Graphical Abstract

Bayesian Deep Learning framework for Uncertainty-Aware Intrusion Detection to enhance the Security in High Dense WSN

Venkateswari .P , Shanmugasundaram .N

Highlights

Bayesian Deep Learning framework for Uncertainty-Aware Intrusion Detection to enhance the Security in High Dense WSN

Venkateswari .P , Shanmugasundaram .N

- A Bayesian Deep Learning–based intrusion detection framework is proposed to quantify and utilize predictive uncertainty in high-density WSN environments.
- The model integrates Monte Carlo Dropout and Bayesian Neural Networks to distinguish confident predictions from uncertain ones, reducing false alarms.
- The proposed approach enables scalable, energy-efficient, and trustworthy IDS suitable for next-generation IoT and industrial deployments.

Bayesian Deep Learning framework for Uncertainty-Aware Intrusion Detection to enhance the Security in High Dense WSN

Venkateswari .P , Shanmugasundaram .N

AVS College of Technology, Computer Science and Engineering, salem, 636106, Tamil nadu, india

Sri Eshwar College of Engineering, Electronics and Communication Engineering, coimbatore, 641202, Tamil nadu, india

Abstract

Wireless Sensor Networks (WSNs) are vulnerable to evolving cyber threats, necessitating robust intrusion detection systems (IDS) that quantify uncertainty to enhance decision-making. This research presents a Bayesian Deep Learning (BDL) framework for uncertainty-aware intrusion detection in WSNs, leveraging probabilistic neural networks to model uncertainty in predictions. Traditional deep learning IDS lack uncertainty estimation, leading to overconfident but erroneous classifications. The proposed approach combines the Monte Carlo Dropout algorithm and Bayesian Neural Networks (BNNs) to provide probabilistic outputs, enabling the system to distinguish between certain and uncertain predictions. The proposed framework integrates temporal feature extraction with Bayesian inference for robust anomaly detection. The model achieved 99.5% accuracy, 97.5% precision, 96.5% recall, 98.5% f1-score, 95.5% detection rate and 4.1% false alarm rate. The results demonstrate that uncertainty-awareness is critical for trustworthy IDS in dynamic, resource-constrained WSN environments.

Keywords:
Sensors, Networks, Bayesian Deep Learning, Uncertainty, Intrusion Detection, Wireless, Anomaly Detection

1. Introduction

Uncertainty-aware intrusion detection refers to systems that account for uncertainties in data, network dynamics, and attack patterns to improve detection accuracy. WSNs are prone to uncertainties due to unreliable communication, sensor noise, and evolving threats [1]. Traditional intrusion detection systems may generate false alarms or miss attacks if they ignore these uncertainties. Uncertainty-aware IDS is essential because WSNs operate in unpredictable conditions where data loss, noise, and dynamic topology changes are common [2]. Conventional IDS often fail to adapt, leading to high false-positive or false-negative rates [3]. The attackers exploit these uncertainties to evade detection. For instance, low-power jamming or spoofing attacks may mimic natural network fluctuations [4]. An uncertainty-aware approach improves robustness by quantifying doubt in observations, enabling adaptive responses to both known and emerging threats, thus ensuring better security in critical applications like military surveillance or environmental monitoring [5]. Several challenges arise in uncertainty-aware intrusion detection. Data unreliability due to packet loss or sensor errors complicates threat analysis [6]. Resource constraints (limited energy, computation, and memory) restrict complex detection algorithms [7]. Dynamic attack patterns (e.g., zero-day exploits) demand adaptive models. The high false alarms from environmental noise reduce system trust. The scalability issues arise in large-scale deployments where centralized detection becomes impractical [8]. These factors collectively degrade detection efficiency and increase vulnerability. Hybrid approaches combining lightweight machine learning (e.g., Bayesian networks, federated learning) with probabilistic reasoning can enhance accuracy while conserving resources to mitigate these challenges [9]. Redundant data validation and trust-based neighbor collaboration reduce false alarms. Adaptive algorithms can dynamically update detection rules against evolving attacks [10]. Decentralized or hierarchical IDS architectures improve scalability, distributing computational loads. The energy-efficient cryptographic techniques and anomaly detection thresholds tuned to network conditions can further optimize performance, ensuring reliable intrusion detection despite uncertainties [11]. Traditional machine learning (ML) and deep learning (DL) models struggle to effectively capture uncertainty in intrusion detection due to several inherent limitations [12]. Most ML/DL models are designed to provide deterministic predictions rather than probabilistic confidence estimates, making it difficult to distinguish between true attacks and uncertain or noisy observations [13]. Supervised learning approaches rely heavily on labeled datasets, which often lack comprehensive coverage of all possible attack variations and environmental uncertainties, leading to overconfidence in incorrect predictions. Many models do not incorporate explicit uncertainty quantification mechanisms, which are crucial for assessing confidence levels in dynamic and noisy WSN environments [14]. Deep learning models, while powerful, are particularly prone to overfitting in small or imbalanced datasets, further amplifying uncertainty misestimation. The black-box nature of complex DL architectures makes it challenging to interpret and calibrate uncertainty, resulting in unreliable intrusion alerts under real-world conditions where data ambiguity is common [15]. These limitations highlight the need for more robust uncertainty-aware techniques to improve intrusion detection reliability. The key contributions of proposed research work have the following, • The proposed research combines deep neural networks with Bayesian probability to quantify uncertainty in intrusion detection, improving reliability in noisy WSN environments. It provides probabilistic confidence scores for predictions, distinguishing between certain and uncertain detections to reduce false positives/negatives. • The proposed model optimizes computational efficiency using variational inference or Monte Carlo dropout, ensuring feasibility in resource-constrained, large-scale deployments. It handles incomplete, noisy, and adversarial data by leveraging Bayesian inference, making the system resilient to sensor errors and dynamic attack patterns. • The proposed model provides interpretable uncertainty metrics (e.g., entropy, prediction intervals) to aid security analysts in decision-making. It reduces computational overhead through lightweight Bayesian approximations, extending the WSN node lifetime while maintaining security. Remaining parts of the manuscript has organized as the following. Section-2 illustrates the detailed explanation about the materials and methods. Section-3 demonstrates the results and Section-4 explained the complete details about the performance analysis. Section-5 provides the conclusion and future scope of proposed research work.

2. Materials and Methods

High-density WSNs face unique security challenges due to their large-scale deployments, resource constraints, and dynamic environments. Traditional IDS often struggle with false positives and false negatives and adaptability in such settings. Uncertainty-aware intrusion Detection addresses these issues by incorporating probabilistic reasoning, confidence estimation, and adaptive decision-making, making it crucial for securing dense WSNs. Table 1 shows the comprehensive analysis of existing models.

2.1. Research Gap Analysis

- Existing uncertainty-aware IDS struggle with real-time processing in ultra-dense networks. Lightweight, distributed detection mechanisms need to be incorporated to handle massive data flows efficiently.
- Most IDS rely on static uncertainty models, failing to adapt to dynamic WSN conditions. Incorporate adaptive Bayesian or fuzzy logic-based uncertainty modeling for node mobility and link fluctuations.
- Uncertainty modeling increases computational overhead, reducing node lifetime. Energy-aware sensing techniques should be incorporated to balance accuracy and power consumption.
- • Current IDS use either probabilistic or non-probabilistic methods. For improved detection, hybrid uncertainty fusion combining multiple approaches needs to be incorporated.

2.2. Research Novelty

- The proposed framework provides predictive uncertainty to assess intrusion detection confidence. It enables WSNs to distinguish between uncertain predictions (due to noise) and true attacks, reducing false positives.
- Bayesian Neural Networks treat weights as probability distributions rather than fixed values, making them resilient to adversarial evasion attacks. The proposed model has Monte Carlo Dropout (MC-Dropout) at inference time. It allows sampling-based uncertainty estimation without retraining.
- The proposed framework can learn from limited labeled data using Bayesian active learning. Here, Uncertainty-guided sampling selects the most informative data points for labeling, improving efficiency.
- The proposed dynamically adjusts detection thresholds based on uncertainty levels and reduces false negatives in high-noise WSN environments.

