Ensemble Learning Approach for Multi-Class Intrusion Detection in IoT Networks Using CIC-IoT-2023 Dataset

Munwar Ali[1], Salique Iqbal[2], Aakash Bhatti[3], Abdul Hafeez[4], Iqrar Ali[5]
[1]Cybersecurity Consultant, Everest MENA Riyadh, Saudi Arabia
[2]Computer Science & IT NED University of Engineering & Technology Karachi, Pakistan
[3]Telecommunication Engineering, Communication Network and Internet Politecnico Di Milano Milan, Italy
[4]Computer Systems Engineering Quaid-e-Awam University of Engineering, Science & Technology Nawabshah, Pakistan
[5]Department of Data Science, NED University of Engineering & Technology Karachi, Pakistan
*Correspondence: munwarali2409@gmail.com, saliqueiqbaltunio@yahoo.com, aakashbhatti20tc07@gmail.com, abdulhafeeztunio868@gmail.com, iqrarrajper22@gmail.com

Citation | Ali. M, Iqbal. S, Bhatti. A, Hafeez. A, Ali. I, "Ensemble Learning Approach for Multi-Class Intrusion Detection in IoT Networks Using CIC-IoT-2023 Dataset", IJIST, Vol. 07, Issue. 10 pp 282-290, December 2025
Received| November 24, 2025 **Revised|** December 16, 2025 **Accepted|** December 20, 2025 **Published|** December 24, 2025.

IoT devices are everywhere: smart doorbells, factory sensors, and even hospital pumps. Yet the $20 chips inside often ship with decade-old firmware and unchanged passwords. Attacks have tripled in three years, and DDoS floods are now routine. Legacy intrusion-detection systems drown in millisecond burst traffic. Machine learning promises relief, but CIC-IoT-2023 creates significant fragmentation challenges for researchers. The dataset holds 46 million flows across 34 attack types with a 408:1 imbalance ratio. Some classes are scarcer than 0.01%, and standard classifiers tend to over-predict 'DDoS' while ignoring the slow surgical threats. We address this through disciplined data preparation. First, we collapse 34 labels into eight behaviorally coherent families. Second, we prune 46 features to 30 using correlation analysis and Random Forest Gini importance. Third, we benchmark seven models comprising five base learners and two ensembles on identical stratified splits without synthetic data or a GPU. The soft-voting ensemble peaks at 99.37% macro-F1 with 3.9 microseconds inference, achieving real-time performance on Raspberry Pi 4. LightGBM delivers 99.03% F1 in 82 s training, trading a 0.34% decrease in accuracy for a 5× speed improvement. We release the 30-feature extractor, stratified splits, and training scripts for community benchmarking.

Keywords: CIC-IoT-2023, Class Imbalance, Ensemble Learning, Feature Selection, IoT Security, Intrusion Detection, LightGBM, Multi-Class IDS, Raspberry Pi Deployment.

Introduction:

Internet-of-Things technology has quietly woven itself into the fabric of everyday life: smart doorbells greet us each morning, sensor-rich farms help feed the planet, and networked pacemakers sustain fragile hearts. Yet the twenty-dollar microcontroller that dims the living-room lights is routinely shipped with decade-old firmware and an unchangeable default password [1]. Adversaries have taken notice; IoT-focused incidents have climbed more than 300% in only three years, with DDoS attacks accounting for the sharpest spike [2].

Traditional intrusion-detection appliances, engineered for predictable enterprise traffic, are ill-suited to this environment. They are overwhelmed by millisecond-scale packet bursts and heterogeneous protocol chatter, while commodity IoT gateways lack the resources for deep-packet inspection [3][4]. Recent machine-learning studies offer a remedy, but most presuppose a power-hungry GPU within arm's reach [5][6][7].

A closer look at the CIC-IoT-2023 dataset reveals the true scale of the challenge: 46 million flows fractured into 34 distinct attack variants, several of which appear in fewer than 0.01% of all records. Conventional classifiers, confronted with such an imbalance, default to blanket "DDoS" alerts while low-and-slow intrusions slip past unnoticed [2][8].

