Intelligent Intrusion Detection System for Cybersecurity of Water Utilities

Sebastian Mesca
Automation and Industrial Informatics
University Politehnica of Bucharest
Bucharest, Romania
sebastian.mesca@stud.acs.upb.ro

Grigore Stamatescu
Automation and Industrial Informatics
University Politehnica of Bucharest
Bucharest, Romania
grigore.stamatescu@upb.ro

***Abstract*—Real-time cybersecurity of critical infrastructures that include multiple networked automation systems represents an important challenge for the assurance of modern societal functions. In particular water supply and treatment facilities have to operate at high availability and efficiency parameters, with direct impact on public health in the case of performance degradation or unscheduled down time due to network attacks. We present a machine learning (ML)-based approach to detect malicious activities in operational control networks of water utilities. The system accounts for the particularities of the industrial communication protocols used for process control of this critical sector and presents a comparison between enhanced random forest models (XGBoost), hybrid neural network architectures (CNN-MLP) and logistic regression, as reference baseline model. Binary classification results, evaluated on the popular SWaT dataset, show that ML methods can extend intrusion detection system capabilities for accurate attack detection.**

***Index Terms*—intrusion detection system, machine learning, cybersecurity, water utilities, anomaly detection.**

I. Introduction

Industrial Control Systems (ICS) comprise networks of intelligent automation devices tasked with the essential functions of monitoring, controlling, and managing (cyber-)physical and operational processes across critical industries, such as energy production, water treatment [1], transportation, manufacturing and the built environment [2], their primary objective being to maintain continuous operation at appropriate parameters and uphold the overall resilience of vital sectors. The present generation of ICSs is undergoing accelerated modernization, characterized by the integration of Industrial Internet of Things (IIoT) components and mechanisms into existing architectures. This modernization aims to improve the quality and accuracy of sensor-generated data and to significantly increase overall system efficiency [3]. However, the IT/OT convergence has introduced new cybersecurity vulnerabilities into conventional control systems, primarily due to the expanded attack surface resulting from increased connectivity and additional access points within heterogeneous IT/OT network environments.

The direct coupling of ICS to physical processes substantially increases the potential severity of cyberattack consequences, as evidenced by historical incidents. For instance, the Stuxnet malware caused physical destruction of uranium enrichment centrifuges in Iran [4], while BlackEnergy Advanced Persistent Threat (APT) Distributed Denial of Service (DDoS) attacks severely disrupted power grid operations in Ukraine beginning in 2015 [5].

Recent cyberattacks highlight that the security of ICS is inherently connected with physical systems, particularly in critical sectors such as power generation [6], petroleum and petrochemical industries and water and wastewater systems(WWS). Unlike traditional cyberattacks on IT systems that predominantly cause financial and/or data loss for organizations and individuals, security breaches exploited in ICS can trigger catastrophic consequences with direct effects in the physical world, including nationwide power blackouts, nuclear plant explosions, or contamination of potable water [7].

ICSs exhibit highly consistent communication patterns characterized by a repetitive sequence of a limited set of read and write commands [8]. This regularity results in command sequences with fixed device addresses and uniform packet lengths, a predictable pattern exploitable by machine learning(ML) algorithms for modeling normal behavior and identifying anomalies [9].

The main contribution of our work in this context is argued to be the comparative implementation of three machine learning models of different complexity and structure which are evaluated on the Secure Water Treatment (SWaT) dataset based on their performance in the binary classification task for ICS network attack detection.

The scope of this study is intentionally baseline and empirical: we benchmark representative model families under a unified preprocessing pipeline on SWaT to surface deployment-relevant trade-offs, most notably the precision-recall tension in ICS traffic with highly regular patterns. The contribution is therefore a clear and reproducible point of reference for recall-aware tuning and engineering in future IDS work for water utilities, rather than proposing a new architecture.