Table 1: Comprehensive Analysis

Author	**Year**	**Model**	**Advantage**	**Limitation**
Karthikeyan, M., et al. [16]	2024	Firefly Algorithm + ML	Optimized feature selection	High computational overhead
Liu, G., et al. [17]	2022	Improved kNN	Low complexity, real-time detection	Sensitive to imbalanced data
Pan, J. S., et al. [18]	2021	Lightweight ML-based IDS	Low resource usage	Less robust against advanced attacks
Ramana, K., et al. [19]	2022	WOCRU-IDS (Whale + GRU)	High accuracy	Complex, not ideal for low-power WSNs
Sadia, H., et al. [20]	2024	ML-based IDS	Detects multiple attack types	Needs large labeled datasets
Alsahli, M. S., et al. [21]	2021	SVM and RF	Helps choose best network level	Less precision and sensitive to imbalanced data
Jingjing, Z., et al. [22]	2022	MC-GRU	Captures temporal dependencies well	High memory usage
Arkan, A., et al. [23]	2023	Unsupervised hierarchical IDS	No need for labeled data	Higher false positives
Talukder, M. A., et al. [24]	2024	ML + SMOTE-Tomek	Handles imbalanced data well	Increased computational cost
Saleh, H. M., et al. [25]	2024	SGD-based IDS	Fast convergence	May get stuck in local optima
Ahmed, O., et al. [26]	2024	Context-aware ML	Better detection with context	Increased complexity
Subramani, S., et al. [27]	2023	Multi-Objective PSO	Effective feature selection	Premature convergence in PSO
Safaldin, M., et al. [28]	2021	Binary GWO + SVM	High accuracy	SVM scalability issues
Jin, J., et al. [29]	2021	Unspecified ML model	Realistic simulation	Lack of model details
Madhuri, K., et al. [30]	2022	Node-level IDS	Specialized for drop attacks	Limited to specific attacks
Otair, M., et al. [31]	2022	Hybrid GWO-PSO	Better optimization	High computational complexity
Hussain, K., et al. [32]	2022	WOA-ABC + CNN	High accuracy with deep learning	Needs large data/resources

- The proposed framework can be optimized for resource-constrained WSNs using Bayesian approximation techniques. It enables low-latency uncertainty-aware detection at the edge.

2.3. System Design

The proposed Uncertainty-Aware Intrusion Detection System (UA-IDS) for high-density WSNs follows a modular, adaptive architecture designed to enhance security while accounting for uncertainty. Fig.1 shows the construction of the proposed design

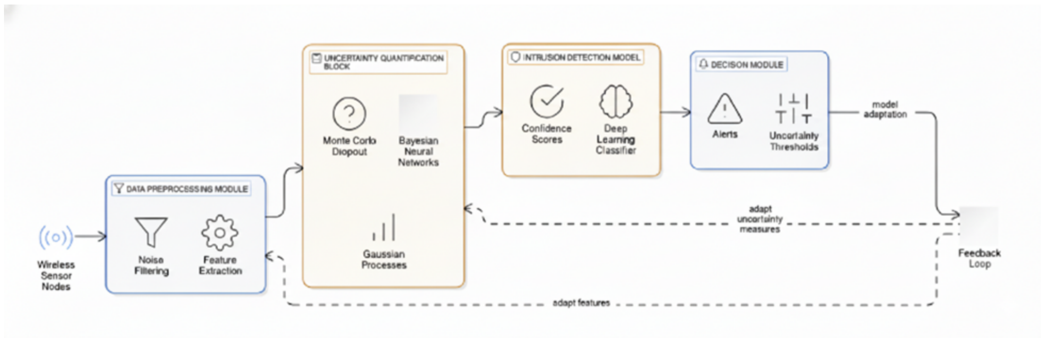


Figure 1: This is a figure. Schemes follow the same formatting

WSN sensor nodes collect raw data, which is preprocessed by a data processing module that performs noise filtering (e.g., Kalman filtering) and feature extraction (e.g., statistical, temporal features). The uncertainty quantification block leverages Monte Carlo Dropout (MC-Dropout), Bayesian Neural Networks (BNNs), and Gaussian Processes (GPs) to estimate both aleatoric (data noise) and epistemic (model uncertainty). These uncertainty measures feed into an intrusion detection model, which combines deep learning classifiers with confidence scores to distinguish true attacks from false alarms. The decision module applies dynamic uncertainty thresholds to trigger alerts—high-confidence detections generate immediate warnings, while uncertain predictions undergo further analysis. The system incorporates a feedback loop where anomalies and uncertainty trends are detected, and the model is refined to ensure adaptability. The model adaptation mechanism adjusts uncertainty measures (e.g., recalibrating dropout rates) and feature selection (e.g., discarding noisy features) based on real-time performance. This closed-loop design enables continuous improvement, making the UA-IDS robust against evolving threats and dynamic WSN conditions.

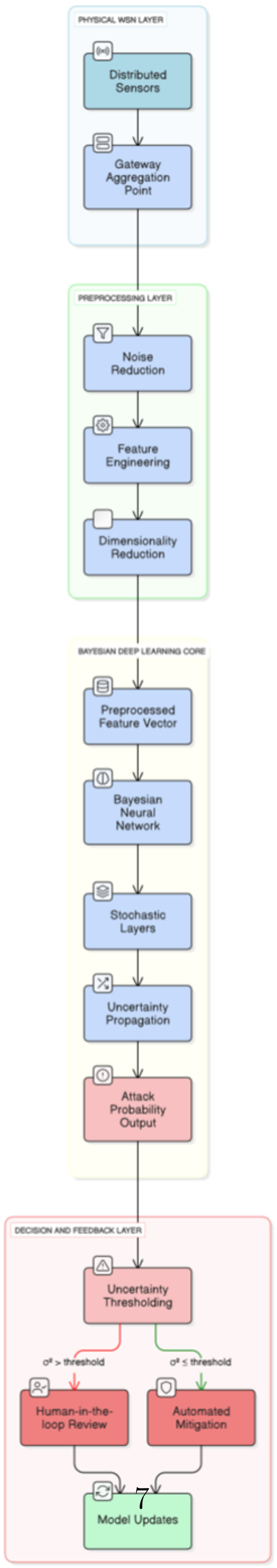


Figure 2: Bayesian Neural Network Architecture

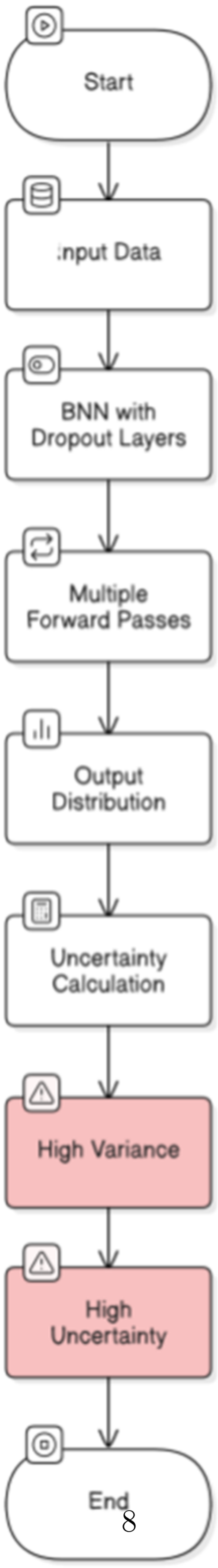


Figure 3: Monte Carlo Dropout Sampling Process

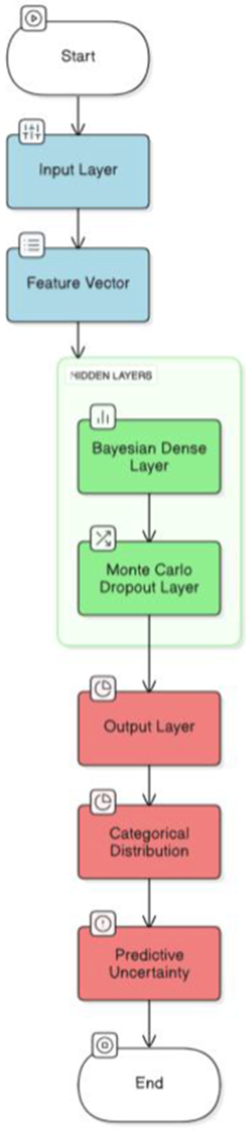


Figure 4: Uncertainty-Aware Decision Workflow

2.4. Bayesian Neural Network Architecture

The proposed framework operates across four key layers to enhance security in high-density Wireless Sensor Networks (WSNs):

- **Physical WSN Layer :** This layer comprises distributed sensor nodes and collects raw network traffic and environmental data. Adversarial exposure are highlighted, and data is aggregated at a central gateway node for further processing.
- **Preprocessing Layer :** Raw sensor data undergoes noise reduction to mitigate environmental interference. Feature engineering extracts discriminative attributes, followed by dimensionality reduction to optimize computational efficiency for resource-constrained nodes.
- **Bayesian Deep Learning Core :** The pre-processed feature vector $\mathbf{X} \in \mathbb{R}^d$ is fed into a Bayesian Neural Network (BNN) with stochastic layers, where the weights follow probability distributions rather than fixed values. The uncertainty propagates through multiple stochastic forward passes, generating both attack probabilities $p(y \mid \mathbf{X})$ and predictive variance σ^2. This quantifies the model confidence, distinguishing between true attacks and noise.
- **Decision & Feedback Layer :** A dynamic thresholding mechanism triggers actions based on uncertainty:
  - If $\sigma^2 >$ threshold $\rightarrow$ Human-in-the-loop review for ambiguous cases.
  - If $\sigma^2 \leq$ threshold $\rightarrow$ Automated mitigation.
  - Continuous model updates via Bayesian inference adapt to emerging threats, ensuring robust performance in evolving WSN environments.

