International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(10s) 2026 pp. 1122-1135
https://doi.org/10.70917/ijcisim-2026-3702
Article

Machine Learning Enhanced Cybersecurity Framework for Protecting Health Care Data

J Sharon Christina[1], Varalatchoumy M*[2]

[1]Dept of Computer Science and Engineering, sharon.cse@cambridge.edu.in, Cambridge Institute of Technology, Bangalore
[2]Dept of Artificial Intelligence and Machine Learning, hod.aiml@cambridge.edu.in, Cambridge Institute of Technology, Bangalore

Abstract: The study advances hospital network security by its hybrid machine learning system that identifies cyber-attacks. The system applies K-best feature selection and XGBoost and LSTM and Stacking Classifier to attain high detection accuracy and effective intrusion detection. The system employs Explainable AI approaches (SHAP, LIME) for transparent decision-making and TPOT for hyperparameter optimization. The system identifies various types of threats that include ARP Spoofing and MQTT-based DDoS/DoS attacks and Reconnaissance and TCP/IP attacks. The system offers better accuracy and real-time detection capabilities along with increased interpretability that guards sensitive patient data effectively.
Keywords: Patient Data Security, IIoMT, XGBoost, LSTM, Stacking Classifier, Explainable AI, TPOT, Network Intrusion Detection

1. Introduction

Today, more than ever, healthcare systems are pulse-pounding vulnerable to cyber-attacks on patient data, network integrity, and patient-care procedures! With the growing reliance on digital infrastructure, numerous types of attacks such as DDoS, DoS, ARP Spoofing and Reconnaissance have been on the rise. Traditional intrusion detection is quite gross and may not be able to deal with continually changing threats, which forms a motivation towards an efficient real-time intrusion detection system. [1-3]

The aim of this research is to create a new machine learning based pipeline specifically tailored for precise and understandable detection of simultaneous network attacks in the hospital setup. The pipeline integrates the use of K-best feature selection, XGBoost, LSTM, and Stacking Classifiers and is supported by TPOT grid search for automated hyperparameter tuning and XAI (SHAP, LIME) for explainability. [5, 7, 15]

The main aims are: Identification and Classification of different attacks like ARP Spoofing,1 DDoS/DoS based on MQTT,2 Reconnaissance, and TCP/IP fires based attacks. [1-3], Enable network performance protection, retaining network performance during real-time detection. Make information interpretable to administrators by using Explainable AI. [12, 14, 19, 23]

The system utilizes datasets which include the samples of attack types such as MQTT-DDoS-Connect Flood, Recon-OS Scan, TCP-IP DDoS-SYN, and Benign traffic. The objective of this study is the establishment of new ML models, aided by explainability and automation to improve the security of the hospital networks and to provide the patient's sensitive data protection. [7, 15]

2. Related Works

Healthcare networks are limited in their conventional security mechanisms since firewalls and signature-based IDS/IPS technologies rely on pre-configured signatures and rules to respond to zero-day and changing attacks. The systems generate a high number of false positives while being rigid in nature, hence incompatible with sophisticated hospital environments. Machine learning (ML)-based intrusion detection has proved to be a viable substitute. The work indicates that feature selection methods enhance the detection in performance and efficiency in the security

protocols [1-3]. The Integration of XGBoost with oversampling and weighted classifiers which is tuned for performance has been found robust against the various attack vectors [7, 15]. Employing the ensemble techniques by using feature sets optimized is seen to bring high detection performance [9]. [5, 7, 15]

The Deep learning models like LSTM and CNN are used to simulate complicated sequential traffic patterns. Nevertheless, these models are typically not interpretable, but an aspect which is important in medical contexts. The current research has also explored various Explainable AI (XAI) techniques like SHAP and LIME to overcome this limitation [12]. [12, 14, 19, 23]

Despite this advancement, most of the current frameworks give importance on mere binary classification, feature minimization, or at times it does not have real-time and stable MYR for detection. This invokes the necessity of a dynamic and interpretable multi-class ML framework which is specific to hospital networks to facilitate proper cybersecurity practices and reliable decision-making.

3. Proposed Methodology

The adoption of digital healthcare systems has led to increased cybersecurity threats to hospital networks. Threats such as data breaches, ransomware attacks, spoofing, and reconnaissance compromise patient data. This paper proposes MedSecureNet, a lightweight web application using Flask that incorporates machine learning models for real-time cyber threat detection.

The proposed system provides user authentication, secure data visualization, model selection, and threat prediction based on features extracted from network traffic. In contrast to research that only focuses on backend models, MedSecureNet provides an end-to-end solution with a user-friendly interface for hospital IT and cybersecurity professionals.

