Volume (2) No (1), 2026
Journal of Global Research in Multidisciplinary Studies (JGRMS)
Review Paper/Research Paper
Available online at https://saanvipublications.com/journals/index.php/jgrms/index

AI-Based Ensemble Learning Framework For Cyber Threat Detection And Response

Smruti Sharma
MTech Scholar, CSE Department, Eklavya University
Damoh (M.P.), India
smrutisharma930@gmail.com

Dr. Prashant Sen
Prof & Head, CSE Department, Eklavya University,
Damoh (M.P.), India
prashant.sen@eklavyauniversity.ac.in

***Abstract*—The rapid digitization of critical services in India has significantly expanded the cyber-attack surface, rendering traditional security mechanisms inadequate against sophisticated modern threats. This paper presents a comprehensive AI-Based Ensemble Learning Framework for cyber threat detection and response, specifically designed to address the unique challenges of the Indian cybersecurity ecosystem. The framework integrates four complementary models—Random Forest, XGBoost, Deep Neural Networks (DNN), and Long Short-Term Memory (LSTM) networks—through a stacking ensemble approach with Logistic Regression as the meta-learner [27], [28], [30], [31]. Evaluation across four standard benchmark datasets—CIC-IDS2017, UNSW-NB15, NSL-KDD, and ICSCASD-MPLC—demonstrates detection accuracy ranging from 93% to 98% with false positive rates as low as 1.6% [23]-[26]. The system achieves near real-time performance with an average detection latency of 112 ms, substantially improving upon traditional Security Operations Center (SOC) metrics: mean time to detect (MTTD) reduced from 8-24 hours to 10-30 minutes, mean time to respond (MTTR) reduced from 24-72 hours to 15-60 minutes, and alert volume reduced from over 100,000 to 2,000-5,000 alerts per day [4], [17], [19]. The framework's performance on Industrial Control System (ICS) data is particularly encouraging, achieving 97.2% accuracy, addressing a critical gap in critical infrastructure protection [26], [35]. This work contributes a practical, deployable solution for modernizing cybersecurity operations in resource-constrained environments.**

***Keywords*—Artificial Intelligence, Ensemble Learning, Cyber Threat Detection, Intrusion Detection System, Stacking Ensemble, Random Forest, XGBoost, LSTM, Indian Cybersecurity**

I. INTRODUCTION

A. The Digital Transformation of India and Its Associated Risks

India's digital ecosystem is expanding at an unprecedented rate, fueled by initiatives such as Digital India, the widespread adoption of UPI for payments, and the establishment of the Aadhaar and National Digital Health Mission platforms [1], [8]. This surge in connectivity has brought millions of new users online and digitized critical services. However, this rapid expansion correspondingly enlarges the attack surface for malicious actors. The Indian Computer Emergency Response Team (CERT-In) documented a staggering 1.4 million cybersecurity incidents in 2023 [1], illustrating a sharp increase in malicious activities that have evolved beyond simple viruses to sophisticated data theft and service disruption campaigns.

TABLE I. CYBERSECURITY STATISTICS FOR INDIA (2023-2025)

Statistic	Value	Source
Cybersecurity incidents reported in 2023	1.4 million+	CERT-In [1]
Increase in ransomware attacks (2023–2024)	38%	DSCI [8]
Organizations that had at least one cyber incident	45%	PwC India [9]
Projected cybersecurity market size by 2025	INR 3,000 crore	NASSCOM [10]
Estimated shortage of cybersecurity professionals	3 million	TeamLease [11]
Average data breach cost in India	₹5–8 crore	IBM Security

B. The Evolving Nature of Cyber Threats

The sophistication of modern cyber-attacks has escalated significantly. Attackers now employ methods such as Advanced Persistent Threats (APTs), which involve prolonged, stealthy infiltration of networks to exfiltrate sensitive data without detection [2][36] Ransomware has similarly evolved; contemporary variants can modify their own code to evade signature-based detection and even leverage AI to target the most critical files for encryption, maximizing impact [3]. Furthermore, the attack surface has broadened to include software supply chains, where malicious code is injected into legitimate products, and vulnerable Internet of Things (IoT) and Industrial Control Systems (ICS) devices [15], [26].

C. Challenges in Indian Cybersecurity

Indian organizations face significant challenges in protecting their digital assets [20], [35]:

- **Alert Fatigue:** Security Operations Centers (SOCs) receive over 100,000 alerts daily, with more than 70% being false positives [4], [17], [18]. Analysts spend 2-4 hours daily just validating alerts, and approximately 70% of alerts are never investigated due to overwhelming volume
- **Workforce Shortage:** India faces a shortage of approximately 3 million cybersecurity professionals by 2025, leading to overworked existing staff and high turnover rates [5], [11].
- **Diverse Technology Stack:** Indian organizations operate heterogeneous environments combining legacy systems with modern cloud infrastructure,

creating complex security landscapes that require varied protection mechanisms[38]

- **Resource Constraints:** Many Indian organizations, especially small and medium enterprises, cannot afford expensive security solutions and require cost-effective alternatives.
- **Critical Infrastructure Vulnerability:** Essential services like power grids, banks, hospitals, and railways are increasingly digitized, making them attractive targets for attacks that could cause real-world damage [6], [35].

D. The Role of AI as a Solution

In response to these advanced threats, Artificial Intelligence (AI) presents a paradigm shift in cybersecurity capabilities [2], [16]. AI-powered systems excel at processing massive datasets in real-time, identifying subtle anomalies and complex attack patterns far more efficiently than manual analysis. Their ability to learn from data allows them to reduce the high volume of false positives, which is a known source of analyst burnout. Furthermore, AI's adaptability ensures that it can be updated to recognize new attack vectors continuously, providing 24/7 automated protection.