Bayesian inference is based on Bayes' theorem, which states the following:

$$P(H \mid D) = \frac{P(D \mid H) \cdot P(H)}{P(D)} \tag{1}$$

Where, $P(H \mid D)$ is the posterior probability of hypothesis H given data D, $P(D \mid H)$ is the likelihood of data D given hypothesis H, $P(H)$ is the prior probability of hypothesis H, and $P(D)$ is the marginal likelihood of data D.

A neural network can be represented as a function $f(x;\theta)$, where x is the input data and θ represents the weights and biases of the network. In a Bayesian Neural Network, the weights θ are treated as random variables with a prior distribution $P(\theta)$.

The posterior distribution of the weights given the data can be expressed as:

$$P(\theta \mid D) = \frac{P(D \mid \theta) \cdot P(\theta)}{P(D)} \tag{2}$$

Where, $P(D \mid \theta)$ is the likelihood of the data given the weights and $P(\theta)$ is the prior distribution of the weights. For intrusion detection, the likelihood can be modeled using a softmax function for multi-class classification or a sigmoid function for binary classification.

$$P(D \mid \theta) = \prod_{i=1}^{N} f(y_i \mid f(x_i;\theta)) \tag{3}$$

Where, y_i is the true label for input x_i. Common choices for the prior distribution $P(\theta)$ include the following cases,

- **Case 1: Gaussian prior:** $\theta \sim \mathcal{N}(0, \sigma^2)$
- **Case 2: Laplace prior:** $\theta \sim \text{Laplace}(0, b)$

Since the exact posterior $P(\theta \mid D)$ is often intractable, we use variational inference to approximate it. We define a variational distribution $Q(\theta)$ and minimize the Kullback–Leibler (KL) divergence:

$$KL(Q(\theta) \,\|\, P(\theta \mid D)) = \int Q(\theta) \, \log \frac{Q(\theta)}{P(\theta \mid D)} \, d\theta \tag{4}$$

The loss function for training the BNN can be derived from the evidence lower bound (ELBO) as follows:

$$L = \mathbb{E}_{Q(\theta)}[\log P(D \mid \theta)] - KL\big(Q(\theta) \parallel P(\theta)\big) \tag{5}$$

The uncertainty in predictions can be quantified using the posterior distribution of the weights. For a new input x^*, the predictive distribution is given by:

$$P(y^* \mid x^*, D) = \int P(y^* \mid x^*, \theta)\, P(\theta \mid D)\, d\theta \tag{6}$$

This integral can be approximated using Monte Carlo methods or variational inference. The proposed Bayesian Neural Network enhances security by quantifying uncertainty in intrusion detection, which is crucial for distinguishing real attacks from noisy sensor data. It treats model weights as probability distributions rather than fixed values, enabling them to capture epistemic uncertainty (model uncertainty due to limited training data) and aleatoric uncertainty (inherent data noise). By integrating BNNs, WSNs gain a probabilistic defense mechanism that balances detection accuracy with operational efficiency, which is critical for high-density deployments. In a traditional neural network, dropout is used only during training to prevent overfitting, but in this Bayesian approach, dropout remains active during both training and inference. It allows the network to generate multiple stochastic predictions for the same input, effectively sampling from the posterior distribution of the model parameters. The uncertainty in predictions is captured by the variability in these stochastic forward passes. Algorithm.1 shows the working of BNN algorithm to detect uncertainty.

Algorithm 1 BNN Algorithm

```
1: function CREATE_BNN($input\_dim$, $output\_dim$)
2:     model ← Sequential()
3:     model.add(Dense(units=128, activation='relu', input_dim=$input\_dim$))
4:     model.add(Dropout(rate=0.5))                  ▷ Dropout for uncertainty
5:     model.add(Dense(units=64, activation='relu'))
6:     model.add(Dropout(rate=0.5))                  ▷ Dropout for uncertainty
7:     model.add(Dense(units=$output\_dim$, activation='softmax'))   ▷ Output layer
8:     return model
9: end function
```

The model is structured as a sequential neural network with dense layers and ReLU activations. The key Bayesian component is the inclusion of Dropout layers with a rate of 0.5, which introduces randomness during inference. By performing multiple forward passes with dropout enabled, the model produces a distribution of outputs rather than a single deterministic prediction. The final layer uses a softmax activation for classification tasks, where the output probabilities can be interpreted alongside their uncertainty. This approach provides a computationally efficient way to approximate Bayesian inference without explicitly defining prior distributions over weights, as required in full Bayesian neural networks.

2.5. Monte Carlo Dropout Sampling Process

The Monte Carlo (MC) Dropout sampling process enables Bayesian uncertainty estimation in deep learning models without modifying the base neural network architecture. The workflow begins by feeding preprocessed input data into a Bayesian Neural Network with active dropout layers during inference. It is a regularization technique used in neural networks to prevent overfitting. During training, dropout randomly sets a fraction of the neurons to zero. This can be mathematically represented as the following,

$$\hat{y} = f(x; \theta * r) \tag{7}$$

Where, y is the output of the network with dropout, $f(x;\theta)$ is the neural network function with weights θ, and $\mathbf{r}$ is a binary mask vector where each element is drawn from a Bernoulli distribution $r_i \sim \text{Bernoulli}(p)$ (where p is the probability of keeping a neuron active). In a Bayesian Neural Network, we treat the weights as random variables. The dropout process can be interpreted as a way to approximate the posterior distribution of the weights. Monte Carlo sampling using dropout to estimate uncertainty in predictions. For a given input x^*, perform multiple forward passes through the network with dropout enabled. Each forward pass will yield a different output due to the random dropout mask. Let y_i^* be the output from the i-th forward pass. The outputs can be collected to form a distribution:

$$OD = \{y_1^*, y_2^*, \ldots, y_N^*\} \tag{8}$$

Where N is the number of Monte Carlo samples. The mean and variance of the outputs can be computed as following.

Mean Prediction

$$\widehat{y^*} = \frac{1}{N}\sum_{i=1}^{N} y_i \tag{9}$$

Variance (Uncertainty) Prediction

$$\sigma^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i^* - \widehat{y^*})^2 \tag{10}$$

The final prediction for a given input x^* can be reported as:

- **Predicted class:** The class corresponding to the highest mean output.
- **Uncertainty:** The variance σ^2 can be used to assess the confidence in the prediction. A higher variance indicates greater uncertainty.

The MC Dropout retains stochasticity, executing multiple forward passes with random dropout masks applied to neurons. Each pass yields a slightly different prediction due to the model's inherent randomness, generating a distribution of output probabilities. Uncertainty is quantified by computing the variance (σ^2) across these predictions. Key variance metrics include:

- **Epistemic Uncertainty (model uncertainty):** High variance indicates insufficient training data or ambiguous inputs.

- **Aleatoric Uncertainty (data noise):** Captures inherent sensor measurement variability.

If the variance exceeds a predefined threshold, the sample is flagged as high uncertainty, triggering human review or additional verification. Conversely, low-variance predictions undergo automated mitigation. The process terminates with updated model confidence estimates, enhancing intrusion detection robustness in dynamic WSN environments.The Monte Carlo Dropout (MC Dropout) Sampling Algorithm is a practical approximation of Bayesian inference in deep neural networks, leveraging dropout layers to estimate prediction uncertainty. In this method, dropout remains active during inference (by setting `training=True`), enabling the network to produce stochastic predictions for the same input across multiple forward passes. Algorithm 2 provides the functional structure of the Monte Carlo Dropout Sampling Algorithm.