To clearly understand the quality of model predictions, several metrics are considered. The **F1-score** combines precision and recall into a single metric and represents their harmonic mean, effectively balancing false positives and false negatives. **The Area Under the Curve (AUC)** summarizes the **Receiver Operating Characteristic (ROC) curve**, illustrating the trade-off between sensitivity (true positive rate) and specificity (true negative rate) across all possible classification thresholds. A higher AUC value indicates a better-performing

model. Finally, the ROC curve itself visually compares the rate of correctly detected attacks (true positives) against the rate of false alarms (false positives), assisting in the informed selection of optimal detection thresholds.

II. Industrial Control Network Structure and IDS Taxonomy

Most ICSs follow the *Purdue Enterprise Reference Architecture (PERA)* described in [10].The Purdue Model rigorously partitions the ICS network into discrete, functionally homogeneous zones, each aligned with distinct operational and security necessities. Figure 1 presents the layout diagram for PERA, with the key layer, the Demilitarized Zone (DMZ) that ensures IT and OT network separation at level 3.5. The main zones are detailed as follows.

The **Enterprise Zone** (IT network) consolidates conventional IT assets-ranging from logistic business systems to the corporate network infrastructure.

The **Demilitarized Zone (DMZ)** functions as a hardened intermediary, precisely regulating data exchange between the Control Zone and the Enterprise Zone, thus ensuring robust segregation between IT and OT domains.

The **Control Zone** (or OT network) embodies the critical systems and instrumentation required for real-time monitoring, control, and maintenance of automated industrial processes, and is methodically subdivided into four hierarchical tiers:

- **Level 0** encompasses sensors and actuators that interact directly with physical processes;
- **Level 1** includes intelligent devices such as Programmable Logic Controllers (PLCs), Intelligent Electronic Devices (IEDs), and Remote Terminal Units (RTUs);
- **Level 2** integrates control systems like Human-Machine Interfaces (HMIs), alarm systems, and control room workstations;
- **Level 3** consolidates manufacturing operations systems that rigorously manage plant processes to achieve the desired outputs.

Communication between Levels 2 and 3 and the Enterprise Zone is strictly mandated to pass through the DMZ. Additionally, the **Safety Zone** comprises specialized devices and systems tasked with the proactive detection of anomalies and the prevention of hazardous failures within the ICS. Despite the stringent requirement, prescribed by the Purdue Model, that all traffic between OT and IT networks be funneled through the DMZ, practical deployments often diverge from this protocol, primarily due to integration complexities and insufficient prioritization of security during the system development phase. This deviation inevitably exposes the critical OT network to heightened risks of cyber attacks. The increase in advanced cyber threats targeting Industrial Control Systems (ICS) has elevated their security to a critical research focus, particularly following the discovery of Stuxnet, which marked a turning point in cyber-physical threat awareness. Intrusion Detection Systems (IDS) have since been acknowledged as a

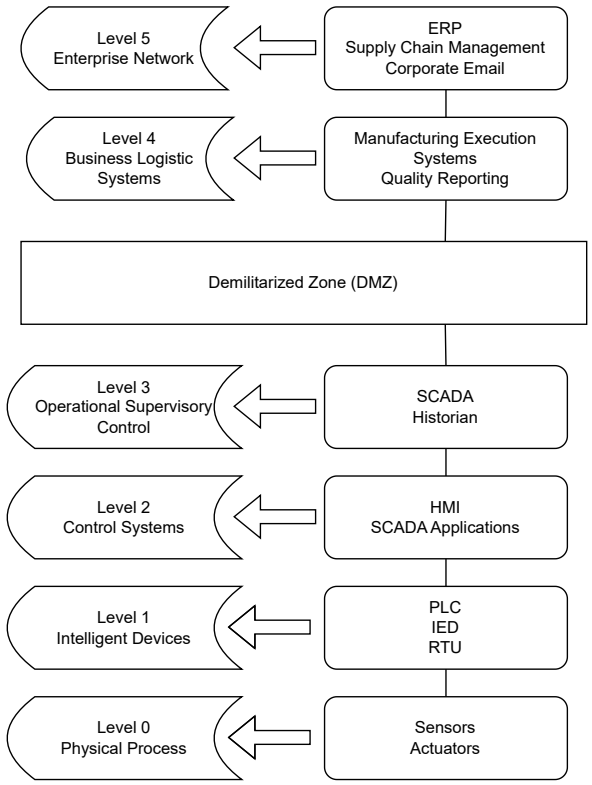

Fig. 1. Purdue Reference Model: ICS network architecture showing IT/OT segmentation across Levels 0-5

