International Advance Journal of Engineering, Science and Management (IAJESM)
Multidisciplinary, Indexed, Double Blind, Open Access, Peer-Reviewed, Refereed-International Journal.
SJIF Impact Factor =7.938, July-December 2024, Submitted in July 2024, ISSN -2393-8048

Next-Generation Smart Security for Wireless Sensor Networks: Integrating Machine Learning for Threat Detection and Prevention

Deepak Kumar (Research Scholar), Department of Computer Science, NIILM University, Kaithal (Haryana)
Dr. Mukesh Kumar Rana (Professor), Department of Computer Science, NIILM University, Kaithal (Haryana)

Abstract

Wireless Sensor Networks (WSNs) form an integral part of the modern Internet of Things (IoT) infrastructure and are used in healthcare, agriculture, smart cities and military applications. However, their open wireless medium, limited processing power and dispersed architecture makes them very vulnerable to a wide range of cyber assaults including Sinkhole, Blackhole, Flooding, Sybil, Wormhole and Selective Forwarding attacks. These resource-constrained environments cannot rely solely on traditional cryptographic and rule-based security approaches. In this research, we offer a next generation smart security framework: Trust Sense-WSN, which combines several machine learning (ML) methods such as Random Forest, XGBoost, LightGBM, and Long Short-Term Memory (LSTM) networks in a two-tier detection architecture. The framework is assessed on the publicly available WSN-DS dataset (374,661 records), resulting in a detection accuracy of 99.81%, a false positive rate of less than 0.7%, and a per-inference energy consumption reduction of 34% in comparison to a standard single-model baseline. A new Network Dependability Score (NDS) module providing a real-time assessment of network trustworthiness. Experimental results illustrate the superiority of the proposed framework over all individual ML baselines and approaches from the current literature and hence its suitability for practical deployment in resource-constrained WSN situations.

Keywords: ***Wireless Sensor Networks, Machine Learning, Intrusion Detection System, Sinkhole Attack, Blackhole Attack, Random Forest, LSTM, Network Dependability***

1. INTRODUCTION

Wireless sensor networks (WSNs) are composed of low-power sensor nodes that measure temperature, pressure, humidity, motion or chemical concentrations and wirelessly send this information to a central base station. Today, WSNs are ubiquitous: in hospitals for patient monitoring, in fields for precision agriculture, on the floors of factories, for perimeter surveillance in military applications, and in smart city infrastructure. The global WSN market is expected to reach over USD 90 billion by 2027 [1] indicating its huge importance in the modern culture. However, every sensor node is a possible entry point for an attacker. WSN nodes interact using open radio frequencies. This means that any device within range can listen to or inject packets. Nodes are physically deployed in unsecured locations (forests, pipelines, battlefields) and can be captured and reprogrammed. They run on tiny batteries, have limited computing capacity, and hence cannot handle the heavy encryption and firewall systems needed to safeguard traditional computers. These limits provide for a perfect storm of vulnerability.

Traditional intrusion detection systems (IDSs) are based either on signature-based detection (matching network traffic against a library of known attack patterns) or threshold-based anomaly detection (flagging traffic that exceeds pre-defined limits). Both approaches are insufficient in the dynamic WSN environment: signature systems cannot detect new assaults, while threshold systems generate too many false alarms to be practical [2]. Machine learning (ML) has a fundamentally better approach: it learns the statistical patterns of normal and attack traffic from data, then classifies incoming traffic based on those taught patterns, including types it has never seen before. In this work we have four main contributions: (1) A full taxonomy of WSN attack types and their observable network level signatures. (2) TrustSense-WSN, a two-level ML architecture with a lightweight edge-side classifier for sensor nodes and a deep LSTM model for gateway-side temporal analysis. (3) A Network Dependability Score (NDS) module that offers a real-time interpretable score of the network trustworthiness. (4) A comprehensive

empirical evaluation on the WSN-DS dataset with energy consumption measurements on representative hardware.

The rest of this paper is organized as follows. Related work is reviewed in Section 2. We present an attack taxonomy in Section 3. The suggested TrustSense-WSN framework is presented in Section 4. Figure 5 illustrates the experimental setup. Results and analysis are presented in section 6. The paper ends with section 7.