Algorithm 2 Monte Carlo Dropout Sampling Algorithm

```
1: function MonteCarloDropout($model, input\_data, num\_samples$)
2:     predictions ← [ ]
3:     for $i = 1$ to $num\_samples$ do
4:         $pred$ ← $model.predict(input\_data,\ training = \text{True})$    ▷ Enable dropout during prediction
5:         append $pred$ to predictions
6:     end for
7:     mean_prediction ← mean(predictions)
8:     uncertainty ← variance(predictions)    ▷ Estimate uncertainty
9:     return mean_prediction, uncertainty
10: end function
```

The algorithm works by sampling predictions ‘num_samples’ times, each time with different neurons randomly dropped out, simulating draws from the model’s posterior distribution. The mean prediction is computed by averaging these samples, representing the final prediction, while the variance across samples quantifies the model’s uncertainty. MC Dropout is particularly useful in safety-critical applications where understanding prediction confidence is essential.

2.6. Uncertainty-Aware Decision Workflow

The system starts by feeding preprocessed sensor data into the input layer as a feature vector $\mathbf{X} \in \mathbb{R}^d$. The hidden layers consist of Bayesian dense layers where weights follow probability distributions (e.g., Gaussian priors $\mathcal{N}(\mu, \sigma^2)$), enabling weight uncertainty, and Monte Carlo Dropout layers that remain active during inference, applying random masks to neurons. The output layer produces a categorical distribution over attack classes via a softmax function, where each forward pass generates slightly different predictions due to the stochastic layers. Predictive uncertainty is quantified by computing variance across multiple passes (epistemic uncertainty) and inherent noise in predictions (aleatoric uncertainty). The process ends by outputting both the attack probability distribution and an associated uncertainty measure, allowing the system to make confidence-aware security decisions.

Let $\hat{y}$ be the predicted output and σ^2 be the estimated uncertainty (variance) from the model. The uncertainty can be quantified as follows:

$$U(x^*) = \sigma^2 \tag{11}$$

where $U(x^*)$ represents the uncertainty associated with the prediction for input x^*. Now, the threshold level can be used for decision making:

- **Threshold 1: Low Uncertainty**: If $U(x^*) < \tau_1$ (where τ_1 is a predefined low uncertainty threshold), classify the input as normal or as a specific type of intrusion confidently.
- **Threshold 2: Moderate Uncertainty**: If $\tau_1 \leq U(x^*) < \tau_2$ (where τ_2 is a predefined moderate uncertainty threshold), flag the input for further analysis or human review.
- **Threshold 3: High Uncertainty**: If $U(x^*) \geq \tau_2$, do not make a classification and alert the system for potential anomalies, indicating that the model is uncertain about the input.

Use feedback from the decisions made (correct or incorrect) to retrain the model periodically, enhancing its ability to handle uncertainty in future predictions. Algorithm.3 provides the detailed structure of Uncertainty-Aware Decision Algorithm.

Algorithm 3 Uncertainty-Aware Decision Algorithm

```
function DECISION_WORKFLOW($model, input\_data$)
    mean_pred, uncertainty ← MONTECARLODROPOUT($model, input\_data, 100$)
    if $uncertainty < \tau_1$ then
        decision ← "Classify"    ▷ Low uncertainty: classify confidently
    else if $\tau_1 \leq uncertainty < \tau_2$ then
        decision ← "Review needed"    ▷ Moderate uncertainty: flag for analysis
    else
        decision ← "Alert: Uncertain input"    ▷ High uncertainty: potential anomaly
    end if
    return decision
end function
```

This algorithm uses Monte Carlo Dropout to make reliable predictions while accounting for model uncertainty. First, it generates multiple predictions (num_samples=100) and computes the mean prediction (most likely output) and uncertainty (variance across samples). Based on predefined thresholds (tau1 and tau2), the algorithm takes different actions: if uncertainty is low (below tau1), it confidently classifies the input; if uncertainty is moderate, it flags the result for human review; and if uncertainty is high (above tau2), it raises an alert for potential anomalies. This approach ensures robust decision-making in high-stakes scenarios where unreliable predictions must be detected and handled appropriately.

3. Results

The performance of proposed Bayesian Deep Learning framework (BDLF) has compared with the existing uncertainty-aware cluster-based deployment approach (UCBA) [13], Firefly algorithm (FFA) [16], enhanced intrusion detection model (EIDM) [17]. light-weight intelligent intrusion detection model (LIDM) [18] and Intrusion detection algorithm (IDA) [29]. Here, the Network simulator (NS-2) and WSN Intrusion dataset [33] I s used to execute the results. Table.2 provides the simulation parameters.

Table 2: Simulation Parameters

Parameters	**Values**
Number of sensor nodes	100
Network area	1000×1000 m
Base station location	500×500
Maximum transmission range	150 m
Packet loss probability	0.1
Attack occurrence probability	0.05
Duration of attack	10 ms
Input feature dimensions	10
Channel type	Wireless
Routing protocol	AODV
Maximum packet queue	50
Simulation time	100 s

3.1. Estimation of Accuracy

The estimation of accuracy has considered both the predictive posterior distribution and uncertainty quantification. Let us consider $f_\theta(x)$ to denote the BNN with parameters θ treated as random variables following a prior distribution $p(\theta)$. Given input data x and true label y, the predictive posterior is obtained via Bayesian marginalization:

$$p(y|x, D) = \int p(y|x, \theta) * P(\theta|D) * d\theta \tag{12}$$

where D is the training dataset and $p(\theta \mid D)$ is the posterior distribution over parameters. Since exact inference is intractable, By Using MC sampling with T stochastic forward passes:

$$p(y \mid x, D) \approx \frac{1}{T} \sum_{t=1}^{T} p(y \mid x, \theta_t) \quad \text{where} \quad \theta_t \sim p(\theta \mid D) \tag{13}$$

The predicted class $\hat{y}$ is the maximum a posteriori (MAP) estimate the following :

$$\hat{y} = \arg\max_y \ p(y \mid x, D) \tag{14}$$

The accuracy is computed as the fraction of correct predictions over N test samples:

$$A = \frac{1}{N} \sum_{i=1}^{N} \vartheta(\hat{y}_i = y_i) \tag{15}$$

where $\vartheta(\cdot)$ is the indicator function.

The Expected Calibration Error (ECE) is computed as follows:

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{N} \left|acc(B_m) - conf(B_m)\right| \tag{16}$$

where B_m denotes bins partitioning confidence scores, $acc(B_m)$ is the accuracy in bin m, and $conf(B_m)$ is the average confidence.

The Brier Score (BS) assesses probabilistic prediction quality and is given by:

$$BS = \frac{1}{N} \sum_{i=1}^{N} \left(p(y_i \mid x_i, D) - \vartheta(y_i = \hat{y}_i)\right)^2 \tag{17}$$

The proposed BDL framework's accuracy is not only derived from classification performance but also validated through uncertainty calibration, ensuring robust intrusion detection in high-density WSNs under adversarial and noisy conditions. Table.3 shows the estimation of accuracy.

Fig.5 (a) shows the comparative analysis of accuracy. In a estimation tip, the proposed BDLF reached 99.5% accuracy. The existing UCBA reached 88.6%, FFA obtained 82.7%, EIDM reached 84.7%, LIDM obtained 79.8% and IDA reached 76.8% accuracy.