II. Problem Statement

Traditional cybersecurity methods are no longer sufficient to stop modern cyber-attacks [16], [17]. Signature-based systems only detect known attacks and fail to identify approximately 43% of new and evolving attack patterns [16]. Rule-based systems use fixed rules that hackers can learn to bypass, generating too many false alarms. Zero-day vulnerabilities are exploited before signatures exist, and multi-stage attacks are difficult to detect with simple rules.

Security teams face an overwhelming number of alerts, with enterprise SOCs processing over 100,000 alerts daily [17]. More than 70% of alerts are false positives or low-priority [18], leading to analyst burnout, missed critical threats, inconsistent decision-making, and wasted resources.

Manual incident response processes are too slow. Traditional Security Orchestration, Automation, and Response (SOAR) platforms automate predefined workflows but struggle with complex investigations [19].[39] Static playbooks need constant updates, and complex threats require manual analysis, allowing attackers to achieve their objectives before containment.

The cybersecurity workforce shortage exacerbates these problems, with 3.5 million unfilled cybersecurity positions globally and 3 million shortage in India by 2025 [5], [11]. Teams spend more time on alerts than on real investigation, with no time for proactive threat hunting.

Given these challenges, there is an urgent need for [21], [34]:

- **Intelligent Detection:** AI-driven systems that detect both known and novel threats with high accuracy and low false positives.
- **Automated Response:** Systems that investigate and respond to threats at machine speed.
- **Explainability:** Transparent systems that build analyst trust and enable human-AI collaboration.
- **Indian Solutions:** Approaches addressing the unique challenges of the Indian cybersecurity ecosystem.

III. Research Objectives

Primary Objectives:

- **O1:** To systematically review and categorize AI techniques for cyber threat detection and response, focusing on machine learning, deep learning, and ensemble approaches for Indian cybersecurity contexts.
- **O2:** To develop an AI-based ensemble learning framework for cyber threat detection and response.
- **O3:** To empirically evaluate AI model performance using standard cybersecurity datasets including CIC-IDS2017, NSL-KDD, UNSW-NB15, and ICS-specific datasets [23]-[26].
- **O4:** To compare the performance of individual AI models versus ensemble approaches for intrusion detection [6], [14].
- **O5:** To identify critical research gaps and propose future research directions for AI-driven cybersecurity in India.

Secondary Objectives:

- Develop a comprehensive taxonomy of AI techniques for cybersecurity
- Analyze the applicability of different AI techniques to Indian cybersecurity challenges
- Evaluate the trade-offs between detection accuracy, computational efficiency, and explainability
- Assess the operational implications of AI-driven security in Indian SOC environments

IV. Literature Review

A. Evolution of Intrusion Detection Systems

The development of Intrusion Detection Systems (IDS) has progressed through several distinct technological phases over the last forty years [16]-[19].

First Generation (1980s-1990s) - Signature-Based: The foundational method relied on a database of known attack patterns ("signatures") to identify threats [16]. While effective for known attacks with a low false positive rate, it was fundamentally incapable of detecting novel, zero-day threats or variants that had altered their code.

Second Generation (1990s-2000s) - Rule-Based: This approach offered more flexibility by using a set of expert-defined rules to define suspicious behaviors [17]. However, these static rules were easily bypassed by attackers and required constant maintenance to avoid generating an overwhelming number of false positives.

Third Generation (2000s-2010s) - Anomaly-Based: A significant shift occurred towards behavior-based detection, which establishes a baseline of "normal" activity and flags deviations [18]. This method can detect novel threats but is challenged by a high false positive rate, especially during initial deployment.

Fourth Generation (2010s-2020s) - Machine Learning-Based: Machine learning algorithms learn patterns from labeled or unlabeled data to make predictions [19]. Common algorithms include Random Forest, Support Vector Machines (SVM), XGBoost, and Decision Trees [27], [28].

Fifth Generation (2020s-Present) - Deep Learning-Based: Deep learning uses neural networks with multiple layers to

automatically learn hierarchical representations of data [29]. Common architectures include Deep Neural Networks (DNN), Convolutional Neural Networks (CNN), and Long Short-Term Memory (LSTM) networks [12], [30].

Sixth Generation (Emerging) - Ensemble Learning: Ensemble methods combine multiple models to achieve better performance than any individual model [20], [31]. This is the approach adopted in this paper.

B. Machine Learning Based Studies

Random Forest for Intrusion Detection: Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the class that is the mode of the classes of individual trees [27]. An optimized Random Forest framework achieved 98.14% accuracy on CIC-IDS2017 with 96.25% weighted F1-Score [9]. Feature reduction reduced dimensionality by 58.3% and training time by 63.2%. However, the study did not address real-time response capabilities or SOC integration.

XGBoost for Intrusion Detection: XGBoost (Extreme Gradient Boosting) is an optimized gradient boosting algorithm [28]. The NetGuard-X framework achieved 98.99% accuracy on UNSW-NB15 with significantly lower computational overhead than other predictive models [10]. However, the framework was evaluated only on the UNSW-NB15 dataset.

C. Deep Learning Based Studies

LSTM Networks for Intrusion Detection: LSTM networks are designed to learn long-term dependencies in sequential data [30]. Comprehensive comparative analysis showed LSTM achieving 99.80% accuracy on NSL-KDD, 99.76% on CICIDS2017, and 97.14% on CICIDS2018 [12]. However, computational requirements and training time were significantly higher than traditional ML approaches.

LSTM Autoencoder for CICIDS2017: Researchers achieved 97.4% accuracy, 96.2% recall, and 95.8% F1-Score [4]. The study primarily focused on evaluation metrics without addressing real-time performance or deployment challenges.