fundamental line of defense for safeguarding ICS infrastructures. However, traditional IDS solutions, originally designed for conventional IT environments, are not inherently equipped to address the operational constraints, real-time requirements, and high availability demands specific to ICS environments. As a result, their applicability in industrial contexts remains limited. Although research efforts toward ICS-tailored IDS have intensified in recent years, the domain continues to lack clear definitions, standardized methodologies, and universally accepted evaluation benchmarks. [11]

Existing IDSs for ICSs can be classified according to various distinguishing criteria. When categorized based on the underlying detection methodology, IDS methodologies are broadly divided into signature-based and anomaly-based paradigms.

In ICS, intrusion detection spans **signature-based** and **anomaly-based** methods and is deployed as **host-** or **network-based** (HIDS/NIDS), depending on visibility and latency constraints [12], [13]. Signature methods typically yield high precision for known behaviors, whereas anomaly methods can flag previously unseen patterns at the cost of higher false-alarm rates [12]. In this work we focus on a network-based IDS (NIDS) setting with supervised learning on SWaT network

logs, reflecting passive OT monitoring needs and limited host instrumentability in industrial environments.

III. Methodology and Data

An important challenge in the application and evaluation of supervised machine learning and deep learning techniques in ICS cybersecurity is the scarcity of accurately labeled and representative datasets. Real-world cyber-attacks targeting critical infrastructure are relatively rare, resulting in sparse data points for malicious activities. [14] Even when operational or testbed data are available, the process of generating reliable ground-truth labels is resource intensive and costly, demanding substantial time and domain expertise to analyze complex system behaviors and differentiate true attacks from noise or operational anomalies. [15] This combined difficulty in obtaining sufficient and accurately labeled attack data forms a major bottleneck for research, highlighting the critical importance of realistic, publicly accessible testbed datasets. This paper compares the intrusion detection performance achieved by training three machine learning models on the SWaT dataset [16]. The specific models evaluated were:

- An **XGBoost classifier**, configured for GPU acceleration.
- A hybrid TensorFlow/Keras implementation of a **Neural Network employing both 1D Convolutional and Dense** (MLP) **layers** (CNN-MLP).
- A standard **Logistic Regression** model - used as baseline for reporting the improvements achieved by more complex models.

The SWaT testbed architecture realistically models a modern water treatment plant, scaled down but fully operational. It implements a sequential six-stage purification process, involving raw water intake, chemical pre-treatment, ultrafiltration, UV dechlorination, reverse osmosis, and permeate storage. Control and supervision rely on ICS components, including dedicated PLCs governing each stage, HMIs for operator interaction, and a central SCADA system connected to a Historian for data logging. Communication is structured in a layered network: Level 1 utilizes a star topology connecting the SCADA system to the stage PLCs, while Level 0 consists of ring networks linking PLCs to their respective sensors and actuators.

The SWaT dataset was generated through a data collection process over 11 days of continuous, 24/7 operation of the water treatment testbed. To establish a baseline reflecting realistic system startup and stabilization, the process started with all tanks emptied and ran for seven consecutive days in normal conditions without any faults or cyber-physical attacks. Subsequently, during the final four days, the system was subjected to a series of 36 distinct, pre-defined attack scenarios targeting various sensors and actuators across different stages. Throughout the 11-day period, data was logged at a one-second granularity. Two primary types of data were collected: detailed physical process information recorded by a central Historian, and network traffic data captured between the SCADA system and the PLCs. Detailed logging of each attack, including exact timing and targets, was essential for the accurate manual labeling used to distinguish normal from attack states in both datasets.