3.2. Estimation of Precision

The precision as the ratio of true positive prediction (TPP) to the sum of true positive prediction and false positive prediction (FPP).

$$P = \frac{TPP}{TPP + FPP} \tag{18}$$

In the Bayesian context, predictions are probabilistic, obtained via Monte Carlo (MC) sampling over T stochastic forward passes of the neural network with dropout enabled at test time. Let x denote the input data sample,

Table 3: Estimation of Accuracy

2*Sample Size	Accuracy					
	BDLF	**UCBA**	**FFA**	**EIDM**	**LIDM**	**IDA**
100	0.951	0.852	0.823	0.824	0.785	0.756
200	0.955	0.854	0.803	0.823	0.782	0.752
300	0.961	0.858	0.806	0.826	0.784	0.754
400	0.965	0.862	0.809	0.829	0.786	0.756
500	0.971	0.866	0.812	0.832	0.788	0.758
600	0.975	0.872	0.815	0.835	0.795	0.766
700	0.981	0.874	0.818	0.838	0.792	0.762
800	0.985	0.878	0.821	0.841	0.794	0.764
900	0.991	0.882	0.824	0.844	0.796	0.766
1000	0.995	0.886	0.827	0.847	0.798	0.768

and $y_i \in \{0, 1\}$ indicate whether it is an intrusion (1) or benign (0). The predictive mean for the intrusion class is the following:

$$p(y_i = 1 \mid x_i, D) \approx \frac{1}{T} \sum_{t=1}^{T} f_{\theta_t}(x_i) \tag{19}$$

where f_{θ_t} is the softmax output of the BNN for the intrusion class in the t-th MC sample. The final predicted label $\hat{y}_i$ is obtained via thresholding. To compute True Positive Predictions (TPP) and False Positive Predictions (FPP), aggregate predictions over the test set D_{test}:

$$TPP = \sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 1) \quad \text{and} \quad FPP = \sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 0) \tag{20}$$

The Bayesian precision (P) thus becomes the following:

$$P = \frac{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 1)}{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 1) + \sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 0)} \tag{21}$$

which simplifies to the standard precision expression:

$$P = \frac{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \land y_i = 1)}{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1)} \tag{22}$$

To incorporate uncertainty awareness, the classification threshold may adjust based on predictive variance or entropy, reducing FP by rejecting low-confidence predictions. This ensures higher precision in security-critical WSNs, where false alarms must be minimized. The derived precision, combined with recall and uncertainty metrics, validates the framework's reliability in adversarial environments. Table.4 shows the estimation of precision.

Table 4: Estimation of Precision

2***Sample Size**	**Precision**					
	BDLF	**UCBA**	**FFA**	**EIDM**	**LIDM**	**IDA**
100	0.931	0.832	0.783	0.843	0.765	0.736
200	0.935	0.834	0.783	0.803	0.762	0.732
300	0.941	0.838	0.786	0.806	0.764	0.734
400	0.945	0.842	0.789	0.809	0.766	0.736
500	0.951	0.846	0.792	0.812	0.768	0.738
600	0.955	0.852	0.795	0.815	0.774	0.746
700	0.961	0.854	0.798	0.818	0.772	0.742
800	0.965	0.858	0.801	0.821	0.774	0.744
900	0.971	0.862	0.804	0.824	0.776	0.746
1000	0.975	0.866	0.807	0.827	0.778	0.748

Fig.5 (b) shows the comparative analysis of precision. In a estimation tip, the proposed BDLF reached 97.5% precision. The existing UCBA reached 86.6%, FFA obtained 80.7%, EIDM reached 82.7%, LIDM obtained 77.8% and IDA reached 74.8% precision.

3.3. Estimation of Recall

The recall (R) as the ratio of true positive predictions (TPP) to the sum of true positive predictions and false negative predictions (FNP).

$$R = \frac{TPP}{TPP + FNP} \tag{23}$$

The predictions are obtained through MC sampling over T stochastic forward passes of the neural network. For a given input x_i, the predictive probability of the intrusion class ($y_i = 1$) is approximated as derived from Eq. 19. To compute TPP and FNP, predictions must be aggregated over the test set D_{test} as follows:

$$TPP = \sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 1) \qquad \text{and} \qquad FNP = \sum_{i=1}^{N} \vartheta(\hat{y}_i = 0 \wedge y_i = 1) \tag{24}$$

The Bayesian recall (R) thus becomes the following:

$$R = \frac{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 1)}{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 1) + \sum_{i=1}^{N} \vartheta(\hat{y}_i = 0 \wedge y_i = 1)} \tag{25}$$

which simplifies to:

$$R = \frac{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 1)}{\sum_{i=1}^{N} \vartheta(y_i = 1)} \tag{26}$$

To enhance uncertainty awareness, the threshold can be dynamically adjusted based on predictive uncertainty. For instance, in high-uncertainty scenarios, the system may defer decisions or trigger additional verification, ensuring robust intrusion detection while maintaining high recall. This approach is critical in high-density WSNs, where missed intrusions (FNP) can compromise security. The derived recall, combined with precision and uncertainty metrics, ensures the framework's effectiveness in detecting adversarial activities under uncertainty. Table.5 shows the estimation of recall.

Fig.5 (c) shows the comparative analysis of recall. In a estimation tip, the proposed BDLF reached 96.5% recall. The existing UCBA reached 83.6%, FFA obtained 77.7%, EIDM reached 80.7%, LIDM obtained 75.8% and IDA reached 71.8% recall.

3.4. Estimation of F1-Score

The F1-Score is the harmonic mean of precision (P) and recall (R), defined as follows:

$$F1\text{-}Score = 2 \times \frac{(P \times R)}{(P + R)} \tag{27}$$

Table 5: Estimation of Recall

2*Sample Size	Recall					
	BDLF	**UCBA**	**FFA**	**EIDM**	**LIDM**	**IDA**
100	0.921	0.822	0.753	0.784	0.745	0.766
200	0.925	0.804	0.753	0.783	0.742	0.702
300	0.931	0.808	0.756	0.786	0.744	0.704
400	0.935	0.812	0.759	0.789	0.746	0.706
500	0.941	0.816	0.762	0.792	0.748	0.708
600	0.945	0.822	0.765	0.795	0.753	0.714
700	0.951	0.824	0.768	0.798	0.752	0.712
800	0.955	0.828	0.771	0.801	0.754	0.714
900	0.961	0.832	0.774	0.804	0.756	0.716
1000	0.965	0.836	0.777	0.807	0.758	0.718

The precision is derived in Eq. 22 and recall is derived in Eq. 26. The predicted label $\hat{y}_i$ is obtained by thresholding the predictive probability:

$$\hat{y}_i = \vartheta(p(y_i = 1 \mid x_i, D) \geq \tau) \tag{28}$$

where τ is a decision threshold. The TPP, FPP, and FNP are obtained from Eq. 20 and Eq. 24. Now, the F1-Score has the following expanded form:

$$F1\text{-}Score = 2 \times \frac{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 1)}{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 1) + \sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 0) + \sum_{i=1}^{N} \vartheta(\hat{y}_i = 0 \wedge y_i = 1)} \tag{29}$$

To incorporate uncertainty awareness, the threshold τ can be adapted based on predictive uncertainty. For instance, samples with high uncertainty might be excluded from the F1-Score calculation or subjected to further scrutiny, ensuring robust performance in adversarial WSN environments. The derived F1Score, combined with uncertainty metrics, validates the framework's ability to balance precision and recall while enhancing security in high-density WSNs. Table.6 shows the estimation of F1Score.

3.5. Estimation of Detection Rate

The detection rate (DR) is the probability that an actual intrusion is correctly identified by the system. It is expressed as the following:

Table 6: Estimation of F1-Score

2*Sample Size	**F1-Score**					
	BDLF	**UCBA**	**FFA**	**EIDM**	**LIDM**	**IDA**
100	0.941	0.822	0.763	0.794	0.755	0.726
200	0.945	0.824	0.763	0.793	0.752	0.722
300	0.951	0.828	0.766	0.796	0.754	0.724
400	0.955	0.832	0.769	0.799	0.756	0.726
500	0.961	0.836	0.772	0.802	0.758	0.728
600	0.965	0.842	0.775	0.805	0.765	0.736
700	0.971	0.844	0.778	0.808	0.762	0.732
800	0.975	0.848	0.781	0.811	0.764	0.734
900	0.981	0.852	0.784	0.814	0.766	0.736
1000	0.985	0.856	0.787	0.817	0.768	0.738

$$DR = \frac{TPP}{TPP + FNP} \tag{30}$$

The TPP and FNP values are obtained from Eq. 20 and Eq. 24. Now, the estimation of DR has the following:

$$DR = \frac{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 1)}{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 1) + \sum_{i=1}^{N} \vartheta(\hat{y}_i = 0 \wedge y_i = 1)} \tag{31}$$

which simplifies to:

$$DR = \frac{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 1)}{\sum_{i=1}^{N} \vartheta(y_i = 1)} \tag{32}$$

In high-uncertainty scenarios, the system may lower the threshold to reduce false negatives, thereby improving the detection rate while maintaining robustness against adversarial attacks. This approach is particularly critical in high-density WSNs, where missed intrusions can have severe security implications. The derived DR, combined with precision and uncertainty metrics, ensures the framework's effectiveness in identifying intrusions under uncertain and adversarial conditions.