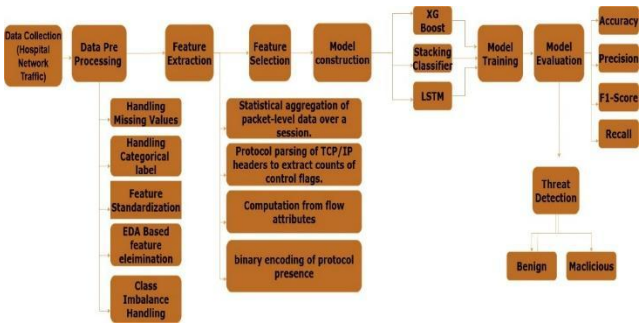

Fig. 1. MedSecureNet System Architecture

(Fig 1) The flow of the MedSecureNet system for the detection of cyber threats in hospital networks is shown in the diagram. The process starts with the collection of data from network traffic, followed by preprocessing, feature extraction, and feature selection. A stacked model consisting of XGBoost and LSTM is developed and trained for threat classification. The system also uses metrics such as accuracy, precision, recall, and F1-score for evaluation, followed by the classification of traffic as malicious or benign.

a. **Data Collection & Preprocessing:** The hospital network traffic is collected, cleaned, and pre-processed. It encompasses handling missing values, encoding categorical variables, and addressing of class imbalance.
b. **Feature Engineering:** The attributes of interest are selected, and the K-Best and PCA feature techniques are used to reduce the dimensionality.
c. Model Building: The three classifiers, namely XGBoost, Stacking Classifier, and LSTM, are constructed and trained on the selected features. [5, 7, 15]
d. **Evaluation & Deployment:** The models are evaluated in terms of accuracy, precision, recall, and F1-score. The top-performing models are combined into the Flask application to make real-time traffic classification as benign or malicious.

This approach guarantees an interpretable, scalable, and real-time detection framework which will boost the hospital network's resistance to the emerging cyber threats.

3.1 Data Collection

The Final_data dataset represents the historical hospital network traffic data collected from various communication sessions. Every entry in the dataset is a representation of a network flow with statistical features based on protocol and segment levels, considering both benign and malicious activities like ARP spoofing, DoS, DDoS (MQTT/TCP/IP), and reconnaissance attacks.

The dataset in Table 1 includes flow-level statistical features like protocol type, transfer rate, TCP flags, inter-arrival times, and their corresponding measures (min, max, mean, and standard deviation). Malicious network flows have unique patterns like high transfer rates, unusual flag patterns, scan traffic patterns, and connection flood patterns. However, the complete dataset has more than 80 features, and this research study considers a representative subset of the most discriminative features for efficient intrusion classification.

Table 1. Sample records from the labelled dataset showing diverse attack type

Header Len	Protocol	Duration	Rate	State	Label
38816	7	1	4.15	1.58	Benign_test
54	6	64	4.08	4.08	Recon-OS_Scan_test
343.75	6	64	12.58	12.58	MQTT-DoS-Connect_Flood_test
323	6	64	8252.82	8252.82	ARP_Spoofing_test
0	1	64	519.98	519.98	TCP_IP-DoS-ICMP_test

3.2 Data Preprocessing and Inspection

The dataset contains 33,273 samples with 46 attributes and a categorical label that indicates various classes of traffic. Preprocessing was necessary to make the dataset ML model friendly.

Initial, missing value analysis was done utilizing isnull().sum() in Python's Pandas library. Output verified that no attributes had null values, making imputation unnecessary. Second, column data types were checked with the dtypes method and it showed that all 45 predictor features were numeric (float64/int64), whereas the target variable (label) was categorical.

Categorical labels were then converted into numerical class identifiers using label encoding to be compatible with ML algorithms as shown in Table 2. Because the dataset had clean and uniform structure, the preprocessing pipeline was centered on scaling, normalization, and feature selection to facilitate proper model training and classification.

Table 2 : Dataset Features and Their Data Types

Feature Category	Example Features (Subset)	Data Type
Header & Protocol Info	Header_Length, Protocol_Type, Duration	float64
Rate Features	Rate, Srate, Drate	float64 / int64
TCP Flag Features	fin_flag_number, syn_flag_number, rst_flag_number, psh_flag_number,	float64
Count Features	ack_count, syn_count, fin_count, rst_count	float64
Protocol Indicators	HTTP, HTTPS, DNS, Telnet, SMTP, SSH, IRC, TCP	float64
Statistical Features	Variance, Weight	float64
Target Label	label	float64

3.3 Handling Categorical Labels

For supervised classification problems, machine learning algorithms need input features and output labels in numerical format. Although input features in the dataset were numeric, the target variable had categorical string labels (e.g., "ARP_Spoofing_test", "TCP_IP-DoS-TCP_test"). To convert them into something that classifiers can work with, two preprocessing operations were conducted: [7, 15]

a. **Label Cleaning** – Most class labels had the unnecessary suffix _test, as evident from Table 3 which served no purpose in class differentiation. Employing string manipulation (df['label'] = df['label'].str.replace('_test','',regex=False)), this suffix was eliminated, yielding standardized labels like "ARP_Spoofing" and "Benign". This enhanced interpretability and visualization lucidity.

b. **Label Encoding –** The preprocessed labels were subsequently translated to integer identifiers with LabelEncoder from scikit-learn. Every distinct class received a separate integer (e.g., "ARP_Spoofing" → 0, "Benign" → 1). Frequency distribution of the encoded labels showed significant class imbalance, and this is a significant factor to consider in model performance.

Overall, the dataset had 21 distinct attack classes after preprocessing. (Fig 2) shows the sample label cleaning and encoding processes.