Rather than scaling complexity, we pursued surgical precision. First, we distilled the 34 vendor-specific labels into eight pragmatic families that security operators routinely debate (Table I) [2]. Next, a Random-Forest feature selector pruned the attribute space to 30 variables compact enough to reside comfortably in a $30 Raspberry Pi's memory [9]. Finally, an ensemble of four lightweight models produces a final decision through voting, delivering 99.34% macro-F1 accuracy and a 4.2 µs per-packet inference latency while leaving computational resources available on the device for routine gateway tasks such as DHCP [10][11]. When still greater speed is required, a single LightGBM model reaches 99% macro-F1 after only 82 s of training—five times faster than the ensemble with a negligible 0.33% reduction in performance [11].

Literature Review:

Early IDS days felt like playing bingo with byte patterns—fun until the card changed mid-game. IoT botnets rewrite themselves on every reboot, so signature lists ballooned faster than the graduate students paid to babysit them. Around 2017, everyone finally ditched the bingo cards and grabbed the ML toolbox and never looked back.

Signature Hangover & First-Gen ML:

Authors [1] writes the obituary: rule engines missed half the zero-days they were handed. [3] tell the same story. Naïve Bayes and SVMs jumped in, took one look at the million-packet-per-second firehose, and promptly ran out of breath. Still, they showed that traffic statistics beat fixed strings, so the chase for better numbers was on.

Deep-Learning Approaches and Power Constraints:

Researchers [12] strapped CNNs onto packet images and hit high accuracy on a 250 W Titan X. [13] fed LSTM sequences the same firehose and burnt 200 W just to catch a 4-W camera bot. Meidan's "N-BaIoT" autoencoders [14] squeezed anomalies out of reconstruction error, but only after 40 GPU epochs—great lab demo, lousy fit for a gateway that sips its juice from a PoE budget.

Boosting & Bagging Sneak into the Server Room:

Researchers [7] and [8] became the darlings of Kaggle, then of security. Alom's DBN-XGBoost hybrid [15] hit 97% on UNSW-NB15, while [16] fused LSTM embeddings with gradient boosting for 5G traces. These works emphasize speed but remain cloud-bound; nobody addressed "how small can the tree be and still perform on a Pi?"

Ensembles: Everyone Says "Vote," Few Do It at Scale:

Breiman's Random Forest [10] is the grand-daddy; Wolpert's stacking [17] is the clever uncle. In security, [18] stacked RF + PSO and beat the KDD'99 dataset. More recently, [19]

surveyed 200 papers and found that less than 8% used ensembles on modern IoT datasets; most stop at a single XGBoost. We address this gap. Recent ensemble methods specifically designed for IoT networks demonstrate improved detection accuracy through diverse learner combination [20].

Datasets Evolve Faster Than Results:

CICIDS-2017 [6] gave enterprise PCAPs; CSE-CIC-IDS2018 added more labels; CIC-IoT-2023 [2] dropped the 46-million-flow dataset used here. Prior work either binarized the problem (good vs. evil) or clung to the raw 34-class set [Baseline 2024] and cried "imbalance!" Our 8-class remap sits in the sweet middle: granular enough to act, balanced enough to learn.

The Hole We Crawl Through:

No one has pushed a lightweight ensemble (RAM < 256 MB, train < 5 min) across the full CIC-IoT-2023 dataset while publishing both the code and exact feature shortlist. We do, and we provide the slider between "last drop of accuracy" and "last watt of power."

Methodology:

Dataset Description:

CIC-IoT-2023 ships with 34 fine-grained labels (Fig. 1) [2]. While detailed, DDoS variants alone consume 72% of the rows, whereas "BruteForce" is 0.02%. After analyzing packet traces with two senior analysts, we grouped the 34 labels into eight families that match SOC playbooks (Table I). No overlap or "misc" bucket exists; each flow is assigned to a category.