The Bayesian framework allows for the computation of uncertainty-aware detection rate DR_μ by considering only predictions with low uncertainty as shown in the following:

$$DR_{\mu} = \frac{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \wedge y_i = 1 \wedge \mu(x_i) \leq \epsilon)}{\sum_{i=1}^{N} \vartheta(y_i = 1)} \tag{33}$$

Where, $H(x_i)$ is the entropy of the predictive distribution for sample x_i. This refinement ensures that the reported DR reflects confident and reliable detections, further enhancing the security of the WSN. The integration of uncertainty quantification into the DR metric underscores the framework's capability to provide trustworthy intrusion detection in dynamic and resource-constrained environments.

Table.7 shows the estimation of estimation of detection rate.

Table 7: Estimation of Detection Rate

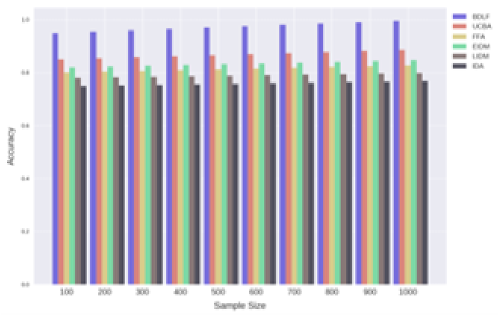
2***Sample Size**	**Detection Rate**					
	BDLF	**UCBA**	**FFA**	**EIDM**	**LIDM**	**IDA**
100	0.911	0.792	0.743	0.774	0.735	0.766
200	0.915	0.794	0.743	0.773	0.732	0.702
300	0.921	0.798	0.746	0.776	0.734	0.704
400	0.925	0.802	0.749	0.779	0.736	0.706
500	0.931	0.806	0.752	0.782	0.738	0.708
600	0.935	0.812	0.755	0.785	0.743	0.713
700	0.941	0.814	0.758	0.788	0.742	0.712
800	0.945	0.818	0.761	0.791	0.744	0.714
900	0.951	0.822	0.764	0.794	0.746	0.716
1000	0.955	0.826	0.767	0.797	0.748	0.718

Fig.5 (e) shows the comparative analysis of detection rate. In a estimation tip, the proposed BDLF reached 95.5% detection rate. The existing UCBA reached 82.6%, FFA obtained 76.7%, EIDM reached 79.7%, LIDM obtained 74.8% and IDA reached 71.8% detection rate.

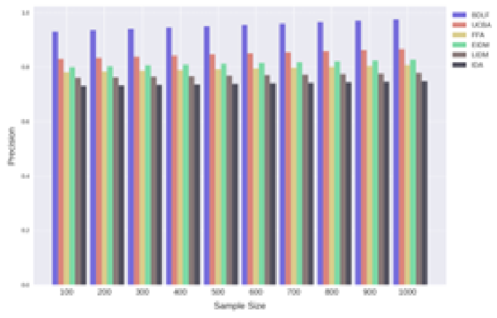
3.6. Estimation of False Alarm Rate

The false alarm rate (FAR) as the probability that a benign instance is incorrectly classified as an intrusion. It is expressed as the following,

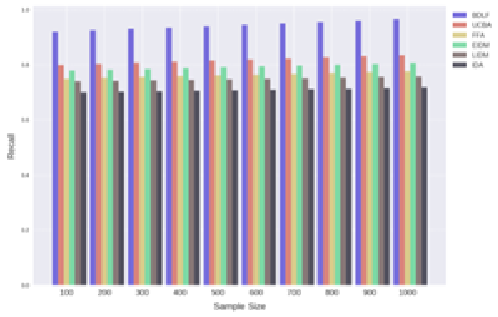
$$FAR = \frac{FPP}{FPP + TNP} \tag{34}$$



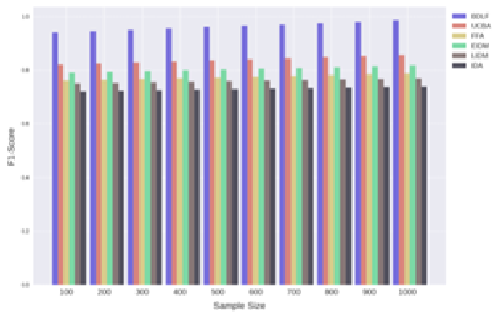
(a) Estimation of Accuracy (A)



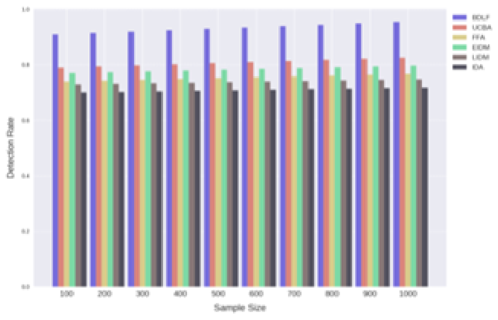
(b) Estimation of Precision (P)



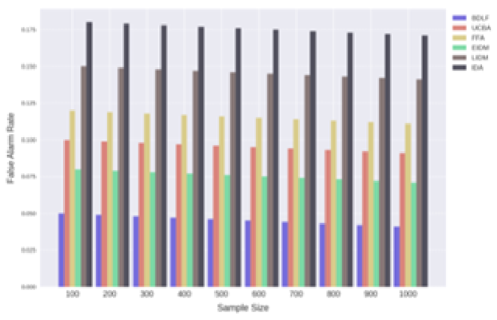
(c) Estimation of Recall (R)



(d) Estimation of F1-Score



(e) Estimation of Detection Rate (DR)



(f) Estimation of False Alarm Rate (FAR)

Figure 5: Comparative analysis of performance metrics across different sample sizes for intrusion detection models.

The FPP and TNP values are obtained from Eq. 20 and Eq. 24. Now, the estimation of FAR has the following form:

$$FAR = \frac{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \land y_i = 0)}{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \land y_i = 0) + \sum_{i=1}^{N} \vartheta(\hat{y}_i = 0 \land y_i = 0)} \tag{35}$$

which simplifies to:

$$FAR = \frac{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \land y_i = 0)}{\sum_{i=1}^{N} \vartheta(y_i = 0)} \tag{36}$$

In high-uncertainty scenarios, the system may increase the threshold to reduce false positives, thereby lowering the FAR while maintaining detection accuracy. This adjustment is critical in high-density WSNs, where false alarms can lead to unnecessary resource consumption and undermine trust in the intrusion detection system.

The Bayesian framework also enables the computation of uncertainty-aware FAR (FAR_u) by considering only predictions with low uncertainty:

$$FAR_U = \frac{\sum_{i=1}^{N} \vartheta(\hat{y}_i = 1 \land y_i = 0 \land \mu(x_i) \leq \epsilon)}{\sum_{i=1}^{N} \vartheta(y_i = 0)} \tag{37}$$

This refinement ensures that the reported FAR reflects confident and reliable classifications, further enhancing the security and efficiency of the WSN. The integration of uncertainty quantification into the FAR metric underscores the framework's capability to minimize false alarms while maintaining robust intrusion detection in dynamic and adversarial environments. Table.8 shows the estimation of false alarm rate.

Fig.5 (f) shows the comparative analysis of false alarm rate. In a estimation tip, the proposed BDLF reached 4.1% false alarm. The existing UCBA reached 9.1%, FFA obtained 11.1%, EIDM reached 7.1%, LIDM obtained 14.1% and IDA reached 17.1% false alarm.

4. Discussion

The predictive entropy (PE) quantifies the uncertainty in the model's predictions by measuring the information content of the predictive posterior distribution. For a test samplex_t, the PE is derived from the MC-sampled class probabilities has the following,

Table 8: Estimation of False Alarm Rate

2*Sample Size	**False Alarm Rate**					
	BDLF	**UCBA**	**FFA**	**EIDM**	**LIDM**	**IDA**
100	0.051	0.122	0.123	0.084	0.155	0.186
200	0.049	0.099	0.119	0.079	0.149	0.179
300	0.047	0.101	0.121	0.081	0.151	0.181
400	0.045	0.103	0.123	0.083	0.153	0.183
500	0.043	0.105	0.125	0.085	0.155	0.185
600	0.041	0.107	0.127	0.087	0.157	0.187
700	0.039	0.109	0.129	0.089	0.159	0.189
800	0.037	0.111	0.131	0.091	0.161	0.191
900	0.035	0.113	0.133	0.093	0.163	0.193
1000	0.041	0.091	0.111	0.071	0.141	0.171

$$p(y_i = k \mid x_i, D) \approx \frac{1}{T} \sum_{t=1}^{T} f_{\theta_t}(x_i)_k \tag{38}$$

$$PE(x_i) = -\sum_{k=1}^{K} p(y_i = k \mid x_i, D) \log p(y_i = k \mid x_i, D) \tag{39}$$

where K is the number of classes. High predictive entropy (PE) indicates ambiguous or uncertain predictions, which is critical for flagging suspicious activities in WSNs.