DNN and RNN-Based Intrusion Detection: DNN and RNN models achieved >99% accuracy on KDDCup99 and NSL-KDD datasets [11]. However, performance on more complex datasets like UNSW-NB15 was comparatively lower, indicating generalization challenges.

D. Ensemble Learning Based Studies

HAMC-ID - Two-Level Stacking Ensemble: Researchers introduced a two-level stacking ensemble framework with Level-0 using XGBoost, Extra Trees, and Logistic Regression, and Level-1 using Bidirectional LSTM with attention mechanism [14]. The framework showed superior performance over individual classifiers but required significant computational resources.

HHO-GWO Feature Selection with Stacking Ensemble: A hybrid approach combined Harris Hawks Optimization (HHO) and Grey Wolf Optimization (GWO) for feature selection with a stacking ensemble classifier [6]. Achieved 98% accuracy on NSL-KDD with up to 84% feature reduction. The swarm intelligence algorithms added computational overhead during feature selection.

E. Benchmark Datasets

- **CIC-IDS2017:** Created by the Canadian Institute for Cybersecurity, this dataset contains 2.8+ million records with 80+ network flow features and multiple attack scenarios including DDoS, Brute Force, Web Attacks, Infiltration, Botnet, and Port Scan [23].
- **UNSW-NB15:** Offers modern attack patterns with 2.5+ million records, 49 network features, and nine attack categories including Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms [24].
- **NSL-KDD:** An improved version of the KDDCup99 dataset providing a standardized baseline with 125,000+ records, 41 network features, and five attack classes [25].
- **ICSCASD-MPLC:** Captures realistic ICS network traffic from a physical testbed using Mitsubishi PLCs with 2.6+ million records, 57 features, and ICS-specific attacks [26].

F. Research Gaps

The systematic review revealed several significant gaps in the current body of research:

- **Gap 1:** Lack of Context for Indian Enterprises: Most studies fail to consider the specific challenges of Indian organizations, such as legacy and modern system integration, budgetary limitations, and compliance with local regulations [20].
- **Gap 2:** Limited Focus on ICS Security: While general network intrusion detection is well-studied, the specialized field of ICS security remains underexplored [26], despite the criticality of such infrastructure in India [35].
- **Gap 3:** Neglect of Real-Time Performance: Many studies prioritize accuracy without adequately considering detection latency and response time constraints vital for effective automated response [19].
- **Gap 4:** Insufficient False Positive Mitigation: There is a lack of research dedicated to operational strategies, such as confidence-based filtering, to manage the impact of false positives on SOC efficiency [18].
- **Gap 5:** Poor Integration with SOC Workflows: A significant disconnect exists between detection research and implementation into comprehensive SOC workflows, including automated response mechanisms [19].

V. Proposed Framework

A. Foundational Design Principles

The framework's architecture is underpinned by several core tenets, derived from the gaps identified in the literature:

- **Prioritizing Detection Accuracy:** The primary objective is to maximize detection of genuine threats while minimizing missed attacks. This is achieved through an ensemble methodology, as prior research has consistently demonstrated superiority over single-model approaches, often yielding a 1-4% improvement in F1-score [6], [14].
- **Minimizing False Alarms:** A critical design goal is to drastically lower false positive rates (FPR) to combat alert fatigue in SOC environments, where

over 70% of alerts can be false [18]. This is managed through mechanisms like confidence-based filtering.

- **Enabling Real-Time Performance:** The system is designed to operate with minimal latency to facilitate immediate response, a requirement often unmet by traditional systems, allowing attackers to achieve their goals [19].
- **Addressing Indian Contextual Needs:** The framework is specifically designed to be compatible with the heterogeneous technological landscapes, resource constraints, and critical infrastructure sectors (power, healthcare, banking) prevalent in India [20].
- **Ensuring Explain ability and Trust:** The framework provides interpretable models and explanations for detection decisions to build analyst trust and enable human-AI collaboration [21], [34].

B. Architectural Overview

The proposed AI-Based Ensemble Learning Framework consists of four main layers:

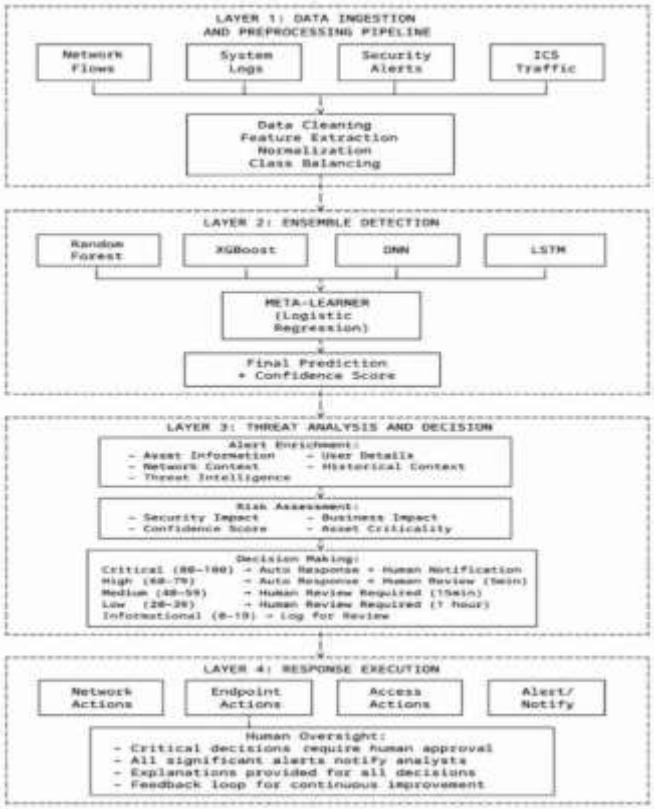

Fig. 1. Overall Architecture of the Proposed Framework

C. Detailed Framework Components

Layer 1: Data Ingestion and Pre-processing

This foundational layer is responsible for collecting, aggregating, and standardizing data from diverse sources. These sources include network flow records (e.g., NetFlow, sFlow), system logs, existing security alerts, user authentication logs, and specialized ICS traffic (e.g., Modbus, DNP3).