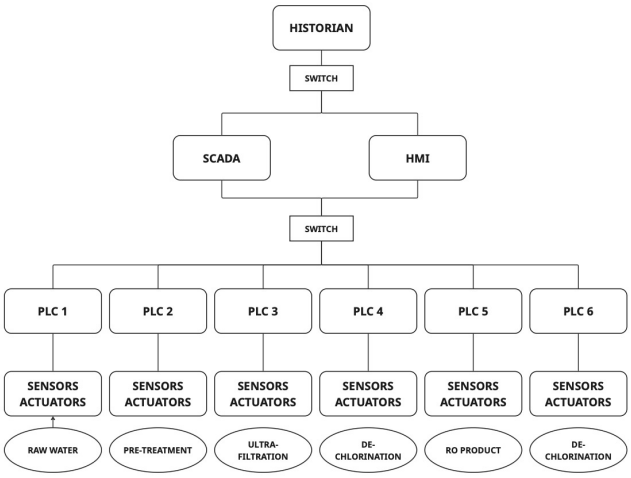

Fig. 2. Secure Water Treatment testbed architecture

The SWaT dataset provides ground truth through a `Tag` column, where `0` signifies normal operation and `1` indicates an attack period, as determined during the original data collection. For this study, a specific subset of files was selected to ensure representation of both conditions while managing computational resources. Due to significant memory constraints encountered when attempting to load and concatenate the fully processed data from all available files, the final dataset used for all model training and evaluation was constructed by randomly sampling 7 verified attack files and 8 normal files, selected using a fixed random seed (`RANDOM_STATE = 42`) for reproducibility.

A. Data Preprocessing and Feature Engineering

Consistent preprocessing was applied to the selected SWaT network log data to generate a unified numerical feature set for all evaluated models. Key steps included:

- **Feature Engineering:** To capture temporal dynamics in the sequence of log entries, the inter-record interval (`time_delta_sec`) was computed, reflecting the time elapsed between successive records. Recent protocol behavior was characterized by calculating the rolling mean and standard deviation over a five-record window for the `service` and `s_port` fields, enabling the model to incorporate short-term fluctuations in network activity.
- **Missing Value Imputation**: Missing numerical values were imputed with zero, while missing categorical/text entries were filled with a placeholder string (`__NaN__`).
- **Numerical Scaling**: All base numerical features along with the engineered time and rolling window features were scaled to have zero mean and unit variance using StandardScaler, which was fitted on a sample of the data.
- **Categorical Encoding**: All remaining non-numerical columns (including IP addresses, protocol names, device tags, and descriptive text) were converted into

32-dimensional numerical vectors using feature hashing (HashingVectorizer).

- **Final Vector Creation**: The scaled numerical features and the multiple 32-dimensional hashed vectors were concatenated (np.hstack) to create the final feature matrix (396 dimensions) used as input for the models.

This entire pipeline was executed chunk-by-chunk per input file, saving intermediate results before final concatenation, as required by memory constraints with the dataset size.

B. Model Training and Implementation

Following the preprocessing pipeline described in subsection A, the resulting feature-engineered datasets were prepared for model training and evaluation.

1) XGBoost Model: An XGBoost classifier, implemented via the `xgboost` Python library (`xgb.XGBClassifier`), was employed as the first modeling approach. This model operates by constructing an ensemble of decision trees (f_k) additively, leveraging gradient boosting principles. Key configurations included setting the objective to `binary:logistic` for this classification task and enabling GPU acceleration using `device='cuda'` and `tree_method='hist'`. The model's prediction probability for a given sample x results from applying the sigmoid function (σ) to the sum of outputs from all K trees:

$$\hat{y}_p = P(y = 1 \mid x) = \sigma\left(\sum_{k=1}^{K} f_k(x)\right) \tag{1}$$

where the sigmoid function $\sigma(z)$, which converts any real-valued input z into a value between 0 and 1, is defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{2}$$

Trees are added sequentially to minimize a regularized objective function based on the logistic loss gradient. Training was performed using the `.fit()` method on the prepared training data $(X_{\text{train}}, y_{\text{train}})$, utilizing the test set $(X_{\text{test}}, y_{\text{test}})$ as the evaluation set for early stopping (`early_stopping_rounds=50`) based on the AUC metric (`eval_metric='auc'`).