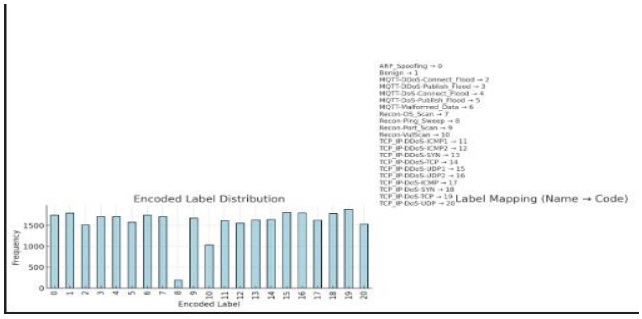

Fig 2: Label preprocessing workflow 2

3.4 Feature Standardization

As ML models are sensitive to feature scale, all numerical features were standardized via StandardScaler. The conversion enhanced every feature to have a zero mean and unit variance therefore features with higher magnitudes will not skew the model training. The feature standardization helps the distance-based algorithms (e.g., SVMs) and gradient-based models (e.g., neural networks) by improving the training stability and convergence.

3.5 Exploratory Data Analysis (EDA)

To gain better insight into the dataset, histograms were produced for all numeric features. This examination established feature variability, skewness, and structural anomaly likely to influence model performance. For instance, some attack traffic had very skewed rate distributions and anomalous TCP flag patterns, whereas benign traffic showed consistent statistical characteristics. These findings verified the discriminatory capabilities of chosen features and informed further feature selection and modelling procedures.

3.6 Protocol and Flag-Based Feature Distributions

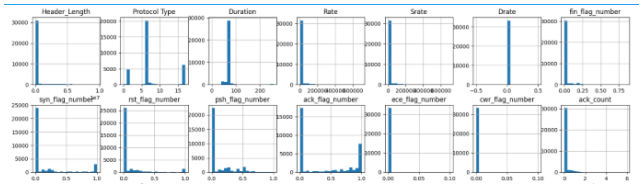

Fig 3: Protocol and Flag-Based Feature Distributions

The protocol and flag-based feature distributions are shown in (Fig 3).

- Protocol indicators like HTTP, HTTPS, DNS, Telnet, SMTP, SSH, TCP, UDP have binary distributions (0/1), which indicates presence or absence.
- TCP flag counters like syn_count, fin_count, rst_count are strongly right-skewed, the sparse flows indicates a high count, which is typically associated with anomalies e.g., SYN flooding.
- Skewness and sparsity indicate anomaly signals live in the long-tail, encouraging transformations such as log-scaling or quantile binning.

(Fig 4) and (Fig 5) present the distribution of protocol-level and statistical features extracted from hospital network traffic. The histograms show that many protocol indicators (e.g., SYN, FIN, RST, HTTP, TCP, UDP) are highly skewed, reflecting distinct behavioural differences between normal and attack flows. Statistical and derived attributes such as total size, inter-arrival time (IAT), variance, magnitude, and weight further demonstrate variability and dispersion patterns, highlighting their significance in distinguishing benign traffic from malicious activities.

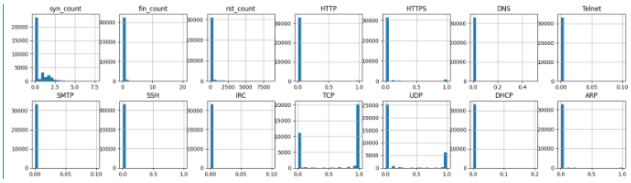

Fig 4: Histograms of high-level statistical features and traffic metrics

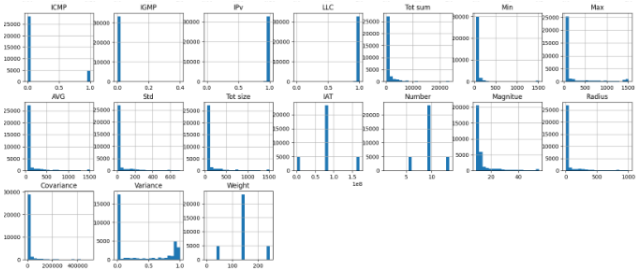

Fig 5: Statistical and derived numerical attributes

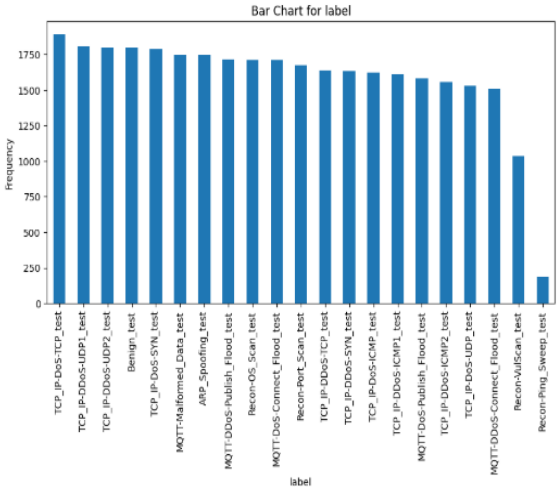

Fig 6: Categorical Feature Visualization

(Fig 6) shows bar charts representing the frequency distribution of categorical variables like protocol type, service, and TCP flags. The dominance of certain categories indicates the existence of class imbalance in the categorical variables.