The Brier Score (BS) evaluates the accuracy of probabilistic predictions by measuring the mean squared error between predicted probabilities and true labels. For binary intrusion detection ($K = 2$):

$$BS = \frac{1}{N} \sum_{i=1}^{N} \left(p(y_i = 1 \mid x_i, D) - \vartheta(y_i = 1)\right)^2 \tag{40}$$

where $\vartheta(\cdot)$ is the indicator function. A lower BS indicates better-calibrated predictions, which is essential for reliable threat assessment.

ECE assesses how well the model's confidence aligns with its accuracy. Predictions are binned into M intervals Bm based on confidence scores (e.g., $p(y_i = 1 \mid x_i, D)$). ECE is computed as:

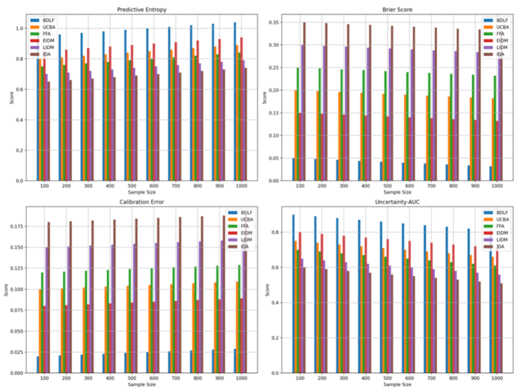


Figure 6: Performance analysis

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{N} \left| acc(B_m) - conf(B_m) \right| \tag{41}$$

Where, $acc(B_m)$ and $conf(B_m)$ are the accuracy and average confidence in bin B_m, respectively. Low ECE ensures that high-confidence predictions are trustworthy, which is vital for security decisions.

Uncertainty AUC (UAUC) evaluates the model's ability to rank uncertain samples higher than certain ones. Let $H(x_i)$ be the entropy of x_i, and $E_i \in \{0, 1\}$ indicate whether the prediction was incorrect (1) or correct (0). The UAUC is the area under the ROC curve plotting the true positive rate (uncertainty for errors) against the false positive rate (uncertainty for correct predictions) across varying entropy thresholds.

$$UAUC = \int_0^1 TPR(h) \cdot FPR(h)\, dh \tag{42}$$

Where,

$$TPR(h) = P(H(x_i) \geq h \mid E_i = 1)$$

$$FPR(h) = P(H(x_i) \geq h \mid E_i = 0)$$

A $UAUC > 0.5$ indicates that the model effectively uses uncertainty to flag errors. Table 9 shows the estimation of predictive entropy.

Fig.6 (a) shows the performance analysis of predictive entropy. In a estimation tip, the proposed BDLF reached 1.04 predictive entropy. The existing UCBA reached 0.89, FFA obtained 0.84, EIDM reached 0.94, LIDM obtained

Table 9: Estimation of Predictive Entropy

2*Sample Size	Predictive Entropy					
	BDLF	**UCBA**	**FFA**	**EIDM**	**LIDM**	**IDA**
100	0.95	0.80	0.75	0.85	0.70	0.65
200	0.96	0.81	0.76	0.86	0.71	0.66
300	0.97	0.82	0.77	0.87	0.72	0.67
400	0.98	0.83	0.78	0.88	0.73	0.68
500	0.99	0.84	0.79	0.89	0.74	0.69
600	1.00	0.85	0.80	0.90	0.75	0.70
700	1.01	0.86	0.81	0.91	0.76	0.71
800	1.02	0.87	0.82	0.92	0.77	0.72
900	1.03	0.88	0.83	0.93	0.78	0.73
1000	1.04	0.89	0.84	0.94	0.79	0.74

0.79 and IDA reached 0.74 predictive entropy. Table.10 shows the estimation of brier score.

Table 10: Estimation of Brier Score

Sample Size	**BDLF**	**UCBA**	**FFA**	**EIDM**	**LIDM**	**IDA**
100	0.050	0.200	0.250	0.150	0.300	0.350
200	0.048	0.198	0.248	0.148	0.298	0.348
300	0.046	0.196	0.246	0.146	0.296	0.346
400	0.044	0.194	0.244	0.144	0.294	0.344
500	0.042	0.192	0.242	0.142	0.292	0.342
600	0.040	0.190	0.240	0.140	0.290	0.340
700	0.038	0.188	0.238	0.138	0.288	0.338
800	0.036	0.186	0.236	0.136	0.286	0.336
900	0.034	0.184	0.234	0.134	0.284	0.334
1000	0.032	0.182	0.232	0.132	0.282	0.332

Fig.6 (b) shows the performance analysis of brier score. In a estimation tip, the proposed BDLF reached 0.032 brier score. The existing UCBA reached 0.182, FFA obtained 0.232, EIDM reached 0.132, LIDM obtained 0.282 and IDA reached 0.332 brier score. Table.11 shows the estimation of estimated calibration error.

Table 11: Estimation of Estimated Calibration Error

Sample Size	BDLF	UCBA	FFA	EIDM	LIDM	IDA
100	0.02	0.10	0.12	0.08	0.15	0.18
200	0.021	0.101	0.121	0.081	0.151	0.181
300	0.022	0.102	0.122	0.082	0.152	0.182
400	0.023	0.103	0.123	0.083	0.153	0.183
500	0.024	0.104	0.124	0.084	0.154	0.184
600	0.025	0.105	0.125	0.085	0.155	0.185
700	0.026	0.106	0.126	0.086	0.156	0.186
800	0.027	0.107	0.127	0.087	0.157	0.187
900	0.028	0.108	0.128	0.088	0.158	0.188
1000	0.029	0.109	0.129	0.089	0.159	0.189

Fig.6 (c) shows the performance analysis of estimated calibration error. In a estimation tip, the proposed BDLF reached 0.029 estimated calibration error. The existing UCBA reached 0.109, FFA obtained 0.129, EIDM reached 0.089, LIDM obtained 0.159 and IDA reached 0.189 estimated calibration error. Table.12 shows the estimation of Uncertainty AUC.

Table 12: Estimation of Uncertainty AUC

Sample Size	BDLF	UCBA	FFA	EIDM	LIDM	IDA
100	0.90	0.75	0.70	0.80	0.65	0.60
200	0.89	0.74	0.69	0.79	0.64	0.59
300	0.88	0.73	0.68	0.78	0.63	0.58
400	0.87	0.72	0.67	0.77	0.62	0.57
500	0.86	0.71	0.66	0.76	0.61	0.56
600	0.85	0.70	0.65	0.75	0.60	0.55
700	0.84	0.69	0.64	0.74	0.59	0.54
800	0.83	0.68	0.63	0.73	0.58	0.53
900	0.82	0.67	0.62	0.72	0.57	0.52
1000	0.81	0.66	0.61	0.71	0.56	0.51

Fig.6 (d) shows the performance analysis of Uncertainty AUC. In a estimation tip, the proposed BDLF reached 0.81 Uncertainty AUC. The existing UCBA reached 0.66, FFA obtained 0.61, EIDM reached 0.71, LIDM obtained 0.56 and IDA reached 0.51 Uncertainty AUC.

5. Conclusion

The proposed Bayesian Deep Learning framework provides a robust, probabilistic approach to enhancing security by quantifying and leveraging predictive uncertainty. The BDL framework employs Monte Carlo (MC) dropout and Bayesian neural networks to generate stochastic forward passes, enabling the estimation of predictive entropy, variance, and confidence intervals alongside intrusion classifications. This allows the system to distinguish between certain and uncertain predictions, flagging ambiguous samples for further analysis—critical in adversarial and noisy WSN environments where false positives and negatives can degrade security. The model achieved 99.5% accuracy, 97.5% precision, 96.5% recall, 98.5% f1-score, 95.5% detection rate and 4.1% false alarm rate. The framework outperforms deterministic deep learning models in detecting novel and adversarial attacks while maintaining computational efficiency. Future work may explore distributed Bayesian inference for scalability in ultra-dense WSNs and adversarial training to further harden the model against evasion attacks. Ultimately, this approach establishes a trustworthy, adaptive, and resilient intrusion detection system crucial for securing next-generation IoT and industrial WSN deployments.