2) Keras Hybrid CNN-MLP Model: The second model evaluated was a hybrid architecture, also implemented in `tensorflow.keras`, combining an initial 1D CNN stage for feature extraction with a subsequent deeper Multi-Layer Perceptron (MLP) head for classification.

Input features were reshaped and passed through two `Conv1D` blocks. The first block used 64 filters with a kernel size of 3 and ReLU activation, followed by a `MaxPooling1D` layer with pool size 2 and a Dropout layer with a rate of 0.25. The second block applied 32 filters with the same kernel size and activation function, followed again by `MaxPooling1D` (pool size = 2) and Dropout (0.25). After flattening the resulting convolutional feature maps, the MLP section consisted of two Dense layers of 64 and 32 neurons respectively, each employing ReLU activation, interspersed with a substantial Dropout of 0.5 before the final output neuron with sigmoid activation for binary prediction.

Formally, the convolutional and dense layers of the model were defined by the general equation:

$$h = \text{activation}(Wx + b) \tag{3}$$

This equation represents a generic dense (fully connected) neural network layer, where W is the weight matrix, x is the input vector, b is the bias vector, and h is the output after applying an activation function. Specifically, activation functions employed are:

- **ReLU activation** defined as:

$$\text{ReLU}(z) = \max(0, z) \tag{4}$$

- **Sigmoid activation** used in the final prediction layer:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{5}$$

The model minimizes the binary cross-entropy loss function given by:

$$L(y, \hat{y}_p) = -\left[y \log(\hat{y}_p) + (1 - y) \log(1 - \hat{y}_p)\right] \tag{6}$$

This binary cross-entropy loss L measures the difference between true labels y (0 or 1) and predicted probabilities $\hat{y}_p$.

The choice of activation functions was guided by their suitability for different roles within neural network architectures. The ReLU activation function was selected due to its computational efficiency, robustness against vanishing gradients, and its ability to introduce non-linearity effectively, thus facilitating deeper networks and enhancing feature extraction in convolutional and dense hidden layers. For the final classification layer, the sigmoid activation function was chosen for its inherent suitability in binary classification, as it transforms raw output logits into probabilistic predictions bounded between 0 and 1. This combination ensures efficient convergence during training and yields accurate and interpretable outputs aligned with the objective of distinguishing between normal and attack instances in ICS network traffic.

The hybrid CNN-MLP was compiled with the Adam optimizer and trained using identical procedures (`.fit()` method, early stopping based on `val_auc`) to allow direct comparison based on architectural differences.

3) Logistic Regression Model: In addition to the tree-based and neural-based approaches, a standard Logistic Regression (LR) model was implemented using the `scikit-learn` library (`sklearn.linear_model.LogisticRegression`). This model predicts the probability of an attack by applying a sigmoid transformation to a linear combination of the engineered features x:

$$\hat{y}_p = P(y = 1 \mid x) = \sigma\big(w^T x + b\big) \tag{7}$$

where w is the learned weight vector, b the bias term, and the sigmoid function $\sigma(z)$ is defined as:

$$\sigma(z) = \frac{1}{1+e^{-z}}. \tag{8}$$

$$\hat{y}_p = \sigma(z) = \frac{1}{1+e^{-z}}. \tag{9}$$

Training proceeds by maximizing the likelihood (or equivalently minimizing the binary cross-entropy loss)

$$L(y, \hat{y}_p) = -\big[y \log(\hat{y}_p) + (1-y)\log(1-\hat{y}_p)\big], \tag{10}$$

subject to a regularization penalty to control model complexity and prevent overfitting.