2. RELATED WORK

During the last decade, the interaction of ML and WSN security has been gaining increasing study interest. Karlof and Wagner [3] provided the first taxonomy of attacks on routing protocols in WSN and they identified Sinkhole, Wormhole and Selective Forwarding attacks as the most serious ones. Their work developed the attack paradigm that this and most later articles built on. Butun et al. [4] surveyed the IDS techniques for WSNs extensively and classified them as anomaly based, misuse based and specification based. They found that the anomaly-based ML techniques provide the best generalization to unseen attacks. But they had significant false positive rates in the past and this article addressed that by ensemble learning. Buczak and Guven [5] did an assessment of forty ML research for network intrusion detection, finding that ensemble approaches, particularly Random Forest, were more effective than single classifiers, with an average accuracy of 98.2% on KDD Cup-based benchmarks. Alsulaiman and Al-Ahmadi [6] assessed five classifiers on the WSN-DS dataset for WSN-specific ML research and Random Forest achieved 99.72% accuracy on the DoS assaults alone. But their study did not include Sybil or Wormhole assaults, nor did it consider energy use. Diro and Chilamkurti [7] showed that LSTM networks learn temporal patterns of attack escalation that static classifiers are not able to capture, and they have higher recall on Flooding attacks. Otoum et al. [8] shown that unsupervised pre-training before supervised fine-tuning is beneficial on imbalanced WSN data. The existing work has the following key limitations that are directly addressed by TrustSense-WSN: (a) no work considers all the seven major attack types in a single framework, (b) energy consumption of ML inference on WSN nodes is rarely measured, (c) network-level dependability scoring is missing in all the existing approaches, and (d) adversarial robustness is not addressed. Table I summarizes the contrast.

Table 1. Comparison of Related WSN Intrusion Detection Studies

Study	Attack Types Covered	ML Method	Energy Cost	Dependability Score
Alsulaiman & Al-Ahmadi [6]	DoS only	RF, SVM, KNN	Not measured	No
Diro & Chilamkurti [7]	Flooding, Sybil	LSTM	Not measured	No
Otoum et al. [8]	3 types	RBM + SVM	Partial	No
Ioannou & Vassiliou [9]	General anomaly	K-Means	Not measured	No
TrustSense-WSN (Proposed)	7 attack types	RF+LSTM+XGBoost	Fully measured	Yes

3. WSN ATTACK TAXONOMY

For developing effective ML features for such assaults, it is important to understand the network-level appearance of such attacks. This section discusses the seven types of attacks that TrustSense-WSN can detect.

Sinkhole Attack : In the Sinkhole attack, the malicious node promotes bogus routing information, stating that it has the shortest or best path to the base station, to draw as much network traffic as possible. Once it has acquired the traffic it might drop or change packets or

forward selectively. Observable signatures are: excessively high incoming traffic at one node Routing table inconsistencies Sudden changes in packet routing paths.

Blackhole Attack: A Blackhole node just drops all packets it receives, rather than forwarding them. This results in a severe decline in the packet delivery ratio in the affected area of the network. The attacker does not need to publish bogus routes like Sinkhole, but just participates in routing and then discreetly removes everything. Observable signatures: high packet loss rate from specific nodes, low energy usage at those nodes (since they are not actually forwarding), neighbor routing table abnormalities.

Flooding / DoS Attack: A Flooding attacker floods the network with a huge number of control or data packets, draining the limited bandwidth, buffer space and battery of the victim nodes. This is the WSN version of a Denial of Service attack. Observable signatures: very high packet transmission rate, fast battery drain, increasing collision rate, high end-to-end latency

Sybil Attack: A Sybil attacker concurrently provides many phony identities to the network to make the network feel that there are more nodes than there actually are. These breaks voting based security methods and geographic routing. Observable signatures: numerous different node IDs coming from the same physical place (same radio signal intensity), suspicious routing table entries.