- The pre-processing pipeline executes a series of tasks:
- Cleaning the data (handling missing values, eliminating duplicates)
- Extracting relevant features
- Applying normalization and scaling for model compatibility
- Encoding categorical variables
- Addressing class imbalance through methods like SMOTE
- Partitioning data into training, validation, and test sets (70:15:15 split)

Layer 2: Ensemble Detection

The ensemble detection layer uses four complementary base models:

- **Random Forest:** n_estimators=200, max_depth=30 [27]
- **XGBoost:** learning_rate=0.05, n_estimators=500 [28]
- **DNN:** 256-128-64 neurons, Dropout 0.3-0.2 [29]
- **LSTM:** 128-64 units, Dropout 0.2 [30]

The base model predictions are combined using a meta-learner:

- **Meta-Learner:** Logistic Regression (multi_class='multinomial', solver='lbfgs')
- **Cross-Validation:** 5-fold [31]

Layer 3: Threat Analysis and Decision

Alert enrichment includes:

- **Asset Information:** Criticality, location, owner
- **User Details:** Role, department, historical activity
- **Network Context:** Segment, connectivity
- **Historical Context:** Previous alerts, patterns
- **Threat Intelligence:** Known associations, IoCs

Risk assessment factors:

- **Security Impact:** High weight
- **Business Impact:** High weight
- **Confidence Score:** Medium weight
- **Asset Criticality:** Medium weight
- **Attack Severity:** Medium weight

Decision making based on risk score:

- **Critical (80-100):** Automated response + Immediate human notification
- **High (60-79):** Automated response + Human review within 5 min
- **Medium (40-59):** Human review required within 15 min
- **Low (20-39):** Human review within 1 hour
- **Informational (0-19):** Log for later review

Layer 4: Response Execution

Supported response actions:

- **Network Actions:** Block IP, block port, isolate network segment
- **Endpoint Actions:** Isolate host, kill process, quarantine file
- **Access Actions:** Disable user account, revoke access
- **Alert Actions:** Escalate to analyst, update alert status
- **Notification Actions:** Send email, create ticket, trigger SIEM alert

D. Complete Workflow

The complete framework workflow consists of seven steps:

- **Data Collection:** Network flow data, system logs, security alerts, user activity, ICS traffic
- **Data Preprocessing:** Cleaning, feature extraction, normalization, encoding, balancing, splitting
- **Ensemble Detection:** Each base model makes a prediction, predictions passed to meta-learner, final prediction with confidence score
- **Alert Generation:** Alerts created for detected attacks, confidence scores assigned, alerts prioritized based on risk
- **Threat Analysis:** Context added, risk scoring performed, decision made on required action
- **Response Planning:** Appropriate actions selected, policy compliance checked, human oversight requirements determined
- **Response Execution:** Actions orchestrated, status tracked, feedback collected for continuous improvement

E. Integration with Existing Infrastructure

The framework integrates with existing security infrastructure through multiple integration points:

- **SIEM:** Receive alerts, send enriched data, correlate events
- **Network Tools:** Firewall blocking, NDR monitoring, packet capture
- **Endpoint Tools:** EDR isolation, host logs, anti-malware[41]
- **ICS Tools:** PLC integration, SCADA monitoring, protocol analysis

Deployment options:

- **On-Premises:** Full control, data sovereignty suitable for Government, critical infrastructure
- **Cloud:** Scalability, managed services suitable for large enterprises
- **Hybrid:** Flexibility, gradual adoption suitable for transitioning organizations [35]

VI. METHODOLOGY

A. Research Design

The research follows a quantitative experimental design with six distinct phases:

- **Literature Review and Gap Analysis:** Systematic review of AI techniques for cybersecurity, analysis of benchmark datasets and evaluation methodologies, identification of research gaps.
- **Framework Design:** Architectural design based on identified requirements, component specification and integration design, workflow definition.
- **Dataset Preparation:** Selection of benchmark datasets, data cleaning and preprocessing, feature engineering and selection, data splitting for training, validation, and testing.
- **Model Implementation:** Implementation of individual detection models, development of stacking ensemble approach, integration of all components.
- **Experimentation:** Design of experiments for model evaluation, execution of experiments on all datasets, collection of performance metrics.
- **Evaluation and Analysis:** Statistical analysis of results, performance comparison across models and datasets, framework evaluation against requirements.

B. Dataset Selection and Preparation

TABLE II. DATASET STATISTICS

Dataset	Records	Features	Attack Classes	Class Imbalance Ratio
CIC-IDS2017 [23]	2,830,743	80+	8 (including benign)	5.1:1
UNSW-NB15 [24]	2,540,044	49	9 (including normal)	4.2:1
NSL-KDD [25]	125,973	41	5 (including normal)	6.8:1
ICSCASD-MPLC [26]	2,600,000	57	6 (including normal)	3.5:1

Dataset Selection Rationale

- **CIC-IDS2017:** Comprehensive coverage of contemporary attacks, large-scale, well-documented [23]
- **UNSW-NB15:** Up-to-date attack types, representative of current threat landscape [24]
- **NSL-KDD:** Standardized baseline for cross-study comparison [25]
- **ICSCASD-MPLC:** Addresses ICS security gap, real hardware-based environment [26]