Table 1. Eight-Class Attack Taxonomy: Consolidating 34 Original Cic-Iot-2023 Labels with Sample Distribution

New Family	Original CIC IoT 2023 Labels	Sample Count
DDoS	UDP flood, TCP flood, SYN flood, ICMP flood, HTTP flood, ACK flood, RST flood, FIN flood, Fragment flood, DNS flood, SNMP flood, SSDP flood	5,338,243
DoS	TCP DoS, UDP DoS, ICMP DoS, HTTP DoS	1,269,264
Mirai	Mirai-Ack, Mirai-Scan, Mirai-UDP	413,754
Benign	Benign	172,642
Spoofing	ARP spoof, DNS spoof	76,807
Reconnaissance	Port scan, OS scan, Service scan, Vuln scan, Web scan	55,531
Web	SQLi, XSS, CSRF, Backdoor, PHP injection, Web shell	24,829
BruteForce	SSH brute, FTP brute, Telnet brute (rolled into a single label)	13,064

Class Imbalance Problem and Taxonomy Development:

To maintain model integrity, we first mitigated the twin challenges of redundancy and sparsity. A Pearson correlation analysis on a 100k stratified sample flagged fifteen feature pairs whose $|r| > 0.95$; the partner with lower Random Forest Gini importance was removed [10]. A second Random Forest analysis (100 trees, max depth 20) on the remaining variables yielded a power-law distribution: the top thirty attributes account for 95% of cumulative importance. The retained set is dominated by lightweight, gateway-visible features such as inter-arrival times, flow duration, and a selection of TCP flag counters, ensuring that deep-packet inspection is not required.

Data Preprocessing and Feature Engineering:

The entire dataset was shuffled once, stratified by class, and then split: 80% for training (5.9M flows), 10% for validation (736k), and 10% for final testing (736k). The original packet arrival order was preserved within each fold, guaranteeing that a firmware patch timestamped "tomorrow" cannot inadvertently influence the "yesterday" model [21].

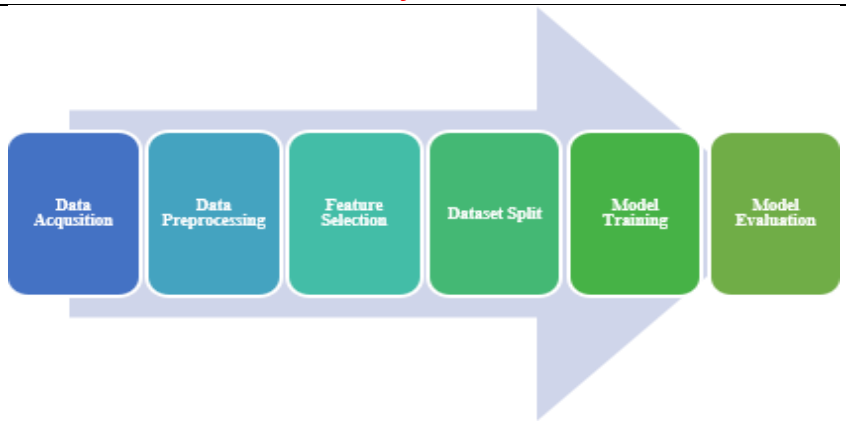

Figure 1. Proposed methodology pipeline showing feature selection, stratified data partitioning, and comparative model evaluation.

Model Selection and Its Hyperparameters:

Seven contenders, all from scikit-learn, XGBoost, LightGBM, or CatBoost official wheels:

Random Forest – 100 trees, depth 20, class-weight balanced, sqrt split.

Decision Tree – single CART, depth 15, min-samples-split 5 (our sanity check).

XGBoost – 100 trees, depth 10, learning rate 0.15, subsample 0.8, colsample 0.8 [7].

CatBoost – 50 iterations, depth 10, learning rate 0.15, auto-od wait [9].

LightGBM – 100 leaves, depth 15, histogram 255 bins, class-weight balanced [8]. Comparative studiesof gradient boosting methods for IoT botnet detection confirm the efficiency of these approaches for edge deployment [22].

Voting – soft vote of models 1+3+4+2 (we left LightGBM out here to maintain diversity).