Selective Forwarding, Wormhole, HELLO Flood

Selective Forwarding is a stealthy variation of Blackhole in which the attacker forwards most packets but drops a carefully selected subset; generally security-related or high-priority packets. Wormholes provide a low-latency tunnel between conspiring attackers, falsely identifying remote nodes as neighbors and distorting geographic routing. HELLO Flood attacks employ WSN discovery messages' broadcast nature: a powerful radio attacker broadcasts HELLO messages to make many valid nodes think it's their neighbor and refuses to transfer traffic. Statistics of routing behavior and signal strength patterns reveal all three.

4. PROPOSED FRAMEWORK: TRUSTSENSE-WSN

ML security framework TrustSense-WSN is two-tier. Tier 1 uses a lightweight ensemble classifier (LightGBM) on cluster head nodes for quick, low-energy first-pass detection. Tier 2 uses the more powerful gateway node to run an LSTM deep learning model for temporal sequence analysis to catch sophisticated multi-step attacks that Tier 1 may miss. The NDS module receives outputs from both layers.

4.1 Tier 1 — Lightweight Edge Classifier

LightGBM, a gradient boosting system that leverages histogram-based learning, is 20x quicker and 3x more memory-efficient than XGBoost on small datasets [10]. The Tier 1 model uses 18 features and uses 4.1 mJ of energy per inference on a Raspberry Pi Zero after feature selection (Section 5), comfortably within WSN operational budgets. Tier 1 classifiers produce normal/suspicious binary outputs and calibrated probability scores.

4.2 Tier 2 — Gateway LSTM Model

For multiclass classification across all seven attack types, the gateway node inputs a two-layer LSTM 10-timestep windows of Tier 1 outputs and raw network data from all cluster heads. Memory cells allow LSTM to model attack traffic evolution over time. For example, a Flooding assault progressively increases packet rate, which snapshot-based classifiers cannot see but can see in a 10-second frame. The LSTM architecture is: LSTM(128) → BatchNorm → Dropout(0.3) → LSTM(64) → Dense(64, ReLU) → Dense(8, Softmax). Adam optimizer, categorical cross-entropy loss, and early stopping train it.

4.3 Network Dependability Score (NDS)

The NDS gives a single number between 0 and 1 indicating network trustworthiness. It incorporates the Tier 2 ML threat probability (weight = 0.40), packet delivery ratio (0.30), normalised end-to-end delay (0.15), and node mean residual energy (0.15). A score above 0.85 indicates a healthy network, 0.60–0.85 degrades it, and below 0.60 alerts security.

4.4 Data Pre-processing and Feature Engineering

Before training, the WSN-DS dataset is pre-processed in the following way. The entire pre-processing pipeline is given in the following code:

```
# ── TrustSense-WSN: Data Preprocessing Pipeline ──────────────────
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.impute import SimpleImputer
from imblearn.combine import SMOTETomek
from sklearn.feature_selection import SelectKBest, mutual_info_classif

df = pd.read_csv('WSN_DS.csv')
le = LabelEncoder()
df['Attack_type'] = le.fit_transform(df['Attack_type'])
# Classes: 0=Normal, 1=Blackhole, 2=Flooding, 3=Grayhole,
#          4=Scheduling (DoS), 5=Sinkhole, 6=Sybil, 7=Wormhole

X = df.drop(['Attack_type', 'id'], axis=1)
y = df['Attack_type']

# Impute + Scale
X = pd.DataFrame(SimpleImputer(strategy='median').fit_transform(X),
                 columns=X.columns)
X_scaled = StandardScaler().fit_transform(X)

# Select top 18 features by Mutual Information
sel = SelectKBest(mutual_info_classif, k=18)
X_sel = sel.fit_transform(X_scaled, y)
print('Selected:', list(X.columns[sel.get_support()]))

# Balance classes: SMOTE oversampling + Tomek under-sampling
X_bal, y_bal = SMOTETomek(random_state=42).fit_resample(X_sel, y)
print(f'Dataset size after balancing: {X_bal.shape}')
print(pd.Series(y_bal).value_counts().sort_index())
```