IV. Results

Table I compares the performance of Logistic Regression, XGBoost, and the Hybrid CNN-MLP models. The results demonstrate that both the XGBoost and Hybrid CNN-MLP models considerably outperform Logistic Regression across all measured metrics. Logistic Regression achieves a relatively low ROC AUC of 0.7560 and accuracy of 0.6945, highlighting its inadequacy for effective detection in the SWaT dataset. XGBoost achieves a significantly higher ROC AUC (0.8840) and accuracy (0.7817), with excellent precision (0.9540), though it has lower recall (0.4735). The Hybrid CNN-MLP model provides similar results, with a ROC AUC of 0.8748 and accuracy of 0.7765, also exhibiting high precision (0.9418) but limited recall (0.4665).

TABLE I
Test-Set Performance Comparison

Model	Acc.	AUC	Pre.	Rec.	F_1	Spec.	FPR
XGBoost	0.782	0.8840	0.954	0.474	0.634	0.985	0.015
Hybrid CNN-MLP	0.776	0.8748	0.942	0.467	0.624	0.981	0.019
Logistic Regression	0.695	0.7560	0.705	0.398	0.510	0.812	0.188

While both the XGBoost and CNN-MLP models achieve high precision and low false-positive rates, their performance is constrained by relatively low recall values (0.47 and 0.46, respectively), indicating a significant number of undetected attack instances (false negatives). This outcome represents a critical limitation in the context of intrusion detection for ICS environments, where undetected attacks may pose severe operational risks and threats to public safety. The lower recall indicates the models' difficulty in detecting subtle attack patterns that closely resemble normal operational behavior, particularly given the high similarity and repetitive nature of ICS network traffic patterns. This implies that the learned feature representations may be insufficiently sensitive to certain types of anomalies present in the dataset. To address the recall deficit within the present framework, a direct next step is to adopt cost-sensitive objectives that penalize missed attacks more than false alarms. Concretely, we would upweight attack class errors during training, encouraging the learner to prioritize sensitivity to minority patterns. This modification preserves the models and features reported here but shifts the decision boundary toward higher recall; we leave its empirical evaluation to future work.

These observations are further validated by the confusion matrix counts shown in Table II. Logistic Regression exhibits substantial misclassification with high false-negative (352,735) and considerable false-positive counts (183,662). XGBoost and the Hybrid CNN-MLP significantly reduce false-positive counts to 13,381 and 21,139, respectively, but they still show a notable number of false negatives (308,843 and 391,201 respectively), indicating difficulties in identifying all attack instances. Figure 2 illustrated the relative confusion matrix values for the neural network model.

To mitigate this limitation, future research should explore advanced training strategies, such as cost-sensitive learning methods designed to explicitly penalize false negatives more heavily, thus emphasizing the detection of attacks. Furthermore, adaptive or dynamic threshold tuning approaches could be employed, systematically balancing the precision-recall trade-off to achieve better recall without substantially increasing false alarms.

TABLE II
Confusion Matrix Counts

Model	TP	FN	TN	FP
XGBoost	277749	308843	876085	13381
Hybrid CNN-MLP	342038	391201	1090694	21139
Logistic Regression	233443	353149	792506	183662

