
Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
 Volume 25, Issues 2, pp. 1246–1262, DOI 10.12694/scpe.v25i2.2657

ENHANCED FEATURE OPTIMIZATION FOR MULTICLASSS INTRUSION DETECTION IN IOT FOG COMPUTING ENVIRONMENTS

SUDARSHAN S. SONAWANE*

Abstract. To overcome the shortcomings of traditional security measures in IoT fog computing, The Multiclass Intrusion Detection (MCID) model is put forward. The model's goal is to improve intrusion detection by identifying and classifying different attack types. The behavioral, temporal and anomaly features are fused through SVM-BFE for obtaining the best possible selection of high worth features. Finally we use a Random Forest algorithm to robustly classify them. There is also its adaptability to the ever-changing security demands of fog computing. MCID's ability to improve the cloud security of fog computing is shown by a 4-fold cross validation, which returns performances including precision rates up to 99.43 %, recall about 95 % and F-measures at as much as 97.17 %. Moreover there are specificity rate totals coming in over this whole range that hit close or right

Key words: Internet of Things (IoT); Intrusion Detection System; UNSW-NB15; Fog Computing; SVM-BFE

1. Introduction. The Internet of Things (IoT) has grown quickly, integrating a wide range of devices and facilitating easy data exchange and communication [1]. This evolution calls for the need for an intermediary computing layer between cloud computing and IoT devices [2]. By processing data closer to its source and increasing efficiency, fog computing was able to solve this issue. Fog computing crosses over devices with disparate operating systems, which presents a security risk. Fog computing raises the stakes for security, especially for intrusion detection. In decentralized environments, traditional detection systems that are built for centralized computing architectures perform poorly [3]. The use of IoT in healthcare, transportation, and defense makes security issues more important to handle.

This research aims to enhance intrusion detection, specifically for fog computing, in light of these challenges. The MCID model takes this into account. Behavioral, temporal, and anomaly detections are all part of the MCID model. The model optimizes feature selection using SVM-BFE optimization to enhance detection. To evaluate the applicability of the MCID model, we conducted extensive testing against established benchmarks. The NUSW-NB15 dataset offers a strong basis for evaluation. For testing, we also employed the Random Forest algorithm. MCID's performance is evaluated using this method, which contrasts it with other models. The MCID model aims to enhance fog computing security. Ensuring a robust intrusion detection framework is our goal in order to increase IoT safety and reliability.

This article follows a logical structure. Following a discussion of fog computing's challenges, we evaluate the MCID model's performance and implications. In order to explain Fog Computing security challenges and solutions, this organized approach tries to highlight the critical role that the MCID model plays in addressing them.

2. Related Work. We face both tremendous security challenges and exciting opportunities as fog computing and the Internet of Things grow quickly. Risks increase as more devices connect and systems get more complicated. Recent research on these risks is examined in the "Related Research" section. New threat mitigation methods have been proposed, the flaws of older security systems in fog environments have been investigated, and their effectiveness has been put to the test. Everything from basic, efficient security to sophisticated deep learning protection is covered in this section.

Belal Sudqi Khater et al. [4] focused on addressing resource limitations in traditional intrusion detection systems when applied in fog computing environments. Fog computing extends cloud capabilities to network edges using diverse hardware devices, posing challenges for resource-intensive intrusion detection systems. The article's objective is to propose a lightweight intrusion detection system capable of running effectively on

*Department of Computer Engineering, R. C. Patel Institute of Technology, Shirpur, India (sudars2000@gmail.com)

resource-constrained fog devices while maintaining high detection rates with minimal computational overhead. To achieve this, the authors introduce a novel perceptron-based intrusion detection system that utilizes vector space representation and a multilayer perceptron model. Experimental results reveal a remarkable achievement, with the system delivering a 94% Accuracy, 95% Recall, and 92% F1-Measure in one dataset and 74% Accuracy, 74% Recall, and 74% F1-Measure in another. Moreover, the system operates efficiently on a Raspberry Pi 3 fog device, with low energy consumption, demonstrating its suitability for fog computing environments.

KISHWAR SADAF et al. [5] proposed a novel approach using deep learning techniques, Autoencoder and Isolation Forest. The objective is to differentiate between normal and attack packets in real-time with high accuracy, enhancing the security of fog devices. The article introduces the Auto-IF method, combining Autoencoder and Isolation Forest, to process incoming network packets efficiently and detect malicious attacks in real-time while minimizing false positives. The authors extensively evaluate the method, utilizing the NSL-KDD dataset, and report superior performance compared to other intrusion detection approaches. Notably, the Auto-IF method effectively reduces false positives and negatives, particularly when the dataset's contamination percentage is around 10-12%. This contribution offers an efficient solution to bolster the security of fog devices and ensure reliable service delivery.

Prabhat Kumar et al. [6] confronted the escalating security concerns surrounding Internet of Things (IoT) networks. The exponential growth of connected IoT devices brings with it an increased risk of attacks and compromise. These devices, often resource-constrained, are susceptible to security breaches that can lead to unauthorized data access, physical infrastructure damage, or their enlistment in botnets for further attacks. Traditional security measures prove inadequate for safeguarding IoT networks. Consequently, the article introduces a distributed ensemble design-based intrusion detection system, leveraging machine learning and fog computing to enhance intrusion detection's accuracy and efficiency in IoT networks. The aim is to protect these networks from diverse attacks and bolster their security.

K. Kalaivani et al. [7] tackled the security challenges prevalent in cloud and fog computing environments, which are vulnerable to various types of attacks such as DDoS, Worm, Probe, R2L, and U2R. These attacks can disrupt network resources and consume bandwidth, posing significant threats. Conventional methods often fall short in accurately detecting and countering these attacks. The article presents a novel solution—a hybrid deep learning intrusion detection model named ICNN-FCID. This model combines Intrusion Classification Convolutional Neural Network (ICNN) with Long Short-Term Memory (LSTM) networks and is specifically designed for fog computing environments. ICNN-FCID demonstrates remarkable effectiveness, achieving an accuracy of about 96.5% and effectively classifying various attack types. By proposing this model, the article contributes to enhancing security in cloud and fog computing environments.

Mohammed Anbar et al. [8] addressed the imperative need for effective intrusion detection systems (IDSs) in Fog computing. Fog computing, an extension of cloud computing, shifts computational resources closer to network edges, reducing communication costs and boosting system performance. However, its distributed nature poses new security challenges, necessitating robust IDSs. This article conducts a comprehensive review of recent IDS research in Fog computing, categorizing various approaches and highlighting their challenges and limitations. The review provides a new taxonomy of IDSs, outlines shortcomings such as scalability and resource constraints, and offers potential solutions and future research directions. In sum, this article contributes to an improved understanding of IDSs in the Fog computing landscape, guiding further research in this vital domain.