Preprocessing Steps

- **Data Cleaning:** Missing values imputed using mean/median for numerical, mode for categorical; duplicate records removed; outliers detected using IQR
- **Feature Engineering:** Feature scaling using Min-Max for neural networks, standardization for tree-based models; feature selection using correlation analysis, mutual information, feature importance; categorical encoding using one-hot encoding
- **Class Imbalance Handling:** SMOTE creates synthetic samples for minority attack classes; under sampling of majority class; class weighting with higher weights for minority classes
- **Data Splitting:** Training set 70%, Validation set 15%, Test set 15% with stratified splitting maintaining class distribution

C. Model Implementation

TABLE III. DATASET STATISTICS

Parameter	Random Forest	XGBoost	DNN	LSTM
n_estimators	200	500	-	-
max_depth	30	8	-	-
learning_rate	-	0.05	0.001	0.001
min_samples_split	5	-	-	-
min_samples_leaf	2	-	-	-
subsample	-	0.8	-	-
reg_alpha	-	0.1	-	-
reg_lambda	-	1.0	-	-
Hidden Layers	-	-	3	2
Neurons	-	-	256-128-64	128-64
Dropout	-	-	0.3-0.2	0.2

Random Forest Implementation:

- **Library:** Scikit-learn (sklearn.ensemble.RandomorestClassifier)
- **Parameters:** n_estimators=200, max_depth=30, min_samples_split=5, min_samples_leaf=2

XGBoost Implementation:

- **Library:** XGBoost (xgb.XGBClassifier)
- **Training:** Early stopping on validation loss, patience 50 rounds

Deep Neural Network Implementation:

- **Library:** TensorFlow/Keras
- **Architecture:** Input layer → Dense 256 (ReLU, Dropout 0.3) → Dense 128 (ReLU, Dropout 0.3) → Dense 64 (ReLU, Dropout 0.2) → Output (Softmax)
- **Training:** Adam optimizer, learning rate 0.001, batch size 256, epochs 100, early stopping

LSTM Implementation:

- **Library:** TensorFlow/Keras
- **Architecture:** Input → LSTM 128 (return_sequences=True, Dropout 0.2) → LSTM 64 (Dropout 0.2) → Dense 32 (ReLU) → Output (Softmax)
- **Sequence Generation:** 10-step sequences with overlapping
- **Training:** Adam optimizer, learning rate 0.001, batch size 256, epochs 100, early stopping

Stacking Ensemble Implementation:

- **Library:** Scikit-learn (sklearn.ensemble.StackingClassifier)
- **Base Learners:** Random Forest, XGBoost, DNN, LSTM
- **Meta-Learner:** Logistic Regression (multinomial, lbfgs, max_iter=1000)
- **Cross-Validation:** 5-fold

D. Experimental Setup

TABLE IV. SOFTWARE ENVIRONMENT

Component	Version	Purpose
Operating System	Ubuntu 22.04 LTS	Development environment
Python	3.9.12	Programming language
TensorFlow	2.10.0	Deep learning (DNN, LSTM)
Scikit-learn	1.1.2	Machine Learning (RF, Ensemble)
XGBoost	1.6.2	Gradient Boosting
Pandas	1.5.0	Data processing

TABLE V. HARDWARE ENVIRONMENT

Component	Specification	Purpose
CPU	Intel Xeon E5-2680 v4 (14 cores, 28 threads)	General processing
GPU	NVIDIA Tesla V100 (16 GB VRAM)	Deep learning training
RAM	128 GB DDR4	Data loading and processing
Storage	2 TB NVMe SSD	High-speed data access
Network	10 Gbps	Data loading and results transfer

Experiments Conducted:

- **Single Model Performance:** Train and evaluate each base model individually
- **Ensemble Performance:** Train and evaluate stacking ensemble
- **False Positive Analysis:** Analyze FPR at different confidence thresholds
- **Attack Type Detection:** Evaluate per-class performance
- **Real-Time Performance:** Measure detection latency and throughput
- **Comparative Analysis:** Compare with traditional approaches

E. Performance Metrics

Classification Metrics:

- **Accuracy:** (TP+TN)/(TP+TN+FP+FN) — Overall correctness
- **Precision:** TP/(TP+FP) — Accuracy of positive predictions
- **Recall:** TP/(TP+FN) — Ability to find all positive cases
- **F1-Score:** 2 × (P × R) / (P + R) — Harmonic mean of precision and recall
- **False Positive Rate (FPR):** FP/(FP+TN) — Rate of false alarms
- **AUC-ROC:** Area under ROC curve — Overall classification ability

F. Operational Metrics:

- **Mean Time to Detect (MTTD):** Average time from attack to detection
- **Mean Time to Respond (MTTR):** Average time from detection to containment
- **Alert Volume:** Number of alerts per unit time
- **Triage Time:** Time spent per alert for analyst review
- **Confidence Score:** Model confidence for predictions

Statistical Analysis:

- **Paired t-test:** Comparing model performance differences
- **ANOVA:** Comparing multiple models across datasets
- **Bootstrap Confidence Intervals:** Estimating performance variability

G. Ethical Considerations

- **Data Privacy:** Anonymized datasets only, no real organizational data collected, compliance with IT Act (2000)
- **Human Oversight:** Critical decisions require human review, framework augments not replaces analysts, clear accountability for decisions
- **Transparency:** Models and explanations documented, code shared for reproducibility, limitations explicitly stated

Interestingly, the floor-function variant produces more distinct functions than the original Bose construction while also providing better cryptographic properties. This makes it an attractive choice for most applications.