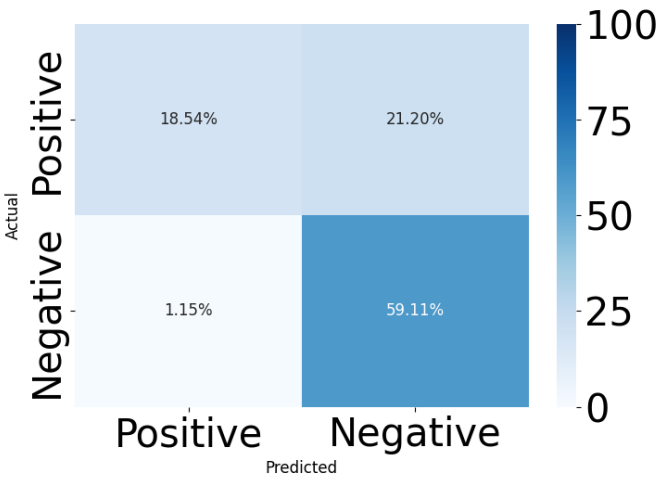

Fig. 3. Confusion Matrix with Relative Values for the Neural Network-based Model

Figure 4 and Figure 5 visually illustrate these distinctions through the Precision-Recall (PR) and Receiver Operating curves. The curves for XGBoost and the Hybrid CNN-MLP are closely overlapping, reflecting very similar performance characterized by high precision across low-to-medium recall thresholds. This suggests these models excel at minimizing false alarms but struggle to maintain recall as classification thresholds are relaxed. In contrast, Logistic Regression consistently shows lower precision at nearly every recall level,

underscoring its limitations in reliably identifying attacks without generating excessive false positives.

Practically, XGBoost emerges as a slightly superior option given its marginally higher ROC AUC, accuracy, and precision, beneficial when false-positive reduction is prioritized. However, both XGBoost and Hybrid CNN-MLP models must be improved in terms of recall to effectively reduce missed detections. Logistic Regression, given its significantly poorer performance, remains unsuitable for practical deployment in critical ICS security contexts.

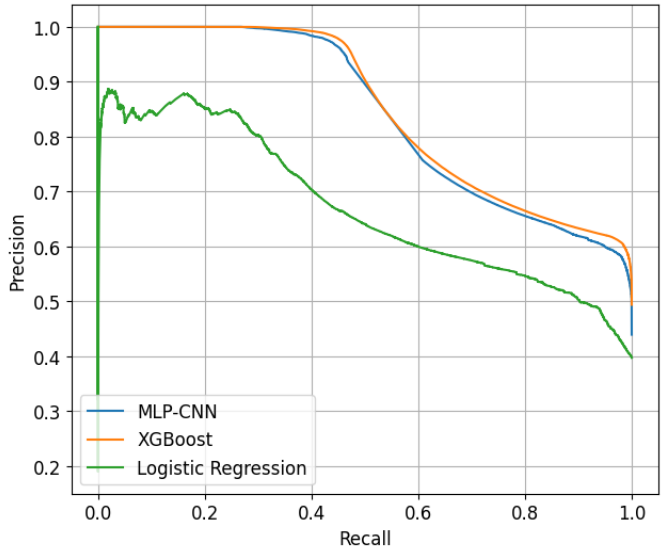

Fig. 4. Precision-Recall Curve Comparison of Logistic Regression, XGBoost, and Hybrid CNN-MLP Models

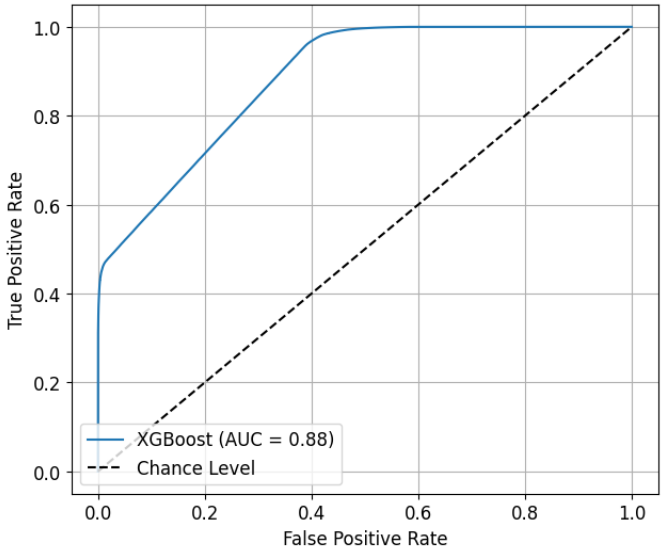

Fig. 5. Receiver Operating Characteristic (ROC) Curve for the XGBoost Model

V. Conclusion

We presented an approach to classify attack patterns in wastewater utility plants by combining industrial network logs with machine learning models. Despite promising precision and low false-positive rates, the relatively low recall highlights the need for enhanced detection sensitivity. The approach shows promise for online anomaly detection of connected automation systems. Ongoing work is focused on extensive evaluation of the proposed models under explainable AI and feature pre-processing requirements.