Hassan Adegbola Afolabi et al. [9] addressed the pressing security and privacy concerns surrounding IoT devices connected via Fog computing. These devices, due to limited resources and inadequate security, are susceptible to various threats like DDoS attacks and brute-force attacks. The article proposes an Intrusion Detection System (IDS) model utilizing Back Propagation Deep Neural Network (BP-DNN) to safeguard the Fog-IoT platform. The model aims to enhance accuracy and detection rates, mitigating security breaches in IoT devices within Fog computing.

M. Ramkumar et al. [10], proposed a Support Vector Machine (SVM)-based Intrusion Detection System (IDS) tailored for the unique challenges of fog computing. The SVM algorithm is employed for its efficiency, and the IDS focuses on characteristics like mean, limit, and median systems to detect intrusions efficiently. Offering a lightweight, efficient IDS solution for fog computing with a low false-positive rate and an impressive average detection rate of 98.6% is the contribution.

An extensive survey examining intrusion detection systems for fog and cloud computing is carried out by Victor Chang et al. [11]. The article seeks to shed light on various aspects of security policy deployment in these kinds of environments. Software-as-a-Service (SaaS) and intrusion detection are highlighted as methods for tracking and evaluating network traffic, and strategies and policies for lowering intrusion detection are also suggested. Despite lacking its own experimental results or statistics, the article is still a useful resource for understanding security policy deployment in fog and cloud computing environments.

Within fog computing, Wu et al. [12] presented a novel Intrusion Detection System (IDS) designed specifically for Internet of Things networks. With Convolutional Neural Network (CNN) as the first classifier and Random Forest as the second, this IDS uses a distributed ensemble design. By using this method, the efficiency and accuracy of identifying malicious attacks on fog nodes will be improved. The suggested model considerably improves detection performance by overcoming the drawbacks of conventional IDS, attaining an astounding 94.43% overall accuracy while effectively allocating the processing load.

The cybersecurity issues that fog computing (FC) and edge computing (EC) environments face because of their dynamic nature and variety of network devices were addressed by Omar A. Alzubi et al. [13]. Traditional intrusion detection systems (IDS) often fall short in these settings. To address this issue, the article proposes an optimized machine learning-based IDS called Effective Seeker Optimization with Machine Learning-Enabled Intrusion Detection System (ESOML-IDS). This system leverages feature selection and parameter optimization through effective seeker optimization and machine learning. Comparative experiments demonstrate the ESOML-IDS's remarkable effectiveness, achieving an accuracy of 99.5% and outperforming other IDS models in FC and EC environments.

Junaid Sajid et al. [14] addressed the susceptibility of UAVs to cyberattacks in smart farming, proposing a fog computing-based framework to bolster security. By combining UAVs and IoT sensors for data collection, a fog computing architecture for data transmission, and an intrusion detection system that employs machine learning, the system identifies compromised UAVs. The framework not only enhances data collection security but also introduces a coin-based recharge system to reward benign UAV behavior while efficiently managing resources.

Kalaivani Kaliyaperumal et al. [15] confronted the vulnerability of fog computing systems to a range of attacks and offer an efficient solution. Their proposed model combines deep learning techniques, such as CNN and IDS-AlexNet, with Random Forest for attack detection. Leveraging feature selection, this model significantly improves attack detection accuracy, achieving an impressive 97.5% accuracy on the UNSW-NB15 dataset. This contribution enhances fog computing security and represents a substantial step toward safeguarding these systems from various threats.

Doaa Mohamed et al. [16] tackled the increasing vulnerability of IoT networks to cyber threats by proposing EHIDS, a hybrid intrusion detection system combining fog and cloud computing. EHIDS aims to identify abnormal behavior in IoT devices and networks to prevent system intrusions. The article details the architecture and implementation of EHIDS and conducts comprehensive experiments to showcase its superior performance compared to other approaches. EHIDS reduces execution time significantly and demonstrates effectiveness in enhancing IoT security.

Cristiano Antonio de Souza et al. [17] provided a comprehensive overview of intrusion detection in IoT and Fog Computing. Their article discusses the challenges, principles of Machine Learning techniques, and state-of-the-art approaches in intrusion detection. While the authors mention experiments with the IoTID20 dataset, they primarily focus on discussing the principles and challenges of intrusion detection, guiding researchers in this field with a rich source of information.

Guosheng Zhao et al. [18] addressed security risks in cloud-fog hybrid computing for IoT by proposing a lightweight intrusion detection model based on ConvNeXt-SF. Their model significantly improves efficiency and performance while maintaining detection and learning capabilities. Detailed experimental results show its superiority, reducing training and prediction times while enhancing accuracy and addressing information security risks effectively. This contribution offers an innovative solution for improving IoT security in cloud-fog hybrid computing environments.

According to research in the cited articles, there are serious security risks associated with the Internet of Things' (IoT) rapid expansion and integration across numerous industries. According to these studies,

IoT intrusion detection systems (IDS) and fog computing are being developed to meet their specific security requirements. The impact of fog environments on traditional IDS resource constraints is discussed by Belal Sudqi Khater et al. [4]. While Prabhat Kumar et al. [6] and M. Ramkumar et al. [10] stress the need for sophisticated algorithms and specialized systems to address vulnerabilities brought on by the growth of connected devices, KISHWAR SADAF et al. [5] use deep learning to improve detection rates.

Different approaches are provided in this domain by the Lightweight Intrusion Detection Model (LIDM) [18] and the Combined Ensemble Intrusion Detection Model (CEIDM) [15]. One layered defense against intrusions is provided by the CEIDM's multiple detection methods. Limitations on IoT resources are addressed by LIDM, which guarantees efficient intrusion detection with low computational overhead.

The proposed MCID model is pertinent and required in light of this situation. Despite being effective, many of the current methods only deal with particular problems. Due to the complexity of fog computing, a broad-spectrum IDS is required. Behavioral, temporal, and anomaly-based detections are addressed by the MCID model, which takes a tri-dimensional approach to fog computing's many challenges. To enhance detection, MCID additionally employs SVM-BFE optimization. MCID's robustness is highlighted by its efficient yet comprehensive solution, which stands in contrast to the lightweight and ensemble methods of CEIDM and LIDM.

The reviewed articles take on key security issues in the rapidly expanding IoT and fog computing space. To avoid resource constraints in fog devices, Khater et al. [4] and Afolabi et al. [9] suggest lightweight IDS solutions Deep learning is also used by Sadaf et al. [5] to improve real-time attack detection using AutoIF and Kalaivani etal. [15], implementing ICNNFCID respectively. This leads to greater accuracy and fewer false positives in the methods. Kumar et al. [6] and Ramkumar et al., advocate for machine learning-based IDS, with consideration towards distributed designs (node autonomous monitoring) to guard against multiple attacks of different types. In addition the researchers focus on efficiency in SVMs based filtration mechanism Anbar et al. [8] and Zhao et al. [18], provide detailed overviews, stressing the importance of an effective policy for security in fog computing systems and cloud communities, respectively. Wu et al. [12] and Alzubi et al. [13], respectively, propose new IDS models utilizing ensemble learning technologies or optimization techniques that have high accuracy and impressive performance records. Sajid et al. [14] proposed a fog computing framework for securing UAVs in smart farming systems. Furthermore, Mohamed et al. [16] and Souza et al. [17] look at hybrid intrusion detection as well as the basics of machine learning in IoT security, demonstrating how hard it is to detect attacks on an IoT network.