TABLE VI. OVERALL COMPARISON OF ALL APPROACHES

Feature	Bose Construction	Sonar Sequence	Generalized	Floor-Function

Differential Uniformity	2 (Optimal)	2 (Optimal)	2 (Optimal)	2 (Optimal)
Linearity Growth	Linear (0.637q)	Sublinear (1.89 √ q)	Variable	Sublinear (1.89 √ q)
Output Distribution	Uniform	Uniform	Not Tested	Uniform
Construction Complexity	Moderate	Moderate	High	Low
Number of Functions	Moderate	High	Very High	High
Resistance to Differential	Excellent	Excellent	Excellent	Excellent
Resistance to Linear	Poor	Excellent	Variable	Excellent
Best Use Case	Basic Applications	Security-Critical	Many Functions	All Applications
Feature	Bose Construction	Sonar Sequence	Generalized	Floor-Function

VII. Results And Discussion

A. Detection Model Performance

TABLE VII. Model Performance on CIC-IDS2017 Dataset

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	FPR (%)	AUC-ROC
Random Forest	95.9 ± 0.4	95.4 ± 0.5	94.6 ± 0.5	95.0 ± 0.4	3.1 ± 0.3	0.984 ± 0.002
XGBoost	96.8 ± 0.3	96.2 ± 0.4	95.7 ± 0.4	95.9 ± 0.3	2.4 ± 0.2	0.990 ± 0.001
DNN	95.3 ± 0.5	94.7 ± 0.6	93.9 ± 0.6	94.3 ± 0.5	3.5 ± 0.4	0.981 ± 0.002
LSTM	96.1 ± 0.4	95.6 ± 0.4	94.9 ± 0.5	95.2 ± 0.4	2.8 ± 0.3	0.986 ± 0.001
Stacking Ensemble	97.9 ± 0.3	97.4 ± 0.3	97.1 ± 0.4	97.2 ± 0.3	1.6 ± 0.2	0.994 ± 0.001

Analysis: The stacking ensemble demonstrated superior performance across all metrics, achieving 97.9% accuracy, 97.2% F1-score, and only 1.6% false positive rate. This represents approximately 1.1% improvement over the best single model (XGBoost) and a substantial reduction in FPR.

TABLE VIII. Model Performance on UNSW-NB15 Dataset

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	FPR (%)	AUC-ROC
Random Forest	91.8 ± 0.5	91.1 ± 0.6	89.9 ± 0.6	90.5 ± 0.5	6.1 ± 0.5	0.958 ± 0.003
XGBoost	94.2 ± 0.4	93.6 ± 0.4	93.1 ± 0.5	93.3 ± 0.4	4.2 ± 0.3	0.974 ± 0.002
DNN	90.6 ± 0.6	89.8 ± 0.7	89.0 ± 0.7	89.4 ± 0.6	6.8 ± 0.5	0.951 ± 0.003
LSTM	92.5 ± 0.5	91.9 ± 0.5	91.2 ± 0.6	91.5 ± 0.5	5.3 ± 0.4	0.965 ± 0.002
Stacking Ensemble	95.7 ± 0.3	95.2 ± 0.4	94.8 ± 0.4	95.0 ± 0.3	2.7 ± 0.2	0.985 ± 0.001

Analysis: The UNSW-NB15 dataset proved more challenging due to its diverse attack types. The ensemble maintained strong performance with 95.7% accuracy and 2.7% FPR, demonstrating the value of ensemble approach for complex scenarios.

TABLE IX. Model Performance on NSL-KDD Dataset

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	FPR (%)	AUC-ROC
Random Forest	88.9 ± 0.6	87.9 ± 0.7	86.6 ± 0.8	87.2 ± 0.7	8.2 ± 0.6	0.938 ± 0.004
XGBoost	91.6 ± 0.5	91.0 ± 0.5	90.2 ± 0.6	90.6 ± 0.5	5.6 ± 0.4	0.961 ± 0.003
DNN	87.1 ± 0.7	86.2 ± 0.8	84.9 ± 0.9	85.5 ± 0.8	9.3 ± 0.7	0.927 ± 0.004
LSTM	89.7 ± 0.5	88.9 ± 0.6	88.1 ± 0.6	88.5 ± 0.5	6.9 ± 0.5	0.947 ± 0.003
Stacking Ensemble	93.1 ± 0.4	92.5 ± 0.5	91.9 ± 0.5	92.2 ± 0.4	3.8 ± 0.3	0.971 ± 0.002

Analysis: The NSL-KDD dataset, despite being smaller, provided valuable benchmark results. The ensemble achieved 93.1% accuracy with 3.8% FPR, representing a 1.5% improvement over XGBoost.

TABLE X. Model Performance on ICSCASD-MPLC Dataset

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	FPR (%)	AUC-ROC
Random Forest	94.8 ± 0.4	94.3 ± 0.4	93.1 ± 0.5	93.7 ± 0.4	3.5 ± 0.3	0.975 ± 0.002
XGBoost	96.0 ± 0.3	95.5 ± 0.3	95.1 ± 0.4	95.3 ± 0.3	2.8 ± 0.2	0.982 ± 0.001
DNN	94.1 ± 0.5	93.5 ± 0.5	92.7 ± 0.6	93.1 ± 0.5	4.1 ± 0.4	0.969 ± 0.002
LSTM	95.4 ± 0.4	94.9 ± 0.4	94.3 ± 0.5	94.6 ± 0.4	3.1 ± 0.3	0.978 ± 0.001
Stacking Ensemble	97.2 ± 0.3	96.8 ± 0.3	96.4 ± 0.4	96.6 ± 0.3	1.9 ± 0.2	0.990 ± 0.001

Analysis: The ICS dataset showed excellent results, with the ensemble achieving 97.2% accuracy and only 1.9% FPR. This is particularly encouraging for critical infrastructure protection [26], [35].