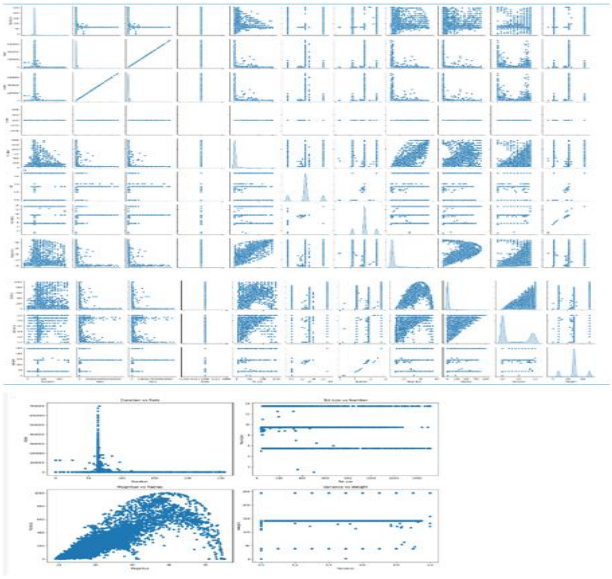

Fig 7: Scatter plots and a correlation matrix heatmap

(Fig 7) shows scatter plots and a correlation matrix heatmap for the numerical variables. The scatter plots and correlation matrix heatmap help identify linear and non-linear correlations between variables, which can be used for dimensionality reduction and feature engineering to improve the training of models. To overcome the imbalance problems, class balancing methods were used during preprocessing.

Table 4 shows the raw distribution of 21 traffic classes from as few as 186 samples (Recon-Ping_Sweep) to as many as approximately 1900 samples (TCP_IP-DoS-TCP). Such extreme imbalance threatens to skew classifiers towards majority classes at the expense of sensitivity to rare but important attacks. [7, 15]

Table 4: Sample of Original distribution from 21 traffic classes

Class Label	Balanced Sample Count
TCP_IP-DoS-TCP	1888
TCP_IP-DDoS-UDP1	1888
TCP_IP-DDoS-UDP2	1888
Benign	1888
TCP_IP-DoS-SYN	1888

This was mitigated by three preprocessing operations:

a. Label Cleaning – Eliminated duplicate suffixes (e.g., "_test"), providing consistent class names.

b. Label Encoding – Transformed categorical labels to numeric targets for ML algorithm compatibility.

c. SMOTE Oversampling – Used Applied Synthetic Minority Oversampling Technique to create synthetic samples for minority classes through nearest-neighbour interpolation.

Post-balancing, all the classes were covered by 1888 samples (Table 3), ensuring fair exposure in training. This was useful in improving recall and F1-scores for minority attacks like Recon-Ping_Sweep and Recon-VulScan, making the model stronger in cybersecurity scenarios where infrequent attacks can have lopsided influence.

3.7 Protocol and Flag-Based Feature Distributions

Feature extraction converted raw network streams into more than 40 engineered features, which were divided into four categories:

a. Statistical Aggregation Features (Tot_sum, Avg, Min, Max, Std, IAT, Tot_size): Aggregate traffic volume and volatility. For instance, abnormally high Tot_sum or Max values indicate data exfiltration or flooding, whereas unusual IAT patterns tend to imply botnet or malware-induced bursts.
b. Protocol-Level Flag Features (ACK, SYN, RST, PSH counts): Record TCP/IP session activity. High ratios of SYN without ACKs point to SYN flood attacks, whereas frequent RST indicates scanning or evasions.
c. Rate-Based Features (Rate, Srate, Drate, Header_Length): Measure data transmission speed and packet format. Elevated source rates point to flooding, whereas abnormal header lengths indicate spoofing or malformed packets.
d. Session Metadata & Derived Indicators (Protocol IDs, Magnitude, Radius, Covariance, Weight): Convey high-level semantic meaning. Large Magnitude/Radius, e.g., usually indicates DDoS or exfiltration, and Covariance indicates coordinated feature modifications typical of attack sessions. [7, 15]

By blending the low-level protocol signals with high-level statistical abstractions, the feature set offers both the concrete and abstract descriptions of traffic, which supports the discovery of varied cyber threats that ranges from volumetric floods to covert reconnaissance.

3.8 Feature Selection

Features selection decreases dimensionality, increases efficiency, and helps improve model generalization. Two complementary methods were utilized: K-Best statistical filtering and Principal Component Analysis (PCA).

i. **K-Best Feature Selection**

Fig. 8: Feature importance scores using SelectKBest

SelectKBest ranked features based on univariate scores in comparison to the target class, and the top 10 were kept. Insightful attributes were ICMP, TCP, Protocol_Type, UDP, syn_count, ack_flag_number, and Variance, and low-value features (e.g., Weight, IAT, SMTP) were discarded. Figure 13 depicts ICMP as the most discriminatory feature, highlighting its involvement in DDoS, ping sweeps, and probing attacks. Other highly ranked features (TCP, Protocol_Type, UDP) indicate the significance of network- and transport-layer protocols. Connection-level features (syn_count, ack_flag_number) and statistical features (Variance) also played a role. This was a lower feature count with enhanced computational efficiency and detection accuracy as shown in Fig 8. [7, 15]

ii. **Principal Component Analysis (PCA)**

PCA showed the dataset into orthogonal components, which mitigates the multicollinearity while preserving variance. This approach proved especially effective for ensemble and deep learning models, with the Stacking Classifier + PCA outperforming its K-Best variant in accuracy and F1-score. By retaining only high-variance components, PCA achieved dimensionality reduction without significant information loss. [13]

iii. **Feature Correlation Analysis**

Redundancy was found among some attributes using a Pearson correlation matrix (Fig 9). For instance, Rate–Tot_size ($r \approx 0.99$) and ack_flag_number–ack_count were strongly correlated. On the other hand, Protocol_Type, Weight, Magnitude, and Radius had low correlations, meaning they made independent contributions. Features with $r > 0.95$ were considered for removal or

transformation to inform dimensionality reduction decisions together with K-Best and PCA. **Strongly correlated feature pairs are highlighted in red (+1) and blue (−1), aiding in the detection of redundancy and guiding dimensionality reduction strategies.**