Based on the literature reviewed, it is apparent that the MCID framework stands out as an important contribution to the area of IoT fog computing security. With fog computing, intrusion detection is a unique challenge. The research landscape shows that there are various attempts to solve it; these efforts focus on resource efficiency, real-time ability and accuracy. Yet, there still is a need for an integrated solution that solves these individual problems. This is the integrated approach offered by MCID, which integrates state-of-the art feature optimization and classification techniques. It promises to avoid some of the limitations of existing models while allowing for a robust system that will be more than adequate for use in highly complex fog computing environments.

3. Methods and Materials. The proposed model addresses a critical need in the realm of IoT fog computing, where the amalgamation of diverse computing entities and the decentralized nature of fog computing render traditional security measures insufficient. The convergence of myriad devices and the ensuing data explosion necessitate advanced intrusion detection mechanisms that can adeptly identify and categorize varied intrusion types, ensuring the security and reliability of fog computing ecosystems. MCID, with its intricate feature optimization and multiclass classification capabilities, is not merely a response to this need but a pioneering initiative, offering nuanced insights and enhanced detection accuracies in distinguishing between normal and malicious activities. The significance of MCID lies in its ability to harness advanced machine learning techniques, optimizing feature selection using SVM-BFE and employing ensemble learning for refined classification, thereby contributing to the fortification of IoT fog computing environments against escalating and evolving cyber threats. The justification for deploying this model is underscored by the urgent imperative to secure multifaceted and dynamic fog computing landscapes, where conventional security protocols are rendered obsolete, and the demand for sophisticated, adaptive, and robust intrusion detection mechanisms is ever-increasing.

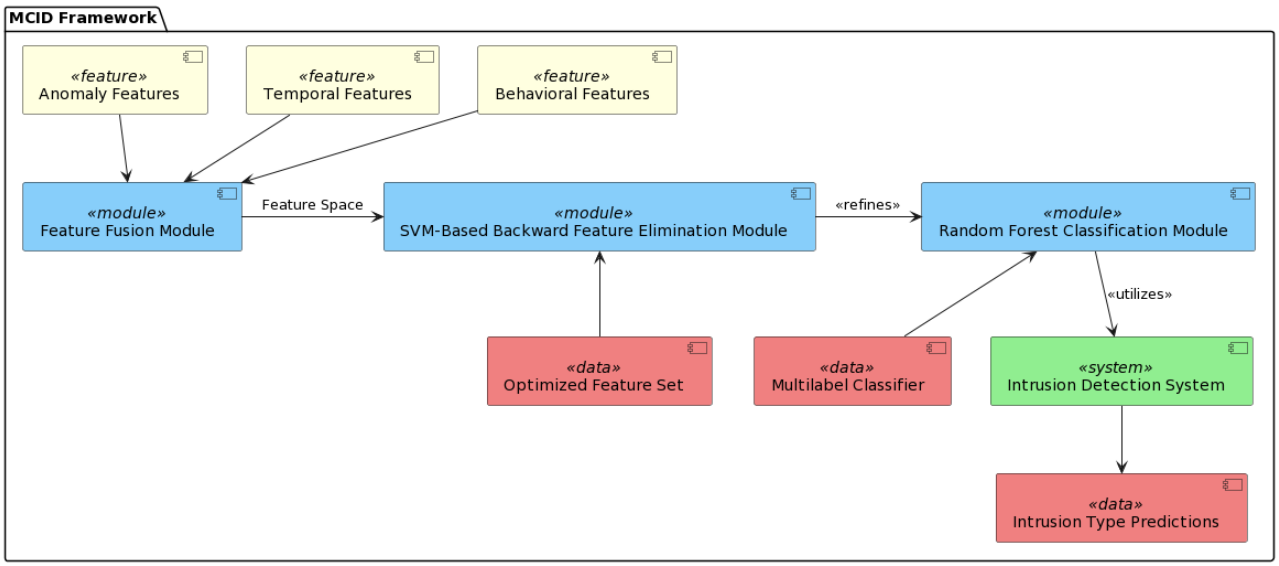

Fig. 3.1: MCID Architecture

In essence, MCID stands as a beacon of innovation, promising enhanced security and resilience in IoT fog computing, paving the way for the safe and seamless integration of fog computing in our interconnected digital world.

The MCID architecture that shown in figure 3.1 is designed to optimize feature selection and enhance intrusion detection capabilities in IoT fog computing environments. This architecture is embedded with a fusion of Behavioral, Temporal, and Anomaly features to construct a multidimensional feature space capable of detailed representation of data patterns and nuances, enabling heightened discernibility between varied intrusion types and normal behaviors.

The optimal feature selection executed through a combination of Support Vector Machine-Based Backward Feature Elimination (SVM-BFE), focusing on refining the feature space to retain the most informative and discriminative features, while eliminating the redundant and less contributive ones. This feature optimization is crucial for enhancing the model's predictive accuracy and computational efficiency, thereby contributing to a more responsive and effective intrusion detection system (IDS).

The architecture figure 3.1 employs a sophisticated multilabel classification model, utilizing ensemble learning with a Random Forest algorithm. This model is tailored to classify diverse intrusion types, such as DDoS, backdoor, ransomware, and normal behaviors, facilitating a comprehensive intrusion detection scope within IoT fog computing landscapes. The ensemble learning approach empowers the model with robustness and stability, aggregating insights from multiple decision trees to ensure reliable and accurate intrusion type predictions. In the MCID architecture, every component, from feature fusion and optimal selection to multilabel classification, is meticulously integrated and fine-tuned. This results in an advanced IDS that is capable of identifying and differentiating multiple intrusion types with enhanced precision, fostering a more secure and resilient environment within the complex and dynamic ecosystems of IoT fog computing.

3.1. The Feature. The basis for intricate and successful models in intrusion detection systems is provided by behavioral, temporal, and anomalous features, particularly in dynamic and complex environments such as IoT fog computing. These features enhance detection capabilities by offering various angles and dimensions for analyzing and comprehending the intricate patterns and subtleties of the data.

Activity patterns of network entities are known as behavioral features. They reveal the typical and atypical behavior of entities through describing interactions, action sequences, and state transitions. These characteristics enable models to pick up on intricate behavioral details that might point to malicious activity or security

lapses. These features include user behaviors, system interactions, and network communication patterns. Early detection of anomalies and intrusions is facilitated by an understanding of these features.