Declarations

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Conflicts of interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethical approval

This article does not contain any studies with human participants or animals performed by any of the authors.

Authors' contributions

Author 1: Conceptualization, methodology, model implementation, experiments, and writing – original draft.
Author 2: Supervision, validation, critical review, and editing.

References

[1] W. Liang, Y. Li, J. Xu, Z. Qin, D. Zhang, and K. C. Li, "QoS prediction and adversarial attack protection for distributed services under DlaaS," *IEEE Transactions on Computers*, vol. 73, no. 3, pp. 669–682, 2023.

[2] M. Gazzan and F. T. Sheldon, "Novel ransomware detection exploiting uncertainty and calibration quality measures using deep learning," *Information*, vol. 15, no. 5, p. 262, 2024.

[3] S. Y. Kuo, Y. H. Chou, and C. Y. Chen, "Quantum-inspired algorithm for cyber-physical visual surveillance deployment systems," *Computer Networks*, vol. 117, pp. 5–18, 2017.

[4] Z. Fei, B. Li, S. Yang, C. Xing, H. Chen, and L. Hanzo, "A survey of multi-objective optimization in wireless sensor networks: Metrics, algorithms, and open problems," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 550–586, 2016.

[5] S. Ismail, K. Shah, H. Reza, R. Marsh, and E. Grant, "Toward management of uncertainty in self-adaptive software systems: IoT case study," *Computers*, vol. 10, no. 3, p. 27, 2021.

[6] V. P. Illiano, A. Paudice, L. Munoz-González, and E. C. Lupu, "Determining resilience gains from anomaly detection for event integrity in wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 14, no. 1, pp. 1–35, 2018.

[7] A. P. Vedurmuidi, K. Milicevic, G. Kok, B. X. Yong, L. Xu, G. Zheng, *et al.*, "Automation in sensor network metrology: An overview of methods and their implementations," *Measurement: Sensors*, p. 101799, 2025.

[8] S. Ahamad and R. Gupta, "Uncertainty modelling in performability prediction for safety-critical systems," *Arabian Journal for Science and Engineering*, pp. 1–15, 2024.

[9] C. Nakas, D. Kandris, and G. Visvardis, "Energy efficient routing in wireless sensor networks: A comprehensive survey," *Algorithms*, vol. 13, no. 3, p. 72, 2020.

[10] A. A. Mehta, A. A. Padaria, D. J. Bavisi, V. Ukani, P. Thakkar, R. Geddam, *et al.*, "Securing the future: A comprehensive review of security challenges and solutions in advanced driver assistance systems," *IEEE Access*, vol. 12, pp. 643–678, 2023.

[11] M. Wei, C. Rong, E. Liang, and Y. Zhuang, "An intrusion detection mechanism for IPv6-based wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 18, no. 3, p. 15501329221077922, 2022.

[12] G. G. Gebremariam, J. Panda, and S. J. C. S. Indu, "Design of advanced intrusion detection systems based on hybrid machine learning techniques in hierarchically wireless sensor networks," *Connection Science*, vol. 35, no. 1, p. 2246703, 2023.

[13] M. R. Senouci and A. Mellouk, "A robust uncertainty-aware cluster-based deployment approach for WSNs: Coverage, connectivity, and lifespan," *Journal of Network and Computer Applications*, vol. 146, p. 102414, 2019.

[14] J. Talpini, F. Sartori, and M. Savi, "Enhancing trustworthiness in ML-based network intrusion detection with uncertainty quantification," *Journal of Reliable Intelligent Environments*, vol. 10, no. 4, pp. 501–520, 2024.

[15] R. Garg, T. Gulati, and S. Kumar, "Wormhole attack detection and recovery for secure range free localization in large-scale wireless sensor networks," *Peer-to-Peer Networking and Applications*, vol. 16, no. 6, pp. 2833–2849, 2023.

[16] M. Karthikeyan, D. Manimegalai, and K. RajaGopal, "Firefly algorithm based WSN-IoT security enhancement with machine learning for intrusion detection," *Scientific Reports*, vol. 14, no. 1, p. 231, 2024.

[17] G. Liu, H. Zhao, F. Fan, G. Liu, Q. Xu, and S. Nazir, "An enhanced intrusion detection model based on improved kNN in WSNs," *Sensors*, vol. 22, no. 4, p. 1407, 2022.

[18] J. S. Pan, F. Fan, S. C. Chu, H. Q. Zhao, and G. Y. Liu, "A lightweight intelligent intrusion detection model for wireless sensor networks," *Security and Communication Networks*, vol. 2021, no. 1, p. 5540895, 2021.

[19] K. Ramana, A. Revathi, A. Gayathri, R. H. Jhaveri, C. L. Narayana, and B. N. Kumar, "WOGRU-IDS—An intelligent intrusion detection system for IoT assisted wireless sensor networks," *Computer Communications*, vol. 196, pp. 195–206, 2022.

[20] H. Sadia, S. Farhan, Y. U. Haq, R. Sana, T. Mahmood, S. A. O. Bahaj, and A. R. Khan, "Intrusion detection system for wireless sensor networks: A machine learning based approach," *IEEE Access*, vol. 12, pp. 52565–52582, 2024.

[21] M. S. Alsahli, M. M. Almasri, M. Al-Akhras, A. I. Al-Issa, and M. Alawairdhi, "Evaluation of machine learning algorithms for intrusion detection system in WSN," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 5, 2021.

[22] Z. Jingjing, Y. Tongyu, Z. Jilin, Z. Guohao, L. Xuefeng, and P. Xiang, "Intrusion detection model for wireless sensor networks based on MC-GRU," *Wireless Communications and Mobile Computing*, vol. 2022, no. 1, p. 2448010, 2022.

[23] A. Arkan and M. Ahmadi, "An unsupervised and hierarchical intrusion detection system for software-defined wireless sensor networks," *The Journal of Supercomputing*, vol. 79, no. 11, pp. 11844–11870, 2023.

[24] M. A. Talukder, S. Sharmin, M. A. Uddin, M. M. Islam, and S. Aryal, "MLSTL-WSN: Machine learning-based intrusion detection using SMOTETomek in WSNs," *International Journal of Information Security*, vol. 23, no. 3, pp. 2139–2158, 2024.

[25] H. M. Saleh, H. Marouane, and A. Fakhfakh, "Stochastic gradient descent intrusions detection for wireless sensor network attack detection system using machine learning," *IEEE Access*, vol. 12, pp. 3825–3836, 2024.

[26] O. Ahmed, "Enhancing intrusion detection in wireless sensor networks through machine learning techniques and context awareness integration," *International Journal of Mathematics, Statistics, and Computer Science*, vol. 2, pp. 244–258, 2024.

[27] S. Subramani and M. Selvi, "Multi-objective PSO based feature selection for intrusion detection in IoT based wireless sensor networks," *Optik*, vol. 273, p. 170419, 2023.

[28] M. Safaldin, M. Otair, and L. Abualigah, "Improved binary gray wolf optimizer and SVM for intrusion detection system in wireless sensor networks," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, pp. 1559–1576, 2021.

[29] J. Jin, "Intrusion detection algorithm and simulation of wireless sensor network under Internet environment," *Journal of Sensors*, vol. 2021, no. 1, p. 9089370, 2021.

[30] K. Madhuri, "A new level intrusion detection system for node level drop attacks in wireless sensor network," *Journal of Algebraic Statistics*, vol. 13, no. 1, pp. 159–168, 2022.

[31] M. Otair, O. T. Ibrahim, L. Abualigah, M. Altalhi, and P. Sumari, "An enhanced grey wolf optimizer-based particle swarm optimizer for intrusion detection system in wireless sensor networks," *Wireless Networks*, vol. 28, no. 2, pp. 721–744, 2022.

[32] K. Hussain, Y. Xia, A. N. Onaizah, T. Manzoor, and K. Jalil, "Hybrid of WOA-ABC and proposed CNN for intrusion detection system in wireless sensor networks," *Optik*, vol. 271, p. 170145, 2022.