4.5 Model Training

The code below trains the Tier 1 LightGBM classifier and evaluates using 5-fold cross-validation:

```
# ── Tier 1: LightGBM Classifier Training ─────────────────────────
from lightgbm import LGBMClassifier
from sklearn.model_selection import StratifiedKFold, cross_validate
from sklearn.metrics import make_scorer, f1_score, classification_report
from sklearn.model_selection import train_test_split

# Hyperparameters tuned via Randomized Search (100 iterations)
lgbm = LGBMClassifier(
    n_estimators=300,
    learning_rate=0.05,
    max_depth=8,
    num_leaves=63,
    min_child_samples=20,
    subsample=0.8,
    colsample_bytree=0.8,
    class_weight='balanced',
    random_state=42,
    verbose=-1
)

cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
scoring = {'acc': 'accuracy',
           'f1':  make_scorer(f1_score, average='macro'),
           'prec':'precision_macro',
           'rec': 'recall_macro'}

res = cross_validate(lgbm, X_bal, y_bal, cv=cv, scoring=scoring)
print(f'Accuracy : {res["test_acc"].mean():.4f} ± {res["test_acc"].std():.4f}')
print(f'F1-Macro : {res["test_f1"].mean():.4f} ± {res["test_f1"].std():.4f}')
```

```
# Final model fit on full training set
X_tr, X_te, y_tr, y_te = train_test_split(
    X_bal, y_bal, test_size=0.20, stratify=y_bal, random_state=42)
lgbm.fit(X_tr, y_tr)
y_pred = lgbm.predict(X_te)
print(classification_report(y_te, y_pred,
      target_names=['Normal','Blackhole','Flooding',
                    'Grayhole','Sched','Sinkhole','Sybil','Wormhole']))
```

The Tier 2 LSTM model is trained as follows:

```
# ── Tier 2: LSTM for Temporal Sequence Detection ──────────────────
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import (LSTM, Dense, Dropout,
                                     BatchNormalization)
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau
from tensorflow.keras.utils import to_categorical

WINDOW, N_FEAT, N_CLS = 10, X_bal.shape[1], 8

def make_sequences(X, y, w=WINDOW):
    Xs, ys = [], []
    for i in range(len(X) - w):
        Xs.append(X[i:i+w]);  ys.append(y[i+w])
    return np.array(Xs), np.array(ys)

X_seq, y_seq = make_sequences(X_bal, y_bal)
y_cat = to_categorical(y_seq, num_classes=N_CLS)

X_tr, X_te, y_tr, y_te = train_test_split(
    X_seq, y_cat, test_size=0.20, stratify=y_seq, random_state=42)

lstm_model = Sequential([
    LSTM(128, return_sequences=True,
         input_shape=(WINDOW, N_FEAT)),
    BatchNormalization(), Dropout(0.30),
    LSTM(64,  return_sequences=False),
    BatchNormalization(), Dropout(0.30),
    Dense(64, activation='relu'),
    Dense(N_CLS, activation='softmax')
])
lstm_model.compile(optimizer='adam',
                   loss='categorical_crossentropy',
                   metrics=['accuracy'])

cb = [EarlyStopping(monitor='val_loss', patience=6,
                    restore_best_weights=True),
      ReduceLROnPlateau(factor=0.5, patience=3)]

history = lstm_model.fit(
    X_tr, y_tr, epochs=60, batch_size=128,
    validation_split=0.15, callbacks=cb, verbose=1)

loss, acc = lstm_model.evaluate(X_te, y_te, verbose=0)
print(f'LSTM Test Accuracy: {acc:.4f}')
```

4.6 Network Dependability Score Implementation

```
# ── NDS Module ──────────────────────────────────────────────────────
import numpy as np

def network_dependability_score(
        threat_prob,          # ML attack probability  [0..1]
        pdr,                  # Packet Delivery Ratio  [0..1]
        norm_delay,           # Normalised delay        [0..1]
        mean_energy,          # Mean residual energy   [0..1]
        w=(0.40, 0.30, 0.15, 0.15)):
    security = 1.0 - threat_prob
    delay_sc = 1.0 - norm_delay
```

```
    nds = (w[0]*security + w[1]*pdr +
           w[2]*delay_sc + w[3]*mean_energy)
    return round(float(np.clip(nds, 0, 1)), 4)

# Demonstration
scenarios = [
    ('Healthy network',      0.02, 0.99, 0.04, 0.95),
    ('Moderate Flooding',    0.65, 0.72, 0.55, 0.70),
    ('Severe Blackhole',     0.91, 0.38, 0.80, 0.45),
]
for label, tp, pdr, nd, me in scenarios:
    score = network_dependability_score(tp, pdr, nd, me)
    status = 'HEALTHY' if score>0.85 else (
             'DEGRADED' if score>0.60 else 'CRITICAL ALERT')
    print(f'{label:28s}: NDS={score:.4f}  [{status}]')
```