On the other hand, temporal features are necessary to record the time-related elements and sequences of network events. Timestamps, durations, activity frequency, and the intervals between events or actions are some examples of these features. In order to identify suspicious or unusual patterns and trends, intrusion detection models can use time-based features to analyze activity sequences and timings. A layer of security is added by identifying sophisticated and stealthy attacks that use time to go undetected through the analysis of temporal dynamics.

Anomaly Features draw attention to anomalies, outliers, and deviations from the norm in the data. Statistical measurements, frequency distributions, and outlier scores—which indicate data deviation—are some examples of these features. Detection models can detect unusual patterns and irregularities, even new ones, by integrating anomaly features. By identifying new threats, abnormal features aid in the adaptation and survival of intrusion detection systems.

3.2. Optimal Feature Selection. Feature selection is a crucial process in the realm of machine learning, especially in scenarios where the data dimensionality is large. It becomes even more significant when applied to sensitive domains like intrusion detection in IoT fog computing environments. This is because irrelevant or redundant features can lead to overfitting, poor performance, and reduced interpretability.

SVM is a supervised machine learning algorithm, primarily used for classification tasks. It works by finding the hyperplane that best separates the classes in the feature space. Given its capability to handle high-dimensional data and its effectiveness in finding margin-based distinctions, SVM is a suitable model for the complex and dynamic environment of IoT fog computing.

BFE is a recursive method that begins by training the model using all available features. Post training, it evaluates model performance and identifies the least significant feature. This feature is eliminated, and the model is retrained with the remaining features. This process is repeated until a desired number of features is reached or further elimination results in unacceptable performance degradation.

Support Vector Machine (SVM) [19] combined with Backward Feature Elimination (BFE) offers an effective methodology for optimal feature selection. Here's a comprehensive description of this method in the context of intrusion detection:

The SVM-BFE (Support Vector Machine-Backward Feature Elimination) process (see figure 3.2) starts with standardizing the data due to the sensitivity of SVM to the scale of input data and initializing the model with all available features. The initial SVM model, trained with all features, undergoes performance assessment using metrics such as accuracy, F1 score, or the area under the ROC curve, preferably using cross-validation methods on the training set. The least significant feature, which has the minimum impact on model performance, is then identified and eliminated from the feature set, and the model is retrained with the reduced feature set. This process of feature elimination and model retraining is iteratively carried out, with the model's performance being assessed at each step, until further removal degrades the model's performance significantly, or a predetermined number of features are left.

After the iterative process, the remaining features constitute the optimal feature set, which is expected to provide the best generalization on unseen data. The final model, built using this optimal feature set, undergoes validation against a separate test set to assess its generalization capability, ensuring its reliability and effectiveness in accurately classifying and predicting the target variables. This meticulous and iterative process aims to strike a balance between model simplicity and performance, enhancing the model's predictability and interpretability, crucial for applications such as intrusion detection in IoT fog computing environments.

Algorithm. Let $X = x_1, x_2, ..., x_m$ be the feature space where x_i represents each feature, and mis the total number of features.

Let Y be the set of class labels, $Y = y_1, y_2, ..., y_n$.

Let D be the dataset represented as $D(x^1, y^1), (x^2, y^2), ..., (x^N, y^N)$, where N is the number of samples.

SVM Decision Function. SVM finds the optimal hyperplane that separates the different classes in the feature space. The decision function of a linear SVM can be represented as: $f(x) = w^T x + b$ where $f(x)$ is the decision function. w is the weight vector. x is the input feature vector. b is the bias term.

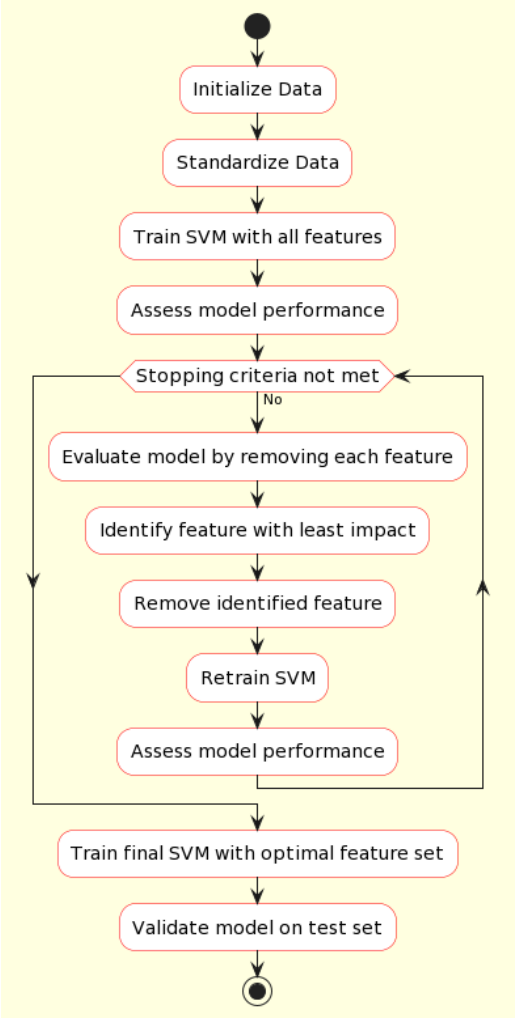

Fig. 3.2: SVM-BFE flow diagram

Backward Feature Elimination (BFE). In the Backward Feature Elimination process, the least important feature is iteratively removed until an optimal subset of features is obtained. This can be represented as: Start with the full feature set:

$$F = x_1, x_2, ..., x_m \tag{3.1a}$$

For each feature x_i in F, evaluate the SVM model, removing x_i from the feature set and note the performance. Identify the feature whose removal has the least impact on (or improves) the model's performance:

$$x_{remove} = argmin_{(x_i F)} P(F - x_i) \tag{3.2a}$$

where $P(F - x_i)$ represents the performance of the model trained with the feature set F excluding the featurex_i. Update the feature set by removing the identified feature: Eq 3.3

$$F = F - \{x_{\text{remove}}\} \tag{3.3a}$$

Repeat steps 2-4 until a stopping criterion is met, such as a pre-defined number of features or a performance threshold.

Objective. The objective is to find the optimal subset of features, F^*, which maximizes the performance of the SVM model:

$$F* = argmax_F P(F) \tag{3.4a}$$

Stopping Criteria. The iterative process stops when either:

The addition of any feature does not improve the model's performance. A predefined number of features is reached.

Final Model. The final optimal SVM model with the selected features is given by:

$$f^*(x) = w^{*T}x + b^* \tag{3.5a}$$

where w^* and b^* are the optimal weight vector and bias term obtained after training the SVM with the optimal feature subset F^*.

3.3. Multiclass classification by Random Forest. IoT Fog Computing forms a pivotal network architecture, allowing data processing, storage, and applications to operate closer to the end users along the cloud-to-thing continuum. Its dynamic and intricate nature makes it susceptible to a variety of cyber threats, including DDoS, ransomware, injections, backdoors, and other malicious activities, highlighting the critical need for robust intrusion detection systems (IDS) [20]. In this context, Random Forest emerges as a potent classification algorithm, proficient in predicting varying intrusion types and discerning normal behavior from malicious. The flow of the multiclass classification using RF [21] has been shown in figure 3.3.