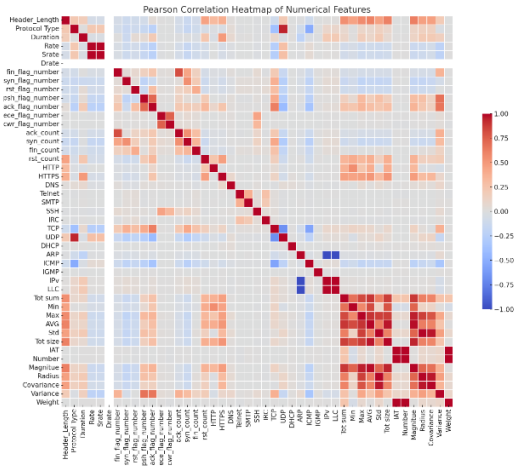

Fig. 9: Pearson correlation heatmap of numerical features

3.10 Model Construction and Training

To compare intrusion detection performance, several machine learning and deep learning models were trained on the K-Best as well as the PCA-transformed feature sets. This allowed to measure how the feature selection affects accuracy, generalization, and class-wise detection.

Extreme Gradient Boosting (XGBoost) [5, 7, 15]

XGBoost, which is a scalable gradient boosting algorithm with L1/L2 regularization, was chosen due to its robust performance on tabular, imbalanced data. TPOT-based AutoML tuning recognized the optimal setting, Learning rate = 0.1, Max depth = 6, Estimators = 150. [5, 7, 15]

By applying an 80:20 ratio, XGBoost had 97% training accuracy and 80.95% test accuracy using K-Best features. Confusion matrices (Figs. 4–5) reflected good performance for majority classes (e.g., TCP_IP-DoS-TCP, DoS_UDP_Flood, Benign), although minority classes like Recon-Ping_Sweep and Malicious_Telnet were classified less accurately. PCA-based training generalised better, especially for the underrepresented classes. [7, 15]

a. **Deployment in MedSecureNet** The models were trained and integrated in a Flask-based web application (MedSecureNet) for real-time intrusion forecasting. Traffic data can be uploaded, and a classifier (XGBoost, Stacking, LSTM) can be chosen. Predictions are converted from numerical indices to human-readable categories of attacks (Fig. 10) and presented through an interactive interface [5, 7, 15]

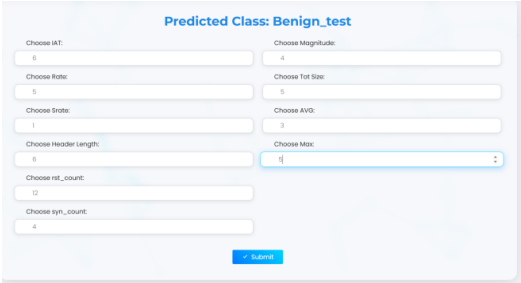

Fig 10: Web interface of prediction page

This architecture decouples backend model logic from the frontend interface, providing scalability, interpretability, and real-time response in hospital networks. Performance was also proved by classification reports (precision, recall, F1-score, accuracy) for both K-Best and PCA feature sets, verifying the proposed system's robustness.

b. **Classification Report Analysis**

i. **XGBoost with K-Best Features**

The XGBoost model with K-Best feature attained 68.4% training and 62.5% test accuracy. Dominant classes (e.g., 2, 4, 13, 18) hit perfect F1-scores (1.00), while mid-frequency classes attained 0.75–0.89 scores. Minority classes (9, 11, 12, 14, 16, 17) had poor recall (<0.10), although the models occasionally had high precision, indicating imbalance problems. Macro and weighted F1-scores (0.57, 0.56) both indicated good performance for majority classes but poor sensitivity to rare attacks as shown in Table 5. [5, 7, 15]

Table 5: Classification of XGBoost with K-Best Feature Set

Class	Precision	Recall	F1-Score	Support
0	0.93	0.86	0.89	1414
1	0.76	0.96	0.84	1435
2	1.00	1.00	1.00	1203
3	0.90	0.84	0.87	1325
4	1.00	1.00	1.00	1352
5	0.83	0.98	0.90	1264
6	0.92	0.94	0.93	1406
7	0.55	0.90	0.68	1372
8	0.52	0.82	0.64	104
9	0.95	1.00	0.95	1302
10	0.83	0.63	0.72	847

ii. **XGBoost with PCA**

Performance was enhanced to 94.48% training and 80.95% test accuracy using PCA-transformed features. Almost all classes had high F1-scores, the lowest being 0.76. Difficulty classes of the past (11, 12, 15, 16) showed great improvement (F1 = 0.76–0.95), while majority classes held close to perfect scores. By using the macro and weighted F1-scores (0.81 each), consistent performance in all classes was shown and the efficiency of PCA in addressing imbalance and enhancing generalization was supported as shown in Table 6.