References

[1] M. Ibrahim and A. Al-Wadi, "Wastewater treatment plant security analysis," in *2022 14th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, pp. 1–6, 2022.

[2] G. Stamatescu, I. Stamatescu, N. Arghira, and I. Făgărăşan, "Cybersecurity perspectives for smart building automation systems," in *2020 12th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, pp. 1–5, 2020.

[3] K. Stouffer, J. Falco, K. Scarfone, *et al.*, "Guide to industrial control systems (ics) security," *NIST special publication*, vol. 800, no. 82, pp. 16–16, 2011.

[4] S. McLaughlin, C. Konstantinou, X. Wang, L. Davi, A.-R. Sadeghi, M. Maniatakos, and R. Karri, "The cybersecurity landscape in industrial control systems," *Proceedings of the IEEE*, vol. 104, no. 5, pp. 1039–1057, 2016.

[5] M. Geiger, J. Bauer, M. Masuch, and J. Franke, "An analysis of black energy 3, crashoverride, and trisis, three malware approaches targeting operational technology systems," in *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, pp. 1537–1543, IEEE, 2020.

[6] G. Kavallieratos, A. Amro, V. Gkioulos, G. Stamatescu, K. Rantos, T. Lagkas, K. Demertzis, F. Paterakis, A. Lekidis, C. Dalamagkas, I. Kotsiuba, and S. Katsikas, "Best-practices-based training for improving cybersecurity in power grids," in *Computer Security. ESORICS 2024 International Workshops*, pp. 344–359, Springer Nature, 2025.

[7] M. R. Asghar, Q. Hu, and S. Zeadally, "Cybersecurity in industrial control systems: Issues, technologies, and challenges," *Computer Networks*, vol. 165, p. 106946, 2019.

[8] M. Ibrahim, R. Elhafiz, and A. Al-Wadi, "Reinforcement learning-based attack graph analysis for wastewater treatment plant," *IEEE Transactions on Industry Applications*, vol. 59, no. 6, pp. 7858–7867, 2023.

[9] D. Bhamare, M. Zolanvari, A. Erbad, R. Jain, K. Khan, and N. Meskin, "Cybersecurity for industrial control systems: A survey," *computers & security*, vol. 89, p. 101677, 2020.

[10] T. J. Williams, "The purdue enterprise reference architecture," *IFAC Proceedings Volumes*, vol. 26, no. 2, pp. 559–564, 1993.

[11] Y. Hu, A. Yang, H. Li, Y. Sun, and L. Sun, "A survey of intrusion detection on industrial control systems," *International Journal of Distributed Sensor Networks*, vol. 14, no. 8, p. 1550147718794615, 2018.

[12] M. Kaouk, J.-M. Flaus, M.-L. Potet, and R. Groz, "A review of intrusion detection systems for industrial control systems," in *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*, pp. 1699–1704, IEEE, 2019.

[13] A. Gangwar and S. Sahu, "A survey on anomaly and signature based intrusion detection system (ids)," *International Journal of Engineering Research and Applications*, vol. 4, no. 4, pp. 67–72, 2014.

[14] M. A. Umer, K. N. Junejo, M. T. Jilani, and A. P. Mathur, "Machine learning for intrusion detection in industrial control systems: Applications, challenges, and recommendations," *International Journal of Critical Infrastructure Protection*, vol. 38, p. 100516, 2022.

[15] A. M. Koay, R. K. L. Ko, H. Hettema, and K. Radke, "Machine learning in industrial control system (ics) security," *Journal of Intelligent Information Systems*, vol. 60, no. 2, pp. 377–405, 2023.

[16] J. Goh, S. Adepu, K. N. Junejo, and A. Mathur, "A dataset to support research in the design of secure water treatment systems," in *Critical Information Infrastructures Security: 11th International Conference, CRITIS 2016, Paris, France, October 10–12*, pp. 88–99, Springer, 2017.