Random Forest is an ensemble learning method, combining multiple decision trees [22] to construct a 'forest' that collaborates to render more accurate and stable predictions. This algorithm is inherently suited for multiclass classification tasks, making it apt for identifying diverse intrusion types like DDoS [23], ransomware [24], injection [25], backdoor [26], and normal activities within. IoT fog computing ecosystems [27]

Figure 3.3 depicts the workflow of multiclass intrusion detection module of the MCID that begins with loading a raw dataset and then extracts behavioral, temporal, anomaly features through feature engineering to form comprehensive feature matrix X; next is preprocessing data devoid of missing values or normalization being processed as input required for Data is pre-processed, then split into training data and testing data. The former is used to train a Random Forest model which employs an ensemble approach by constructing numerous decision trees at different depths; the result of classifying the example chosen for given features are aggregated from all these trees so that output takes one with highest frequency as answer. Each tree in the model gives a vote to classify incoming classification data, which culminates in a majority decision. The final class label is then decided by potential classes such as Normal, DDoS (distributed denial-of-service), Ransomware, Injection or Backdoor. This procedure ends up with hyper-parameter tuning and optimizing, to raise as high the model accuracy possible so that the intrusion detection system can be made more reliable and effective. Detailed description of each step involved in this module follows:

Feature Engineering and Preprocessing. In the realm of intrusion detection, feature engineering is a foundational step, where the raw data is transformed into a structured format. It includes the extraction of relevant features like behavioral, temporal, and anomaly features, followed by preprocessing tasks such as normalization, encoding, and handling missing values.

Let D be the raw dataset, and let $X = x_1, x_2, ..., x_n$ represent the set of extracted features after feature engineering, where n is the number of features.

Preprocessing. Normalization:

$$x_{i,\text{norm}} = \frac{x_i - \min(x_i)}{\max(x_i) - \min(x_i)} \tag{3.6a}$$

where $x_{i,\text{norm}}$ is the normalized value of feature x_i.

Training the Model. The preprocessed data is utilized to train the Random Forest model. The algorithm constructs multiple decision trees during training, each deliberating on a random subset of the features, and thus, learning varied aspects of the data. Let $T = t_1, t_2, .., t_m$ be the set of decision trees in the Random Forest, where m is the number of trees. Each tree t is constructed using a subset of the feature set $X' \subset X$ and a subset of the training dataset $D' \subset D$.

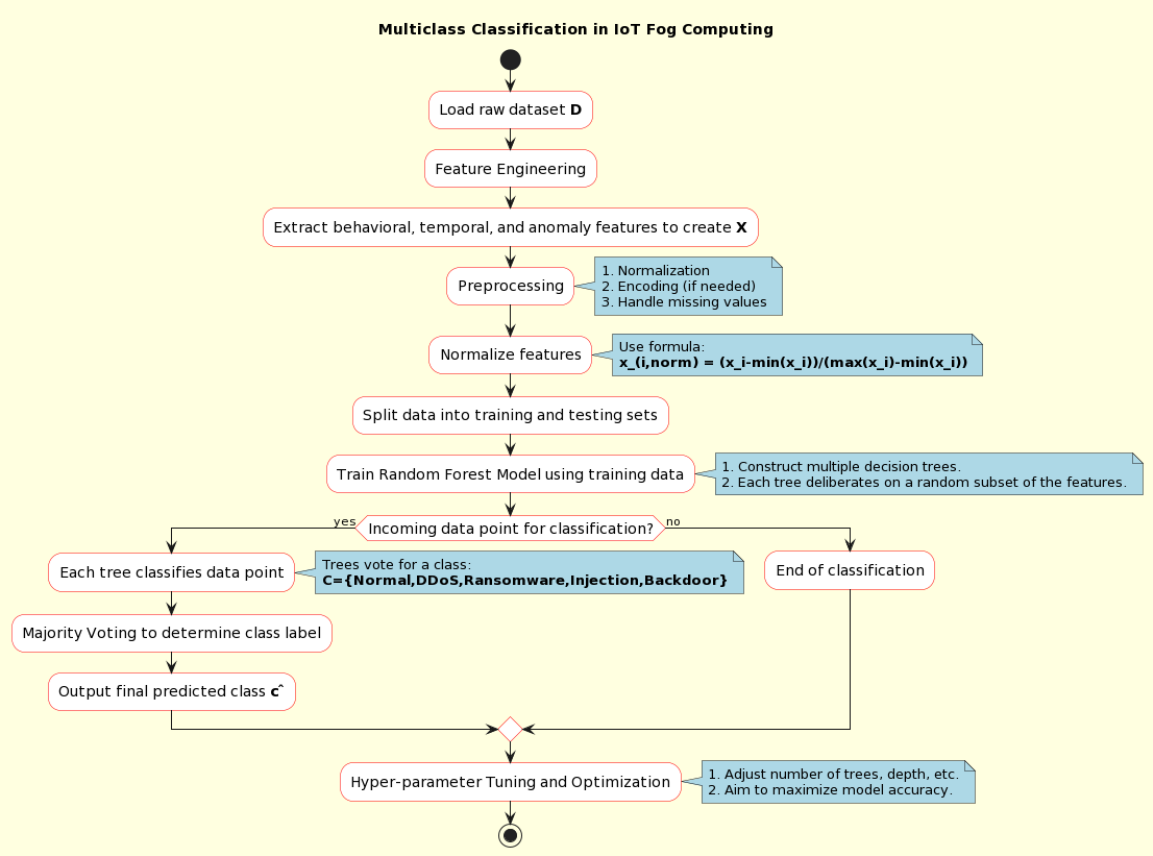

Fig. 3.3: Flow diagram of multiclass classification

Classification and Prediction. For every incoming data point, each tree in the 'forest' makes an individual decision, classifying the data point as either normal, DDoS, ransomware, injection, or backdoor. Subsequently, the Random Forest algorithm applies a majority voting mechanism to finalize the predicted class label, ensuring the collective wisdom of the 'forest' is reflected in every prediction [28]. Each tree t in votes T for a class c in the set of classes C = Normal, DDoS, Ransomware, Injection, Backdoor. The final predicted class $\hat{c}$ is determined by majority voting: $\hat{c} = \arg\max_{c \in C} \sum_{t \in T} 1(t(x) = c)$ where 1 is the indicator function, and $t(x)$ is the class predicted by tree t for input x.