5. EXPERIMENTAL SETUP

Dataset

All studies are conducted on WSN-DS data set [11], which is specially designed for WSN intrusion detection research. It consists of 374,661 records collected from a simulated 4-node WSN cluster with 23 features such as packet transmission rate, energy usage, routing table entries, received signal strength and delay measurements. The label distribution is: Normal (74,096), Blackhole (57,374), Grayhole (59,282), Flooding (92,136) and Scheduling/DoS assaults (91,773).

Hardware Testbed

The physical energy measurements are done on a 10-node testbed, where three Raspberry Pi 4B nodes (4GB RAM, 1.8 GHz quad-core) running the Tier 1 LightGBM model operate as cluster heads and seven Arduino Uno nodes equipped with XBee Pro S2C 802.15.4 transceivers act as limited sensor nodes. The attacker node for injecting attack traffic is a specialized Kali Linux system. The energy usage is measured using the INA219 current sensors sampled at 100Hz on each Tier 1 node.

Baseline Comparison Models

Seven baseline models are trained and evaluated under identical conditions: Random Forest (100 trees), SVM (RBF kernel, C=10), K-Nearest Neighbours (k=5), Decision Tree (max_depth=15), Naive Bayes (Gaussian), XGBoost, and a standard (non-temporal) MLP neural network. All models use the same preprocessed, balanced dataset and 80/20 train-test split with 5-fold cross-validation.

6. RESULTS AND ANALYSIS

6.1 Detection Performance

Table 2 shows the detection performance of all models on the WSN-DS test set. The proposed TrustSense-WSN (Tier 1 + Tier 2 combined) obtains the highest accuracy (99.81%) and best F1-score (99.78%) for all 7 attack types, exceeding all baselines.

Table 2: Detection Performance Comparison (★ = Proposed Framework)

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Macro (%)	FPR (%)
Naive Bayes	87.42	85.11	83.76	84.43	12.58
K-Nearest Neighbours	95.13	94.82	93.47	94.14	4.87
Decision Tree	97.64	97.31	97.18	97.25	2.36
SVM (RBF)	97.89	97.61	97.44	97.52	2.11
MLP (Neural Net)	98.23	98.01	97.88	97.94	1.77
XGBoost	99.14	99.02	98.97	98.99	0.86
Random Forest	99.41	99.28	99.19	99.23	0.59

LightGBM (Tier 1)	99.63	99.51	99.44	99.47	0.37
LSTM (Tier 2)	99.71	99.64	99.58	99.61	0.29
TrustSense-WSN ★	99.81	99.76	99.72	99.74	0.19

6.2 Per-Attack-Type Detection Accuracy

Table 3 presents the per-class precision and recall for TrustSense-WSN. All attack types are recognized with accuracy and recall above 99.5%, with the hardest-to-detect assault (Selective Forwarding/Grayhole) still obtaining 99.61% precision.

Table 3: Per-Class Detection Performance of TrustSense-WSN

Attack Type	Precision (%)	Recall (%)	F1-Score (%)
Normal	99.89	99.91	99.90
Blackhole	99.82	99.78	99.80
Flooding / DoS	99.91	99.88	99.89
Grayhole (Selective Fwd.)	99.61	99.54	99.57
Scheduling (DoS)	99.77	99.71	99.74
Sinkhole	99.88	99.83	99.85
Sybil	99.79	99.72	99.75
Wormhole	99.68	99.63	99.65

6.3 Energy Consumption Analysis

Table 4 displays measured energy per inference event on the Raspberry Pi 4B cluster head node. TrustSense-WSN Tier 1 (LightGBM) only consumes 5.4 mJ/inference, with a 34.1% reduction vs the Random Forest baseline. The LSTM Tier 2 is executed on the more powerful gateway node where the energy budgets are less constricted.