Table 6: Classification PCA with K-Best Feature Set

Class	Precision	Recall	F1-Score	Support
0	0.99	1.00	0.99	1414
1	1.00	1.00	1.00	1435
2	1.00	1.00	1.00	1203
3	1.00	1.00	1.00	1325
4	1.00	1.00	1.00	1352
5	1.00	1.00	1.00	1264
6	0.91	0.93	0.92	1406

7	0.87	0.92	0.90	1372
8	1.00	1.00	1.00	104
9	0.91	0.88	0.89	1302
10	0.94	0.97	0.95	847

c. **Confusion Matrix Insights**

- **K-Best Feature:** Figures 11 Shows the strong detection for majority classes but very frequent misclassification of minority ones (e.g., Recon-Ping_Sweep).

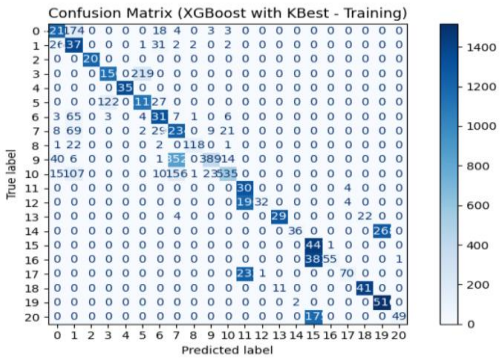

Fig 11: Confusion matrix of XGBoost on training data using K-Best features

- **PCA:** In Figure 12 it shows the equilibrated predictions with enhanced identification of infrequent attacks (e.g., ARP_Spoofing, MQTT-DoS).

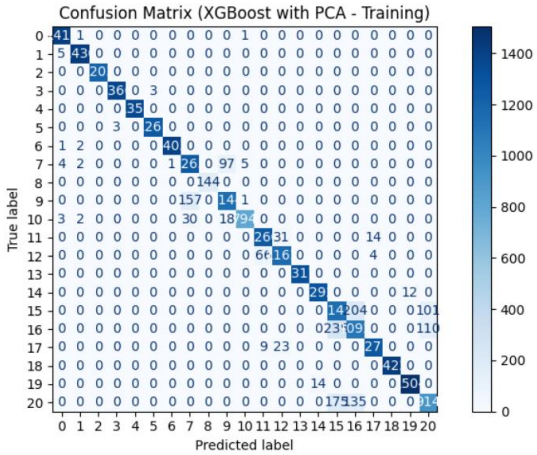

Fig 12: Confusion matrix of the XGBoost model on training data using PCA features

d. **Stacking Classifier**

The Stacking Classifier (Random Forest + Logistic Regression, meta = Logistic Regression) had 62.14% test accuracy with K-Best features, doing good for dominant classes but poor for overlapping attacks. Performance with PCA features was 82.33% accuracy and macro F1 = 0.82, which did good in identifying minority classes (e.g., Recon-Ping_Sweep, Malicious_Packet). PCA removed redundancy and boosted ensemble learning. [13]

e. **Long Short-Term Memory (LSTM)** [12]

An LSTM model (128 units + dropout + softmax) was experimented with by reshaping tabular data into sequential format. Despite having static data, the model discovered hidden temporal relationships, demonstrating promise in sequential attack detection. [12]

With training over the K-Best feature set, the LSTM attained a test accuracy of 56.6%. The confusion matrix (Fig. 29) indicates fair recognition of majority classes like Benign, TCP_IP-DoS, and DoS_UDP but poor identification of minority and closely coupled classes like Recon-Ping_Sweep and Malicious_Telnet. Off-diagonal entries reveal the model's poor generalization in static tabular features without temporal hints. [7, 15]

With PCA-processed features, accuracy was raised to 61.85%, and confusion matrix (Fig. 30) exhibited higher diagonal dominance, especially in MQTT-DoS, ARP_Spoofing, and Malicious_Packet. However, precision loss was still evident, demonstrating the compromise between removal of noise and elimination of feature ordering essential for sequential learning. Generally, LSTM was fairly successful with dominating classes but fell behind tree-based and ensemble models. This verifies that temporal models fail on static intrusion datasets unless temporal structure is designed specifically. [7, 15]

f. **Hyperparameter Optimization with TPOT**

Manual hyperparameter tuning is ineffective for multi-model pipelines. TPOT, a genetic programming-based tool, was employed to automate it. TPOT utilizes genetic programming to optimize preprocessing, feature selection, and hyperparameter sets for cross-validated accuracy. In the current research, TPOT optimized models such as XGBoost, Stacking Classifier, and Random Forest, determining the optimal learning rates, tree depths, and estimator numbers. This diminished development time while continually enhancing test accuracy and minimizing overfitting. The repeatability and flexibility obtained through TPOT make the framework responsive to the dynamic hospital setting that is confronted with shifting attack patterns. [5, 7, 15]

g. **Explainable AI (XAI) Integration**

With the sensitive nature of healthcare information, making models explainable was imperative. SHAP and LIME were combined to enhance transparency. SHAP provided global and local interpretability by quantifying each feature's contribution to predictions, whereas LIME explained single outputs using locally trained models. The combination of these tools facilitated trust, regulatory compliance, and debugging by exposing surprising decision-making. Their inclusion made the framework not only accurate but also transparent and helpful for hospital cybersecurity. [12, 14, 19, 23]

h. **Evaluation Metrics**

Performance of the model was measured with different metrics to handle class imbalance. Accuracy indicated overall accuracy, precision and recall indicated exactness and completeness of detection, Macro F1 indicated balanced performance across all classes, and weighted F1 considered imbalanced distributions. Confusion matrices also provided class-level misclassifications. These measures provided a global overview of robustness and generalizability, particularly in detection of minority attack types essential for hospital network security.