Hyper-parameter Tuning and Optimization. The model undergoes hyper-parameter tuning to refine its performance. Parameters like the number of trees, maximum depth of the trees, and minimum samples split are optimized to enhance the model's predictive accuracy and generalization capability [29]. Let Θ represent the set of hyperparameters of the Random Forest model, such as the number of trees and maximum depth. Hyperparameters tuning aims to find the optimal set of hyperparameters Θ^* that maximize a given performance metric, say accuracy:

$$\Theta^* = argmax_{\Theta} Accuracy(\Theta) \tag{3.7a}$$

4. Experimental Study. The experimental study focuses on the evaluation of "Enhanced Feature Optimization for Multiclass Intrusion Detection in IoT Fog Computing Environments" (MCID) using data from the UNSW-NB15 [30] dataset. Balanced samples for each intrusion type and benign records were extracted, each consisting of 25,000 entries for BENIGN, DDOS, RANSOMWARE, and INJECTION. These were systematically divided into training and test subsets with 18,750 records designated for training and 6,250 for testing

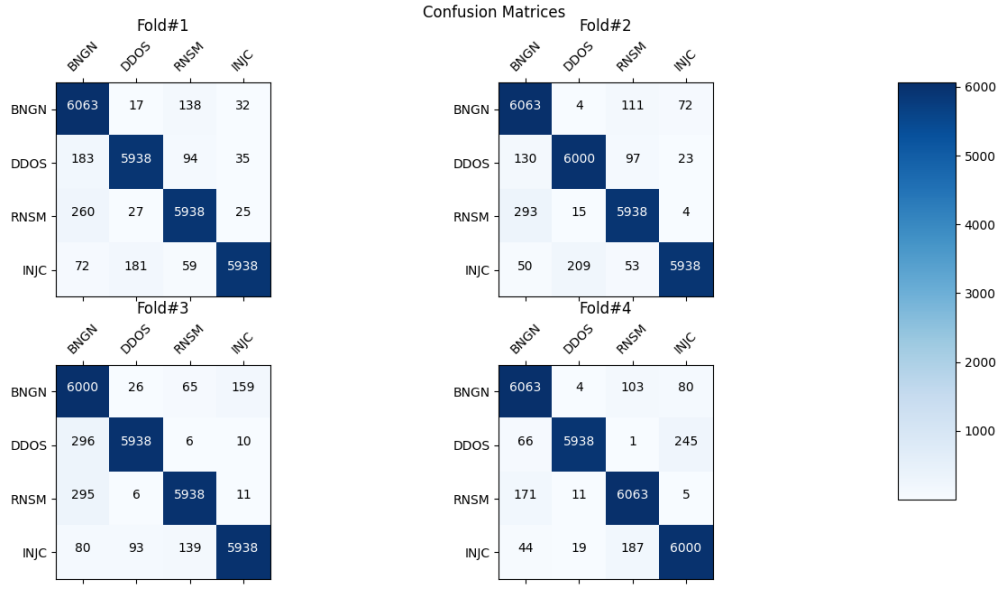

Fig. 4.1: Flow diagram of multiclass classification

across each label.

To validate the model's performance, a four-fold cross-validation method was applied, ensuring consistent and reproducible results across various data partitions. The implementation leveraged Python and its associated libraries, highlighting the capabilities of this programming language in handling such sophisticated tasks.

The effectiveness and relevance of MCID were assessed in comparison to two contemporary models. These include "Lightweight Intrusion Detection Model of the Internet of Things with Hybrid Cloud-Fog Computing" (LIDM) [18] and "Combined Ensemble Intrusion Detection Model using Deep learning with Feature Selection for Fog Computing Environments" (CEIDM) [15]. This comparative analysis offers insights into the model's capabilities, strengths, potential areas for enhancement, and its position in the broader spectrum of intrusion detection in IoT fog computing environments.

4.1. Performance Analysis. We examine the evaluation outcomes of a thorough 4-fold cross-validation exercise on the MCID, LIDM, and CEIDM datasets in the "Performance Analysis" section. To comprehend classification behavior and class predictions, we make use of confusion matrices. This performance metrics analysis teaches readers about recall, f-measure, specificity, precision, and more. The performance of the model is evaluated by looking at both aggregated macro metrics and individual class-wise metrics. The model's performance is given by the overall accuracy metric. This part provides readers with a comprehensive overview of the model's performance on various datasets, including its advantages and disadvantages. **MCID:** The provided confusion matrices in figure 4.1 represent the performance of a classification model across four folds of cross-validation on the MCID dataset. In each matrix, diagonal values depict correct predictions, while off-diagonal entries indicate misclassifications. Across the four folds, the model consistently demonstrates strong performance, particularly in the "DDOS" class, with minimal misclassifications. Notably, there are occasional misclassifications between "BENIGN" and the other classes, especially "RANSOMWARE" and "INJECTION". Despite these, the overall high true positive rates across classes suggest the model's robustness on the MCID dataset.

In the provided metrics of the table 4.1, there is a consistent observation of high performance across all folds for the classes BNGN, DDOS, RNSM, and INJC. For Fold#1, the precision values range from 0.9217 for BNGN to 0.9847 for INJC. Similarly, for Fold#2, the precision values are slightly higher, ranging from 0.9276 for BNGN to 0.9836 for INJC. Fold#3 showcases the lowest precision for BNGN at 0.8994, but it also has the

Table 4.1: Performance Metric Values obtained from 4 fold cross validation performed on MCID

Fold	Metrics	BNGN	DDOS	RNSM	INJC
#1	Precision	0.9217	0.9635	0.9533	0.9847
#1	Recall	0.9701	0.9501	0.9501	0.9501
#1	F-measure	0.9453	0.9567	0.9517	0.9671
#1	Specificity	0.9725	0.988	0.9845	0.9951
#2	Precision	0.9276	0.9634	0.9579	0.9836
#2	Recall	0.9701	0.96	0.9501	0.9501
#2	F-measure	0.9484	0.9617	0.954	0.9666
#2	Specificity	0.9748	0.9878	0.9861	0.9947
#3	Precision	0.8994	0.9794	0.9658	0.9706
#3	Recall	0.96	0.9501	0.9501	0.9501
#3	F-measure	0.9287	0.9645	0.9579	0.9602
#3	Specificity	0.9642	0.9933	0.9888	0.9904
#4	Precision	0.9557	0.9943	0.9542	0.9479
#4	Recall	0.9701	0.9501	0.9701	0.96
#4	F-measure	0.9628	0.9717	0.9621	0.9539
#4	Specificity	0.985	0.9982	0.9845	0.9824

Table 4.2: Accuracy and Macro Metric Values obtained from 4 fold cross validation performed on MCID

Fold Id	Accuracy	Macro Precision	Macro Recall	Macro F-measure	Macro Specificity
#1	0.9775	0.9558	0.9551	0.9552	0.9850
#2	0.9788	0.9581	0.9576	0.9576	0.9859
#3	0.9763	0.9538	0.9526	0.9528	0.9842
#4	0.9813	0.9630	0.9626	0.9626	0.9875

highest precision for DDOS at 0.9794. Fold#4 stands out with a notably high precision for DDOS at 0.9943. Recall values remain fairly consistent across folds, mostly hovering around the 0.95 mark. The F-measure, which combines precision and recall, also shows commendable results with the highest being 0.9717 for DDOS in Fold#4. Specificity, which indicates the true negative rate, is highest for DDOS in Fold#3 and Fold#4, reaching up to 0.9982. Overall, the metrics indicate a strong performance with slight variances across folds and classes.