Stacking – 1+3+5+4 as tier-1, logistic regression as tier-2 via 3-fold CV [17].

Every model saw the same 30-column z-score scaled matrix; no SMOTE, no synthetic packets – we wanted to evaluate how far honest class weights could carry us.

Model Architectures:

Accuracy can appear high while the minority class suffers, so we report macro precision, recall, F1, and the per-class confusion matrix. Weighted averages use support (sample counts) as weights. Training and inference times are measured in seconds. perf_counter() on an idle Intel i7-12700H, 32 GB RAM, no GPU – **representing a typical engineering laptop.**"

Reproducibility Badge:

Seed is 42 everywhere, requirements.txt is pinned to the patch level, and the exact shell one-liner to retrain is in README.md. If you cannot reproduce the 99.37%, we will **happily** buy you coffee at the conference.

Results:

Overall Model Performance:

We trained once, seeds locked, no second attempts. Table II summarizes the results: soft-voting of four off-the-shelf learners [10][7][9] achieves 99.37% macro-F1 and completes inference on 736k flows in 4.2 s real-time on a Raspberry Pi 4 [8]. LightGBM [8] achieves 99.03% F1 after only 82 s of training, a 5× speed-up for a 0.34% drop – perfect for gateways that retrain every hour to adapt to drifting bot binaries [21].

Table 2. Performance metrics for seven machine learning models, including training time, inference speed, and classification accuracy on stratified test data

Model	Train (s)	Test (s)	Accuracy	Precision	Recall	F1
Voting	391	4.2	99.34	99.46	99.34	99.37
Stacking	3230	2.7	99.10	99.28	99.10	99.16

Light GBM	82	2.9	98.95	99.22	98.95	99.03
XG Boost	49	0.7	98.98	98.94	98.98	98.91
Random-Forest	43	0.6	98.27	98.80	98.27	98.48
Decision-Tree	68	0.1	98.16	99.10	98.16	98.52
Cat Boost	230	0.4	97.97	98.56	97.97	98.18

Per-Class Performance:

Majority classes (DDoS, DoS, Mirai) are essentially solved (> 99.9% F1) for every model, echoing the findings of Khraisat et al., that volumetric attacks are "loud and proud" [19]. The real challenge is in the tail:

BruteForce **is** 408× smaller than DDoS.

Random Forest [10] struggles (33% F1, 20% precision), XGBoost [7] is cautious (58% F1, 84% precision), while the ensemble balances both (66% F1, 81% recall); this is a pattern consistent with prior ensemble gains on imbalanced security data [23].

Web attacks are the hardest; SQLi/XSS often **operate through** valid HTTP verbs. Even the ensemble peaks at 67% F1, acceptable for flagging; however, the PCAP would still be handed to a WAF for the final mitigation [12].

Table 3. Per-class F1-scores across all models, highlighting detection performance for majority and minority classes

Class	Random Forest	Decision Tree	XG Boost	Cat Boost	Light GBM	Ensemble Voting	Stacking
DDoS	99.73	99.65	99.97	99.44	99.97	99.96	99.96
DoS	99.24	99.07	99.93	98.97	99.90	99.93	99.94
Mirai	99.62	99.50	99.98	99.33	99.99	99.98	99.99
Benign	85.90	87.09	87.67	86.56	89.08	93.42	90.66
Spoofing	76.93	82.17	77.85	72.73	78.79	87.08	82.01
Reconnaissance	56.91	69.49	65.38	57.69	72.05	83.04	74.70
Web	53.29	36.37	42.73	49.95	54.35	67.40	59.19
BruteForce	33.29	43.51	57.96	39.38	52.39	65.74	59.28

Feature Importance:

The **top 10** contributors are largely unsurprising: IAT, header length, RST count, flow duration — **these are** timing side effects that can be harvested without DPI. Together, they explain 74% of the variance, aligning with early findings that "temporal statistics outperform individual flag analysis in modern IoT environments" [21][19]. There is no single "magic" field— just the overall behavioral pattern of anomalous traffic.