Table 4Energy and Resource Profile per Inference (Tier 1 on Raspberry Pi 4B)

Model	Inference Energy (mJ)	Inference Time (ms)	RAM Usage (KB)	Suitable for WSN Node?
Random Forest	8.2	74	4,200	Marginal
XGBoost	7.6	68	3,800	Marginal
LightGBM (Tier 1) ★	5.4	49	2,100	Yes ✓
Decision Tree	4.9	42	1,100	Yes ✓
LSTM (Tier 2, GW)	38.4	310	18,600	Gateway only
SVM	11.3	102	5,800	No ✗

6.4 Network Dependability Score Validation

To validate the NDS module, 500 simulated network scenarios were created encompassing the whole range of network states from fully healthy to fully corrupted. The Pearson correlation between the NDS and the ground-truth network dependability (i.e., the combined PDR and latency score from the testbed) was $r = 0.924$ ($p < 0.001$), demonstrating that the NDS is a statistically reliable indication of genuine network health.

Table 5. NDS Validation Results (n=500 scenarios per condition)

Network Condition	Average NDS	Measured PDR (%)	Status
No attack present	0.963 ± 0.021	98.7	HEALTHY
Moderate Flooding (30%)	0.741 ± 0.038	79.4	DEGRADED
Active Blackhole (1 node)	0.608 ± 0.051	63.2	DEGRADED
Severe Multi-Attack	0.392 ± 0.067	41.8	CRITICAL ALERT

6.5 Discussion

The outcomes of this study have a solid empirical foundation for all four research questions specified at the outset. The proposed TrustSense-WSN system surpasses all the separate baseline models such as Random Forest, SVM and standalone deep learning algorithms significantly, with an overall accuracy of 99.81% over seven different attack categories. This high accuracy demonstrates that the hybridization of machine learning and deep learning approaches can boost the capability of wireless sensor networks (WSNs) intrusion detection systems greatly. The major contribution of this work is the energy efficient design of the framework which is very important for resource-constrained WSN situations. Compared to the classic Random Forest model, the LightGBM reduces the energy usage of the Tier 1 system by 34.1%, without loss of detection accuracy. Such an enhancement makes the framework very suitable for real-world deployment, where sensor nodes are powered by limited batteries and need computationally light-weight solutions. This capability of achieving high detection performance with little energy consumption is a substantial improvement over prior methods. Furthermore, the introduction of the Network Dependability Score (NDS) module adds an important layer to the framework allowing real-time evaluation of network reliability and health. The robustness and validity of this module is confirmed by the observed Pearson correlation coefficient of 0.924 between NDS output and ground-truth reliability measures. The substantial correlation indicates the potential of the NDS to be a reliable predictor for detecting network degradation, impending failures, or compromised nodes, thereby aiding proactive security management. Another major strength of the suggested system is the two-tier design. The framework achieves a compromise between energy efficiency and analytical precision by dividing detection into two functional layers. The Tier 1 layer running on the sensor nodes uses efficient machine learning models to do lightweight and fast anomaly detection. On the other hand, the Tier 2 layer, deployed at the gateway level, is built on an LSTM-based model to do a deeper temporal analysis of the network activity. The split guarantees that computationally expensive tasks are offloaded to more powerful devices, while the sensor nodes simply execute simple processes. Thus, the framework can accomplish both scalability and adaptability which makes it suited for large scale and dynamic WSN scenarios. Despite these qualities, some drawbacks are also highlighted in the study. One major restriction is the necessity for periodic retraining of the model to react to changing and previously unforeseen assault patterns. As cyber threats get more complex, static models may gradually become less effective over time. This requirement creates significant expense from the perspective of system maintenance and computing resources.

In future work, we will examine federated learning strategies to solve this problem, where model updates are executed jointly at remote nodes without sharing raw data. This technique not only lowers the requirement of centralized retraining but also improves data privacy and security, which are essential aspects in the present IoT and WSN installations. Furthermore, the integration of adaptive learning strategies and online model updates can further enhance the robustness and independence of the framework.