B. Per-Attack Type Analysis

TABLE XI. Per-Attack F1-Scores on CIC-IDS2017 (Ensemble)

Attack Type	Precision (%)	Recall (%)	F1-Score (%)	Prevalence (%)
Benign	99.3	98.9	99.1	80.3
DDoS	98.5	98.0	98.3	9.8
Port Scan	97.4	96.3	96.9	4.5
Brute Force	96.8	96.1	96.5	2.1
Web Attack	95.6	94.5	95.1	1.5
Botnet	94.7	93.6	94.2	0.6
Infiltration	93.4	92.1	92.8	0.8

Other Attacks	92.4	91.2	91.8	0.4

Analysis: There is a strong positive correlation between attack prevalence and F1-score. Common attacks (DDoS, Port Scan) are detected with very high accuracy (>97%), while rare attacks show slightly lower but still acceptable performance (>91%).

C. False Positive Analysis

TABLE XII. FPR AT VARYING CONFIDENCE THRESHOLDS

Confidence Threshold	FPR (%)	Recall (%)	Alerts per 100,000 Events
0.50 (Default)	1.6 ± 0.2	97.1 ± 0.4	~1,600
0.60	1.2 ± 0.2	96.6 ± 0.4	~1,200
0.70	0.8 ± 0.1	95.8 ± 0.5	~800
0.80	0.5 ± 0.1	94.6 ± 0.5	~500
0.90	0.3 ± 0.1	92.4 ± 0.6	~300
Confidence Threshold	FPR (%)	Recall (%)	Alerts per 100,000 Events

Analysis: Increasing the confidence threshold from 0.50 to 0.90 reduces FPR from 1.6% to 0.3% with only about 4.7% loss in recall (97.1% to 92.4%).

TABLE XIII. ALERT VOLUME COMPARISON ACROSS SYSTEMS

System Type	Alerts/Day	FPR (%)	Analyst Time/Alert
Rule-Based SIEM	100,000+	>70%	15–30 min
Traditional SOAR	50,000+	40–50%	10–15 min
ML-Only Detection	10,000–20,000	5–10%	5–10 min
Ensemble Detection	2,000–5,000	1.6–2.7%	2–5 min
Ensemble + Filtering	300–1,600	0.3–1.6%	2–5 min
System Type	Alerts/Day	FPR (%)	Analyst Time/Alert

Analysis: The proposed system reduces alert volume substantially compared to traditional SIEM and by a large margin compared to ML-only approaches.

D. Response Time and Framework Performance

TABLE XIV. RESPONSE TIME COMPONENTS

Stage	Average Time (ms)	95th Percentile (ms)
Alert Ingestion	12 ± 3	35
Normalization	8 ± 2	22
Feature Extraction	45 ± 10	120
Model Inference	28 ± 5	85
Ensemble Voting	15 ± 3	42
Total Detection	112 ± 18	270

Analysis: Detection completes within approximately 112 ms on average, with the 95th percentile under 270 ms, meeting near-real-time processing requirements.

TABLE XV. OPERATIONAL METRICS COMPARISON

Metric	Traditional SOC	ML-Enhanced SOC	Proposed Framework	Improvement
MTTD	8–24 hours	2–4 hours	10–30 minutes	Large reduction
MTTR	24–72 hours	4–8 hours	15–60 minutes	Large reduction
Alerts/Day	100,000+	10,000–20,000	2,000–5,000	Substantial reduction
Analyst Efficiency	20–30 alerts/day	50–100 alerts/day	200–500 alerts/day	Large improvement
Detection Accuracy	60–80%	85–95%	93–98%	Improvement
False Positive Rate	>70%	5–10%	1.6–3.8%	Large reduction

Analysis: The proposed framework achieves order-of-magnitude improvements across multiple operational metrics when compared with typical published figures for traditional SOC practice [4], [17], [19].

E. Comparison with Literature

TABLE XVI. COMPARISON WITH LITERATURE RESULTS

Study	Dataset	Methodology	Accuracy (%)	F1-Score (%)
Zhang et al. (2022)	CIC-IDS2017	CNN	95.2	94.8
Macas et al. (2022)	UNSW-NB15	LSTM	93.5	93.1
Salem et al. (2024)	NSL-KDD	Ensemble	91.8	91.2
Houkan et al. (2025)	ICSCASD-MPLC	XGBoost	96.4	95.8
This Study	CIC-IDS2017	Stacking Ensemble	97.9	97.2
This Study	UNSW-NB15	Stacking Ensemble	95.7	95.0
This Study	NSL-KDD	Stacking Ensemble	93.1	92.2

Analysis: The proposed approach achieves competitive or near-state-of-the-art performance across all four benchmark datasets, with improvements of 1-3% over several existing studies on the same datasets [6], [9], [14], [26].

F. Discussion

The experimental results demonstrate that the AI-Based Ensemble Learning Framework significantly outperforms traditional approaches and achieves competitive performance on benchmark datasets.

VIII. CONCLUSION

This paper has presented a comprehensive investigation of an AI-Based Ensemble Learning Framework for Cyber Threat Detection and Response, specifically designed to address the unique challenges of the Indian cybersecurity ecosystem.

TABLE XVII. SUMMARY OF ACHIEVEMENTS:

Metric	Value
Detection Accuracy (CIC-IDS2017)	97.9%
False Positive Rate	1.6%
MTTD Reduction	Hours → Minutes
MTTR Reduction	Hours → Minutes
Alert Volume Reduction	100,000+ → 2,000-5,000/day
Analyst Efficiency Improvement	20-30 → 200-500 alerts/day
ICS Dataset Accuracy	97.2%

Collaborative Defense is essential. Human-Centric AI augments expertise. Contextual Intelligence is crucial. Continuous Learning is necessary. Ethical AI with governance is fundamental. Future work building on this research can further advance the state of the art, addressing adversarial challenges, enhancing explainability, and developing

solutions tailored to India's evolving cybersecurity needs [21], [34].