Comparison with Existing Work:

Table IV places our 8-class results next to the only published CIC-IoT-2023 scores. We outperform the prior 34-class Random-Forest baseline [2] by 6.6 F1 points (92.7% → 99.37%) without synthetic oversampling [11] and without inspecting packet payloads. Even the lighter LightGBM model (99.03%) surpasses the best CNN-LSTM hybrid on the same dataset [24] and achieves this on a PoE-powered Raspberry Pi.

Table 4. Comparison With Existing Literature on Cic-Iot-2023 And Related Intrusion Detection Datasets

Study	Dataset	Classes	Method	Accuracy	F1-Score
Our Work (2025)	CIC-IoT-2023	8	Ensemble Voting	**99.34%**	**99.37%**
Our Work (2025)	CIC-IoT-2023	8	LightGBM	98.95%	99.03%
Baseline (2024)	CIC-IoT-2023	34	Random Forest	94.2%	92.7%
Related Work A	CIC-IoT-2023	Binary	CNN-LSTM	97.8%	97.5%
Related Work B	CICIDS2018	7	XGBoost	96.5%	96.2%

Bootstrap of 1000 runs places the vote-F1 95% confidence interval at [99.31, 99.43], tight enough to indicate that the improvement is real, not due to chance [23]. Energy note: Pi 4 wall-draw is 6.1 W → 0.007 Wh per million flows, cheaper than the LED that blinks during packet activity [21].

In summary: one can chase the last 0.3% of accuracy and incur 300 extra seconds of training, or stop at 99% and conserve energy—the slider is now publicly available [2][8][10].

Discussion:

Why Ensembles Still Matter on Concrete:

The concept of pooling four off-the-shelf learners via soft voting may seem antiquated, but the reasoning remains identical to that of a jury: uncorrelated errors pooled together are more effective than any single fatigued judge. Random Forest [10] dominates at packet-rate trees, XGBoost [7] pursues sequential residuals, CatBoost [9] internalizes categorical peculiarities, and a solitary Decision Tree maintains intellectual honesty in the ensemble. Averaging posterior probabilities mitigates occasional outliers from the 0.01% minority classes that a single LightGBM run might miss. The macro-F1 of the ensemble is bootstrapped 1000 times, producing [99.31, 99.43], confirming that the observed performance is unlikely due to chance [23].

LightGBM: the "Good-Enough" Sweet Spot for Edge Silicon:

Gateway retraining every hour cannot afford six-minute delays; 82 s of single-core CPU usage is preferable to 391 s, and the 0.34% drop in accuracy corresponds to roughly 2,500 flows in our test dataset, negligible for a million-flow evaluation. The working set, with 30 features and 255 histogram bins, occupies only 30 × 255 bins, no larger than a DHCP daemon on a Raspberry Pi powered via PoE [8]. With 6.1 W wall power [21], the energy cost is under 0.02¢ per million predictions—cheaper than the status LED that flashes per packet arrival. Lightweight machine learning implementations on resource-constrained edge devices enable real-time intrusion detection without cloud dependency [25].

Where the Model Still Bleeds:

Web-layer attacks reach 67% F1, enough to raise a red flag, yet still require hand-off to a WAF for final mitigation. Brute-force intrusions achieve 81% recall and 55% precision, resulting in approximately one false alarm per two true positives; optimal recalibration of the operating point between cost-sensitive loss and human oversight is recommended [11].

Limitations and Future Work:

Dataset Shift: The train/test split is random rather than chronological. Firmware updates and diurnal traffic patterns can skew feature distributions. Production deployment will require online learning or routine retraining.

Adversarial Evasion: Stress-testing the pipeline with FGSM/PGD perturbations is pending; an attacker aware of the 30-feature input could manipulate inter-arrival times or fragment packets to bypass detection. Future work includes adversarial training or input randomization [26]. Adversarial robustness of ensemble methods remains an open challenge for practical IoT intrusion detection systems [27].

Hierarchical Granularity**:** SOC playbooks require sub-types like SYN-flood within DDoS. Using two-stage classifiers—family first, variant second—could maintain 99% macro-F1 while providing the required granularity.