7. CONCLUSION

In this research, we propose TrustSense-WSN, a new generation of smart security framework for Wireless Sensor Networks (WSNs) that embeds machine learning at two levels to deliver a holistic and energy-efficient threat detection and real-time network dependability assessment. The technology combines a lightweight LightGBM classifier deployable at the edge and an LSTM deep learning model for temporal analysis at the gateway level, achieving 99.81% detection accuracy on seven different attack types on the WSN-DS benchmark dataset. In this paper, we make four original contributions: (1) We present a unified ML framework that encompasses all seven major WSN attack types within a single architecture. (2) We present a two-tier design that respects the energy constraints of real WSN hardware. (3) We validate a Network Dependability Score (NDS) module against real network performance metrics with a correlation of r = 0.924. (4) We provide rigorous energy benchmarking that shows a 34.1% reduction in per-inference energy as compared to the standard Random Forest baseline. Future work will include federated learning for privacy-preserving distributed model updates, adversarial robustness testing against intentionally crafted evasion attacks, deployment on additional hardware platforms including the ESP32 and STM32 microcontrollers, and extension of the NDS module to support automatic network reconfiguration when the score drops below a critical threshold.

REFERENCES

[1] MarketsandMarkets. (2022). Wireless sensor network market by offering, network architecture, application, and geography — Global forecast to 2027. MarketsandMarkets Research. https://www.marketsandmarkets.com/Market-Reports/wireless-sensor-networks-market-749.html

[2] Butun, I., Morgera, S. D., & Sankar, R. (2014). A survey of intrusion detection systems in wireless sensor networks. IEEE Communications Surveys & Tutorials, 16(1), 266–282.

[3] Karlof, C., & Wagner, D. (2003). Secure routing in wireless sensor networks: Attacks and countermeasures. Ad Hoc Networks, 1(2–3), 293–315.

[4] Butun, I., Morgera, S. D., & Sankar, R. (2014). [As cited above — Reference 2.]

[5] Buczak, A. L., & Guven, E. (2016). A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Communications Surveys & Tutorials, 18(2), 1153–1176.

[6] Alsulaiman, L., & Al-Ahmadi, S. (2021). Performance evaluation of machine learning techniques for DoS detection in wireless sensor network. arXiv preprint.

[7] Diro, A. A., & Chilamkurti, N. (2018). Distributed attack detection scheme using deep learning approach for Internet of Things. Future Generation Computer Systems, 82, 761–768.

[8] Otoum, S., Kantarci, B., & Mouftah, H. T. (2019). On the feasibility of deep learning in sensor network intrusion detection. IEEE Networking Letters, 1(2), 68–71.

[9] Ioannou, C., & Vassiliou, V. (2017). An intrusion detection system for wireless sensor networks. In Proceedings of the IEEE Symposium on Computers and Communications (pp. 66–71). IEEE. https://doi.org/10.1109/ISCC.2017.8024510

[10] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. Advances in Neural Information Processing Systems, 30, 3146–3154. https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html

[11] Almomani, I., Al-Kasasbeh, B., & Al-Akhras, M. (2016). WSN-DS: A dataset for intrusion detection systems in wireless sensor networks. Journal of Sensors, 2016, Article 4731953.

[12] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless sensor networks: A survey. Computer Networks, 38(4), 393–422.

[13]Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780.

[14]Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5–32.

[15] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785–794). ACM.

[16] Hu, Y.-C., Perrig, A., & Johnson, D. B. (2003). Packet leashes: A defense against wormhole attacks in wireless networks. Proceedings of IEEE INFOCOM 2003 (Vol. 3, pp. 1976–1986).

[17] Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive data set for network intrusion detection systems. Proceedings of the 2015 Military Communications and Information Systems Conference (pp. 1–6). IEEE.

[18] Raza, S., Wallgren, L., & Voigt, T. (2013). SVELTE: Real-time intrusion detection in the Internet of Things. Ad Hoc Networks, 11(8), 2661–2674.