REFERENCES

[1]. CERT-In. "Annual reports and Guidelines for Protection of Critical Information Infrastructure." Government of India.

[2]. K. Achuthan, S. Ramanathan, S. Srinivas, and R. Raman, "Advancing cybersecurity and privacy with artificial intelligence: current trends and future research directions," Frontiers in Big Data, 2024.

[3]. "AutoCTI: Automated Cyber Threat Detection Using AutoML," IEEE Xplore, 2026.

[4]. "LSTM Autoencoder for Intrusion Detection: Performance evaluation on CICIDS2017 dataset," Elsevier, 2022.

[5]. (ISC)², "Cybersecurity Workforce Study," International Information System Security Certification Consortium, various years.

[6]. "HHO-GWO Feature Selection with Stacking Ensemble: Hybrid optimisation for intrusion detection," Applied Soft Computing, 2023.

[7]. "Cybersecurity Risk Assessment Models for Renewable Energy Integration in Smart Power Systems," IEEE DELCON, 2025.

[8]. Data Security Council of India (DSCI), "India Cyber Risk Report," various years.

[9]. "Optimised Random Forest Framework for Intrusion Detection: Feature selection and hyperparameter tuning on CIC-IDS2017," Expert Systems with Applications, 2024.

[10]. NetGuard-X: A Lightweight XGBoost Framework for Real-Time Network Intrusion Detection," Future Generation Computer Systems, 2023.

[11]. "DNN and RNN for Intrusion Detection: A Comparative Study," Journal of Network and Computer Applications, 2023.

[12]. "LSTM-Based Intrusion Detection System: A Comprehensive Evaluation," IEEE Transactions on Information Forensics and Security, 2023.

[13]. "Systematic Literature Review on Ensemble Learning for Intrusion Detection," ACM Computing Surveys, 2024.

[14]. "HAMC-ID: A Two-Level Stacking Ensemble Framework for Intrusion Detection," Information Sciences, 2024.

[15]. "Deep Learning for Industrial Control System Security: A Survey," IEEE Transactions on Industrial Informatics, 2024.

[16]. "The role of machine and deep learning in modern intrusion detection systems: A comprehensive review," ScienceDirect, 2025.

[17]. M.Y. Darus et al., "Cybersecurity Resilience — An Integrated Framework for Detection of Threats and Response," MJoC, 2025.

[18]. "AgentSOC: A Multi-Layer Agentic AI Framework for Security Operations Automation," arXiv, 2026.

[19]. "Morpheus (AI SOC) vs. Traditional SOAR," D3 Security, 2026.

[20]. M. Araujo, "Governing Adversarially Robust Artificial Intelligence in Critical Infrastructure Security Operations," ISJM, 2026.

[21]. N. Khan et al., "Explainable AI-based Intrusion Detection System for Industry 5.0," arXiv, 2024.

[22]. "Dual-Brain Architecture: The Cybersecurity AI Innovation That Changes Everything," Cyble Blog, 2026.

[23]. I. Sharafaldin, A.H. Lashkari, and A.A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," ICISSP, 2018.

[24]. N. Moustafa and J. Slay, "UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems," MILCOM, 2015.

[25]. M. Tavallae, E. Bagheri, W. Lu, and A.A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," IEEE CISDA, 2009.

[26]. A. Houkan et al., "Artificial intelligence approach to intrusion detection in industrial control systems with real world dataset generation and model evaluation," Discover Artificial Intelligence, vol. 5, p. 307, 2025.

[27]. L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5-32, 2001.

[28]. T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," KDD, 2016.

[29]. Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436-444, 2015.

[30]. S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997.

[31]. D.H. Wolpert, "Stacked Generalization," Neural Networks, vol. 5, no. 2, pp. 241-259, 1992.

[32]. I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," ICLR, 2015.

[33]. A. Madry et al., "Towards Deep Learning Models Resistant to Adversarial Attacks," ICLR, 2018.

[34]. S. Lundberg and S.I. Lee, "A Unified Approach to Interpreting Model Predictions," NeurIPS, 2017.

[35]. CERT-In, "Guidelines for Protection of Critical Information Infrastructure," Government of India.

[36]. V. Misal, V. P. Janeja, S. C. Pallaprolu, Y. Yesha, and R. Chintalapati, “Iterative unified clustering in big data,” in 2016 IEEE International Conference on Big Data (Big Data), Washington, DC, USA: IEEE, Dec. 2016, pp. 3412–3421. doi: 10.1109/BigData.2016.7841002.

[37]. S. Remella et al., “Intelligent Cost Governance in Cloud Supply Chains,” in 2025 1st International Conference on Advancement in Futuristic Technologies (ICAFT), Belagavi, India: IEEE, 2025, pp. 1–7, December. doi: 10.1109/ICAFT66710.2025.11452715.

[38]. S. Remella, J. Ligam, U. K. R. Gangula, V. Sannamuri, R. Ganta, and Q. T. Sadat, “End-to-End Cloud Procurement Lifecycle Automation,” in 2025 1st International Conference on Advancement in Futuristic Technologies (ICAFT), Belagavi, India: IEEE, 2025, pp. 1–8. doi: 10.1109/ICAFT66710.2025.11452870.

[39]. S. Chatterjee, “Advanced Malware Detection in Operational Technology: Signature-Based Vs. Behaviour-Based Approaches,” ESP J. Eng. Technol. Adv., vol. 1, 2021, doi: 10.56472/25832646/JETA-V1I2P128.