The table 4.2 provides an overview of the model's performance across four different folds. Accuracy values, which represent the proportion of correct predictions, consistently exhibit strong performance, ranging from 0.9763 in FOLD#3 to a peak of 0.9813 in FOLD#4. This high accuracy is further corroborated by the macro-level metrics. Macro Precision, which indicates the average precision across classes, demonstrates values above 0.95 for all folds, with FOLD#4 having the highest at 0.963. Similarly, Macro Recall, representing the average recall across classes, fluctuates around the 0.95 mark, with FOLD#4 slightly leading the pack. The Macro F-measure, a harmonic mean of precision and recall, echoes these findings with values very close to Macro Recall. Lastly, the Macro Specificity, which measures the true negative rate, consistently remains high across all folds, highlighting the model's capability to correctly identify negative cases, with the highest specificity seen in FOLD#4 at 0.9875. Overall, the model demonstrates robust and consistent performance across different folds.

LIDM. The confusion matrices presented in figure 4.2 provide insights into the performance of the model trained on the LIDM dataset across four folds of cross-validation. In each fold, the diagonal elements of the matrix represent the number of correct predictions for each class: BENIGN, DDOS, RANSOMWARE, and INJECTION. For instance, in Fold#1, the model correctly classified 5938 instances as BENIGN but misclassified 130 as RANSOMWARE and 175 as INJECTION. Similarly, 5813 DDOS instances were correctly

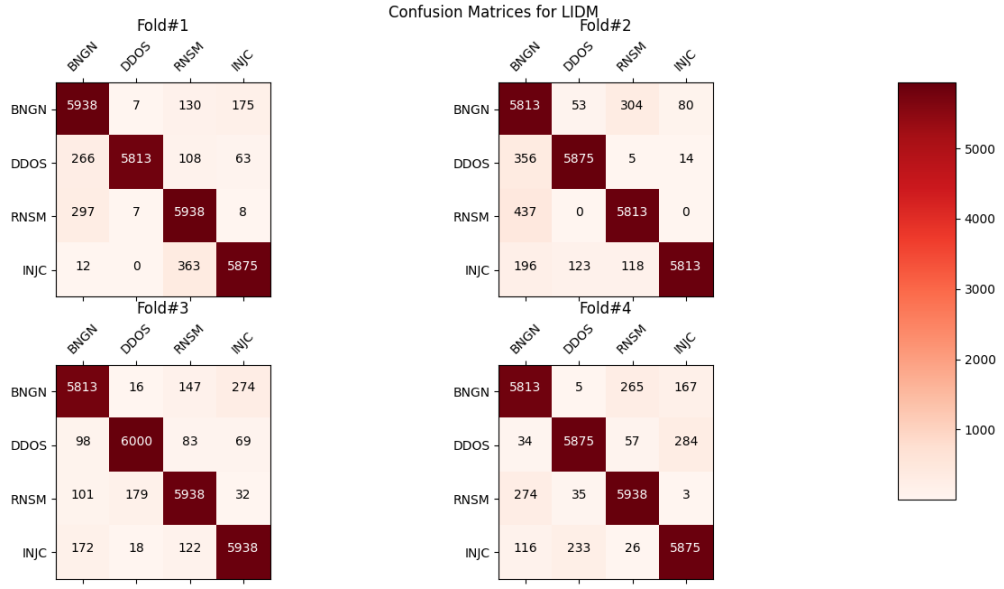

Fig. 4.2: Confusion matrices obtained from 4 fold cross validation performed on LIDM

Table 4.3: Performance Metrics by Fold and Category

	Metrics	BNGN	DDOS	RNSM	INJC
Fold#1	Precision	0.9117	0.9976	0.9081	0.9598
	Recall	0.9501	0.9301	0.9501	0.9400
	F-measure	0.9305	0.9627	0.9286	0.9498
	Specificity	0.9693	0.9993	0.9679	0.9869
Fold#2	Precision	0.8546	0.9709	0.9316	0.9841
	Recall	0.9301	0.9400	0.9301	0.9301
	F-measure	0.8907	0.9552	0.9308	0.9563
	Specificity	0.9473	0.9906	0.9772	0.9950
Fold#3	Precision	0.9400	0.9657	0.9440	0.9406
	Recall	0.9301	0.9600	0.9501	0.9501
	F-measure	0.9350	0.9629	0.9470	0.9453
	Specificity	0.9802	0.9886	0.9812	0.9800
Fold#4	Precision	0.9320	0.9556	0.9446	0.9283
	Recall	0.9301	0.9400	0.9501	0.9400
	F-measure	0.9310	0.9477	0.9474	0.9341
	Specificity	0.9774	0.9854	0.9814	0.9758

identified, but 108 of them were falsely predicted as RANSOMWARE. Each fold displays some variability in misclassifications across classes. For example, Fold#2 shows a noticeable increase in misclassifications of BENIGN as RANSOMWARE compared to Fold#1. On the other hand, in Fold#3 and Fold#4, the number of correctly predicted RANSOMWARE instances is consistent at 5938. Overall, the confusion matrices provide a comprehensive view of the model's strengths and weaknesses across different data splits in the four-fold cross-validation.

In the table 4.3 four-fold cross-validation of the LIDM application, the performance metrics clearly exhibit its robustness in threat detection and classification. For instance, in Fold#1, DDOS detection exhibits an exceptional precision of 0.9976, indicating that 99.76% of the identified DDOS threats were accurate. This

Table 4.4: Accuracy and Macro Metric Values obtained from 4 fold cross validation performed on LIDM

Fold ID	Accuracy	Macro-Precision	Macro-Recall	Macro-F-measure	Macro-Specificity
#1	0.9713	0.944	0.9426	0.9429	0.9809
#2	0.9663	0.935	0.9326	0.9333	0.9775
#3	0.9738	0.948	0.9476	0.9476	0.9825
#4	0.9700	0.940	0.9400	0.9401	0.9800

remarkable precision is echoed in Fold#2 for the INJC category with a value of 0.9841. The recall values, which reflect the system's sensitivity, remain consistent, as evidenced by RNSM's recall of 0.9501 in both Fold#1 and Fold#3. Furthermore, the F-measure, which is a harmonized average of precision and recall, remains commendably high across all threats, such as the 0.9627 for DDOS in Fold#1 and 0.9563 for INJC in Fold#2. Specificity, indicating the system's ability to correctly identify non-threats, is notably high in Fold#1 for DDOS at 0.9993 and for INJC in Fold#2 at 0.995. All these metrics emphasize the reliability and efficacy of the LIDM application in the realm of cyber threat detection and classification.