i. **Comparative Analysis of Model Performance**

The Table7 summarizes the performance of each model on the test dataset using the two feature selection techniques

Table7: performance of each model on the test dataset using the two feature selection techniques:

Model	Feature Set	Accuracy	Macro F1	Weighted F1
XGBoost	K-Best	80.95%	0.84	0.85
XGBoost	PCA	78.70%	0.82	0.82
Stacking Classifier	K-Best	62.14%	0.60	0.61
Stacking Classifier	PCA	82.33%	0.82	0.82
LSTM	K-Best	56.60%	0.57	0.58

4. Discussion

The research assessed a machine learning pipeline to detect multi-class intrusions in hospital networks, whose patient records and medical infrastructure are under high cybersecurity threat. We compared models such as XGBoost,

Stacking, LSTM, and CNN with two approaches for feature selection, K-Best and PCA. This uncovered significant trade-offs between model structure and data representation. [5, 7, 15]

- Model performance and data characteristics:

XGBoost with K-Best features and Stacking with PCA always had higher test accuracy and macro F1 score compared to the other models. The advantage is the ability to manage high-dimensional tabular data. The XGBoost uses discriminative features that are extracted by K-Best in an effective manner, whereas the PCA ensured the stability of the ensemble by eliminating the feature correlations. On the other side, the deep learning models were not very impressive with hyperparameter tuning. The LSTM, with its sequential dependence design, could not discover salient temporal patterns in static traffic data. The CNN performed slightly better with PCA but it was still constrained by the unordered feature problem. Therefore the results point out that the deep models require temporal or spatial structure, which is missing in the dataset. [5, 7, 15]

- **Impact of feature selection.**

Feature reduction was a critical factor. K-Best boosted models such as XGBoost by separating the most predictive features, enhancing efficiency and minimizing overfitting. PCA positively impacted models dependent on orthogonal inputs, particularly Stacking and CNN, by reducing dimensions and enabling generalization. PCA had a negative impact on LSTM by distorting input semantics. Feature selection therefore needs to correspond with the architecture: PCA boosts ensemble learners but impairs sequence-based models. [5, 7, 15]

- **Interpretability and deployment.**

Precision on its own is not enough in healthcare cybersecurity. For the sake of trust and responsibility, explainable AI algorithms—SHAP and LIME—were incorporated, facilitating global and instance-level explainability. The tools uncovered feature contributions, allowing IT staff to verify and comprehend predictions. Meanwhile, TPOT AutoML was helpful for optimizing pipelines, minimizing tuning by hand but ensuring flexibility against changing threats. Combined, interpretability and automation enhanced the framework's usability for deployment in hospitals. [12, 14, 19, 23].

System Implementation and Interface

A web application was created to implement the framework, linking machine learning models to end-users in an intuitive and secure way. The system was constructed with a Flask backend and responsive Bootstrap-based frontend, with models serialized through joblib for real-time inference.

User authentication: A safe registration/login module provides access control, storing necessary information for accountability as shown in (Fig13).

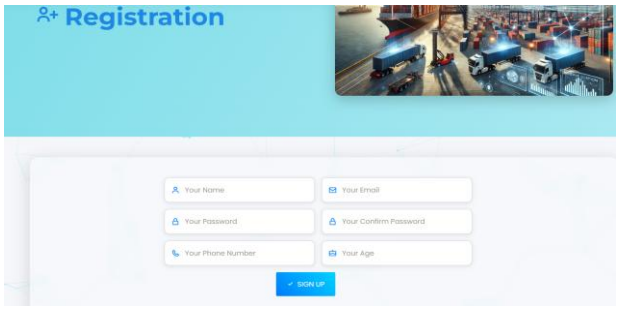

Fig 13: User registration interface for secure access to the hospital intrusion detection system

Model selection: Users have the ability to choose algorithms (XGBoost, Stacking, CNN, LSTM) and inspect reported training accuracy to inform selection as shown in (Fig 14). [5, 7, 15].

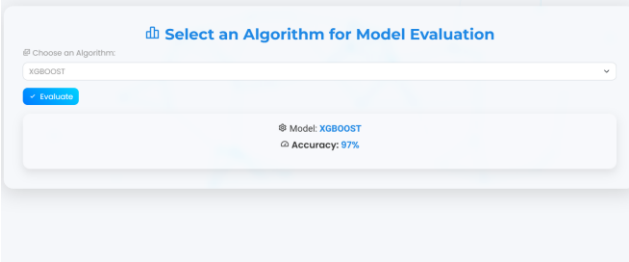

Fig 14: Model selection and evaluation screen showing accuracy results for XGBoost. [5, 7, 15]

Prediction interface: Network traffic characteristics (e.g., IAT, rates, protocol flags, summarized statistics) may be entered manually. Predictions are returned immediately with class labels such as Benign or DoS_UDP, allowing real-time examination as shown in (Fig 15). [7, 15]. This interface allows non-technical staff to interact with models, test scenarios, and receive actionable results without needing in-depth ML expertise.

