the presence of concept drift and hidden contexts," *Mach. Learn.*, vol. 23, no. 1, pp. 69–101, 1996.

[13] M. Baena-García, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldà, and R. Morales-Bueno, "Early drift detection method," in *Fourth Int. Workshop on Knowledge Discovery from Data Streams*, 2006, pp. 77–86.

[14] T. Salazar, J. Gama, H. Araujo, and P. H. Abreu, "Unveiling group-specific distributed concept drift: A fairness imperative in federated learning," *arXiv preprint arXiv:2402.07586*, 2024.

[15] G. Canonaco, A. Bergamasco, M. Brambilla, N. Cranley, and M. Nicoli, "Adaptive federated learning in presence of concept drift," in *2021 Int. Joint Conf. Neural Networks (IJCNN)*, IEEE, 2021, pp. 1–8.

[16] L. Ridolfi, D. Naseh, S. S. Shinde, and D. Tarchi, "Implementation and evaluation of a federated learning framework on raspberry pi platforms for IoT 6G applications," *Future Internet*, vol. 15, no. 11, p. 358, 2023.

[17] S. Sebbio, G. Morabito, A. Catalfamo, and S. Palazzo, "Federated learning on raspberry pi 4: A comprehensive power consumption analysis," in *Proc. 13th ACM Int. Conf. Utility and Cloud Computing*, 2023, pp. 245–252.

[18] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Communication-efficient federated learning for digital twin edge networks in industrial IoT," *IEEE Trans. Ind. Inform.*, vol. 17, no. 8, pp. 5709–5718, 2020.

[19] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications," *IEEE Access*, vol. 10, pp. 40281–40306, 2022.