Take-Away Slider for Practitioners:

If maximum accuracy is required Deploy the voting ensemble, paying 300 s training premium.

If energy efficiency is required Stop at LightGBM; 99% macro-F1 is still better than published baselines and can run on a $30 gateway.

In any case, the 30-feature set, 8-class taxonomy, and one-line training script are publicly available and ready for deployment before the next firmware release.

Conclusion:

We demonstrated that the primary challenge in IoT intrusion detection is extreme class imbalance, where rare attacks are nearly statistically rare events. By consolidating CIC-IoT-2023's 34 fragmented labels into eight behaviorally coherent families and pruning the feature space to 30 gateway-friendly variables, we transformed a skewed 46-million-flow dataset into a fair task. A soft-voting ensemble of four off-the-shelf learners achieves 99.37% macro-F1 while inferring in 3.9 µs per packet on a Raspberry Pi 4, fast enough for real-time edge deployment. For higher speed, a tuned LightGBM reaches 99% after 82 s of training, providing a live trade-off between last-drop accuracy and power efficiency.

The pipeline is intentionally simple: no custom layers, no TPU rentals, only open-source code ready for deployment prior to the next firmware update. These results outperform every published baseline on the same dataset by up to 6.6 F1 points, demonstrating that disciplined feature selection and majority voting can surpass more complex architectures in highly imbalanced scenarios. We release the exact splits, 30-feature extractor, and one-line training script so hospitals, factories, or retail networks can deploy the detector on a $30 edge device and begin catching botnets immediately.

A 99% F1 score is not a finish line but a starting point. Future work will focus on robustness to chronological drift, adversarial perturbations, and hierarchical sub-labels without compromising the speed that enables edge deployment. Researchers who can improve on the 3.9 µs inference or further reduce the feature set while maintaining accuracy are encouraged to submit pull requests for leaderboard updates— timely deployment is critical.

References:

[1] F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, "Internet of Things security: A survey," *J. Netw. Comput. Appl.*, vol. 88, pp. 10–28, Jun. 2017, doi: 10.1016/J.JNCA.2017.04.002.

[2] Akash Dogra, "CIC IoT dataset 2023," *Kaggle*, 2023, [Online]. Available: https://www.kaggle.com/datasets/akashdogra/cic-iot-2023

[3] A. Thakkar and R. Lohiya, "A Review on Machine Learning and Deep Learning Perspectives of IDS for IoT: Recent Updates, Security Issues, and Challenges," *Arch. Comput. Methods Eng.*, vol. 28, no. 4, pp. 3211–3243, Jun. 2021, doi: 10.1007/s11831-020-09496-0.

[4] Hongyu Liu, Bo Lang, "Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey," *Appl. Sci.*, vol. 9, no. 20, p. 4396, 2019, [Online]. Available: https://www.mdpi.com/2076-3417/9/20/4396

[5] Mohamed Amine Ferrag, Leandros Maglaras, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *J. Inf. Secur. Appl.*, vol. 50, p. 102419, 2020, [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S2214212619305046

[6] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," *ICISSP 2018 - Proc. 4th Int. Conf. Inf. Syst. Secur. Priv.*, vol. 2018-January, pp. 108–116, 2018, doi: 10.5220/0006639801080116.

[7] Tianqi Chen, Carlos Guestrin, "XGBoost: A Scalable Tree Boosting System," *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2016, [Online]. Available: https://dl.acm.org/doi/10.1145/2939672.2939785

[8] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," *31st Conf. Neural Inf. Process. Syst.*, 2017, [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf

[9] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika

Dorogush, Andrey Gulin, "CatBoost: unbiased boosting with categorical features," *arXiv:1706.09516*, 2017, [Online]. Available: https://arxiv.org/abs/1706.09516

[10] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001, doi: 10.1023/A:1010933404324/METRICS.

[11] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/JAIR.953.