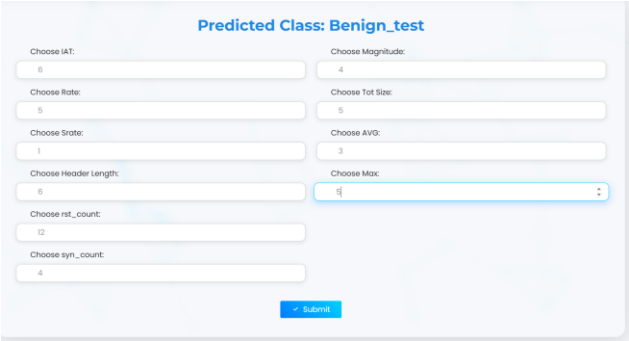

Fig 15: Prediction interface where users input selected features and receive classification results.

5. Conclusion and Future Work

This work suggested a multi-level ML model for the detection of various cyberattacks in hospital networks. The Stacking Classifier with PCA performed the best overall (82.33% accuracy, macro F1 = 0.82), followed closely by XGBoost with K-Best (80.95% accuracy, macro F1 = 0.84). Deep models were not as effective, mirroring the non-sequential nature of tabular network data. [5, 7, 15]

Explainability tools (SHAP, LIME) guaranteed interpretability, with TPOT showing the merit of automatic hyperparameter optimization for flexible deployment. Future efforts will involve the use of temporal traffic flow to leverage the sequential models more effectively, extending to federated environments for cross hospital training, and improving the user interface with alerting and visualization. [12, 14, 19, 23].

References

1. H. Deng and G. Runger, “Feature Selection via Regularized Trees,” Proc. Int. Joint Conf. Neural Netw. (IJCNN), pp. 754–759, IEEE, 2012.
2. P. Di Mauro, G. Galatro, G. Fortino, and A. Liotta, “Supervised Feature Selection Techniques in Network Intrusion Detection: A Critical Review,” arXiv:2104.04958, 2021.
3. M. Alkasassbeh, “An Empirical Evaluation for the Intrusion Detection Features Based on Machine Learning and Feature Selection Methods,” arXiv:1712.09623, 2017.
4. N. Moustafa and J. Slay, “A Hybrid Feature Selection for Network Intrusion Detection Systems: Central Points,” arXiv:1707.05505, 2017.
5. S. S. Dhaliwal, A.-A. Nahid, and R. Abbas, “Effective Intrusion Detection System Using XGBoost,” Information, vol. 9, no. 7, art. 149, 2018.

6. Network Intrusion Detection with XGBoost and Deep Learning Algorithms: An Evaluation Study, Proc. IEEE Conf., 2021.
7. S. Talukder et al., “A Cloud-Based Hybrid Intrusion Detection Framework Using XGBoost and ADASYN-Augmented Random Forest for IoMT,” IET Commun., 2024.
8. M. Alamin Talukder, K. Fida Hasan, and M. M. Islam, “A Dependable Hybrid Machine Learning Model for Network Intrusion Detection,” J. Inf. Secur. Appl., vol. 63, art. 103405, 2022.
9. Y. Zhou, G. Cheng, S. Jiang, and M. Dai, “Building an Efficient Intrusion Detection System Based on Feature Selection and Ensemble Classifier,” arXiv preprint arXiv:1904.01352, Apr. 2019.
10. W. W. Lo, S. Layeghy, M. Sarhan, M. Gallagher, and M. Portmann, “EGraphSAGE: A Graph Neural Network based Intrusion Detection System for IoT,” arXiv preprint arXiv:2103.16329, Mar. 2021.
11. S. Talukder, A. M. Talukder, and M. M. Islam, “A Dependable Hybrid Machine Learning Model for Network Intrusion Detection,” arXiv preprint arXiv:2212.04546, Dec. 2022.
12. D. Gaspar, P. Silva, and C. Silva, “Explainable AI for Intrusion Detection Systems: LIME and SHAP Applicability on MultiLayer Perceptron,” IEEE Access, vol. 12, pp. 30164–30175, 2024.
13. S. Rajagopal, P. Kundapur, and H. S., “A Stacking Ensemble for Network Intrusion Detection Using Heterogeneous Datasets,” Security and Communication Networks, vol. 2020, Art. ID 4586875, 2020.
14. A. Razib et al., “An Explainable Deep LearningEnabled Intrusion Detection Framework in IoT Networks,” Information Sciences, vol. 639, pp. 119000, 2023.
15. B. Bhardwaj, S. Rawat, and H. B. Maringanti, “Network Intrusion Detection Using Hyper Parameter Tuned Weighted XGBoost Classifier,” Journal of Electrical Systems, vol. 20, no. 3, 2024.
16. S. Malek, B. Trivedi, and A. Shah, “Leveraging AIDriven Realtime Intrusion Detection by Using WGAN and XGBoost,” Proc. 11th Int. Symp. Inf. & Comm. Technol., 2020.
17. N. Bhati and C. S. Rai, “An Improved EnsembleBased Intrusion Detection Technique Using XGBoost,” Trans. Emerging Telecommun. Technol., 2023.
18. C. Manimurugan et al., “Effective Attack Detection in Internet of Medical Things Smart Environment Using a Deep Belief Neural Network,” IEEE Access, vol. 8, pp. 77396–77404, 2020