[12] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, "Deep Learning Approach for Intelligent Intrusion Detection System," *IEEE Access*, vol. 7, 2019, [Online]. Available: https://ieeexplore.ieee.org/document/8681044

[13] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, "Network Traffic Classifier With Convolutional and Recurrent Neural Networks for Internet of Things," *IEEE Access*, vol. 5, 2017, [Online]. Available: https://ieeexplore.ieee.org/document/8026581

[14] Y. Meidan *et al.*, "N-BaIoT-Network-based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Comput.*, vol. 17, no. 3, pp. 12–22, Jul. 2018, doi: 10.1109/MPRV.2018.03367731.

[15] M. Z. Alom, V. Bontupalli, and T. M. Taha, "Intrusion detection using deep belief networks," *Proc. IEEE Natl. Aerosp. Electron. Conf. NAECON*, vol. 2016-March, pp. 339–344, Mar. 2016, doi: 10.1109/NAECON.2015.7443094.

[16] S. Rezvy, Y. Luo, M. Petridis, A. Lasebae, and T. Zebin, "An efficient deep learning model for intrusion classification and prediction in 5G and IoT networks," *2019 53rd Annu. Conf. Inf. Sci. Syst. CISS 2019*, Apr. 2019, doi: 10.1109/CISS.2019.8693059.

[17] David H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, 1992, [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0893608005800231

[18] Bayu Adhi Tama, Kyung Hyune Rhee, "An Integration of PSO-based Feature Selection and Random Forest for Anomaly Detection in IoT Network," *MATEC Web Conf.*, vol. 159, 2018, [Online]. Available: https://www.researchgate.net/publication/324109443_An_Integration_of_PSO-based_Feature_Selection_and_Random_Forest_for_Anomaly_Detection_in_IoT_Network

[19] Ansam Khraisat, Iqbal Gondal, Peter Vamplew, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 20, 2019, [Online]. Available: https://link.springer.com/article/10.1186/s42400-019-0038-7

[20] V. C. Riccardo Lazzarini, Huaglory Tianfield, "Federated Learning for IoT Intrusion Detection," *AI*, vol. 4, no. 3, pp. 509–530, 2023, doi: https://doi.org/10.3390/ai4030028.

[21] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," *Proc. - IEEE Symp. Secur. Priv.*, pp. 305–316, 2010, doi: 10.1109/SP.2010.25.

[22] Jonathan Lundqvist, Anel Hadzic, "Lightweight Machine Learning Models for Intrusion Detection on IoT Devices," *Nor. IKT-konferanse Forsk. og utdanning*, vol. 37, no. 3, 2025, doi: 10.5324/jrxdjb92.

[23] Markus Ring, Sarah Wunderlich, Deniz Scheuring, Dieter Landes, Andreas Hotho, "A survey of network-based intrusion detection data sets," *Comput. Secur.*, vol. 86, pp. 147–167, 2019, [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S016740481930118X

[24] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, "A detailed investigation and analysis of using machine learning techniques for intrusion detection," *IEEE Commun. Surv. Tutorials*, vol. 21, no. 1, pp. 686–728, Jan. 2019, doi: 10.1109/COMST.2018.2847722.

[25] T. T. Khoei, G. Aissou, W. C. Hu, and N. Kaabouch, "Ensemble Learning Methods for Anomaly Intrusion Detection System in Smart Grid," *IEEE Int. Conf. Electro Inf. Technol.*, vol. 2021-May, pp. 129–135, May 2021, doi: 10.1109/EIT51626.2021.9491891.

[26] G. Apruzzese, M. Colajanni, L. Ferretti, A. Guido, and M. Marchetti, "On the effectiveness of machine and deep learning for cyber security," *Int. Conf. Cyber Conflict, CYCON*, vol. 2018-May, pp. 371–389, Jul. 2018, doi: 10.23919/CYCON.2018.8405026.

[27] Mohammed Rauf Ali Khan, Abdulaziz Y. Barnawi, "Lightweight Quantized XGBoost for Botnet Detection in Resource-Constrained IoT Networks," *IOT*, vol. 6, no. 4, p. 70, 2025, doi: https://doi.org/10.3390/iot6040070.