In the table 4.4 four-fold cross-validation evaluation of the LIDM application, the metrics consistently demonstrate a high level of performance across all folds. The accuracy of the application remains stellar, ranging between 96.63% in Fold#2 and 97.38% in Fold#3. This trend of remarkable proficiency is also reflected in the Macro-Precision, with values spanning from 0.9353 in Fold#2 to 0.9476 in Fold#3. Similarly, the Macro-Recall, indicating the application's sensitivity in correctly identifying threats, consistently hovers around the 94% mark across all folds. The Macro-F-measure, a harmonized average of precision and recall, fortifies this trend by consistently staying above 93%. Lastly, the Macro-Specificity, a measure of the application's aptitude in correctly recognizing non-threats, remains impressive, reaching as high as 98.25% in Fold#3. All these metrics collectively underscore the effectiveness and reliability of the LIDM application in its operational domain.

CEIDM. The confusion matrices presented in figure 4.3 for CEIDM across four folds depict the classification performance for the categories: "BENIGN", "DDOS", "RANSOMWARE", and "INJECTION". Across the folds, the diagonals of the matrices, which represent the true positive classifications, show consistently high values. This indicates a generally high accuracy in classification for each category. However, there are evident misclassifications. For example, in Fold#1, "RANSOMWARE" has been occasionally mistaken as "DDOS", while "INJECTION" has a considerable number of false predictions across other categories. Similarly, in other folds, while the diagonal values remain dominantly high, indicating successful classifications, the off-diagonal values point out areas where the model struggled, resulting in misclassifications. These matrices offer valuable insights for further refinement of the classification model to enhance its accuracy across all categories.

The table 4.5 presents performance metrics of CEIDM, evaluated across four folds. These metrics include Precision, Recall, F-measure, and Specificity for four classes: BNGN, DDOS, RNSM, and INJC. For Fold#1, the model exhibits high precision for the BNGN class at 0.9747, while the DDOS class has the lowest precision of 0.8971. However, the recall for DDOS is higher than BNGN in this fold. This trend of varying precision and recall figures is observed across all folds. The F-measure, which combines precision and recall, also demonstrates varied performance among classes, with values mostly hovering above 0.9. Specificity, indicating the true negative rate, remains notably high across all classes and folds, often surpassing 0.97. By Fold#4, the model's precision for the RNSM class peaks at 0.9547. In order to sum up, the CEIDM model exhibits good performance for all metrics and classes, though there is some variation between the folds.

The table 4.6 presents an overview of the aggregate performance metrics for the CEIDM model over four folds. The metrics encapsulated include Accuracy, Macro-Precision, Macro-Recall, Macro-F-measure, and Macro-Specificity. Throughout all folds, the model consistently maintains an accuracy around 0.96, indicating a strong overall classification performance. The Macro-Precision and Macro-Recall, which provide an average performance measure across the classes, mostly hover in the low 0.93s to high 0.92s range, denoting that the model is both precise and sensitive in its predictions. The Macro-F-measure, which combines the precision and recall, mirrors these figures closely. Notably, the Macro-Specificity remains robust across all folds, predominantly exceeding 0.97, suggesting that the model is adept at correctly identifying negatives. In essence, CEIDM

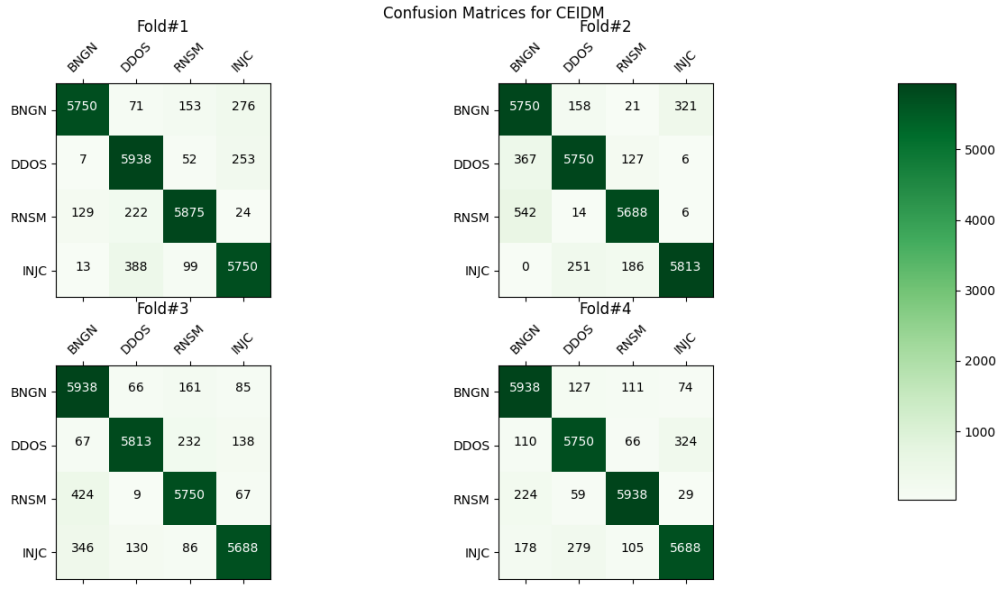

Fig. 4.3: Confusion matrices obtained from 4 fold cross validation performed on CEIDM.

Table 4.5: Performance Metrics by Fold and MetricPerformance Metric Values obtained from 4 fold cross validation performed on CEIDM

Fold's	Metrics	BNGN	DDOS	RNSM	INJC
Fold#1	Precision	0.9747	0.8971	0.9508	0.9123
	Recall	0.9200	0.9501	0.9400	0.9200
	F-measure	0.9466	0.9228	0.9454	0.9161
	Specificity	0.9921	0.9637	0.9838	0.9705
Fold#2	Precision	0.8635	0.9315	0.9445	0.9458
	Recall	0.9200	0.9200	0.9101	0.9301
	F-measure	0.8909	0.9257	0.9270	0.9379
	Specificity	0.9515	0.9774	0.9822	0.9822
Fold#3	Precision	0.8765	0.9659	0.9231	0.9515
	Recall	0.9501	0.9301	0.9200	0.9101
	F-measure	0.9118	0.9477	0.9215	0.9303
	Specificity	0.9554	0.9891	0.9745	0.9845
Fold#4	Precision	0.9206	0.9252	0.9547	0.9302
	Recall	0.9501	0.9200	0.9501	0.9101
	F-measure	0.9351	0.9226	0.9524	0.9200
	Specificity	0.9727	0.9752	0.9850	0.9772

Table 4.6: Accuracy and Macro Metric Values obtained from 4 fold cross validation performed on CEIDM

Fold ID	Accuracy	Macro-Precision	Macro-Recall	Macro-F-measure	Macro-Specificity
#1	0.9663	0.934	0.9325	0.9327	0.9775
#2	0.9600	0.921	0.9200	0.9204	0.9733
#3	0.9638	0.929	0.9276	0.9278	0.9759
#4	0.9663	0.933	0.9326	0.9325	0.9775

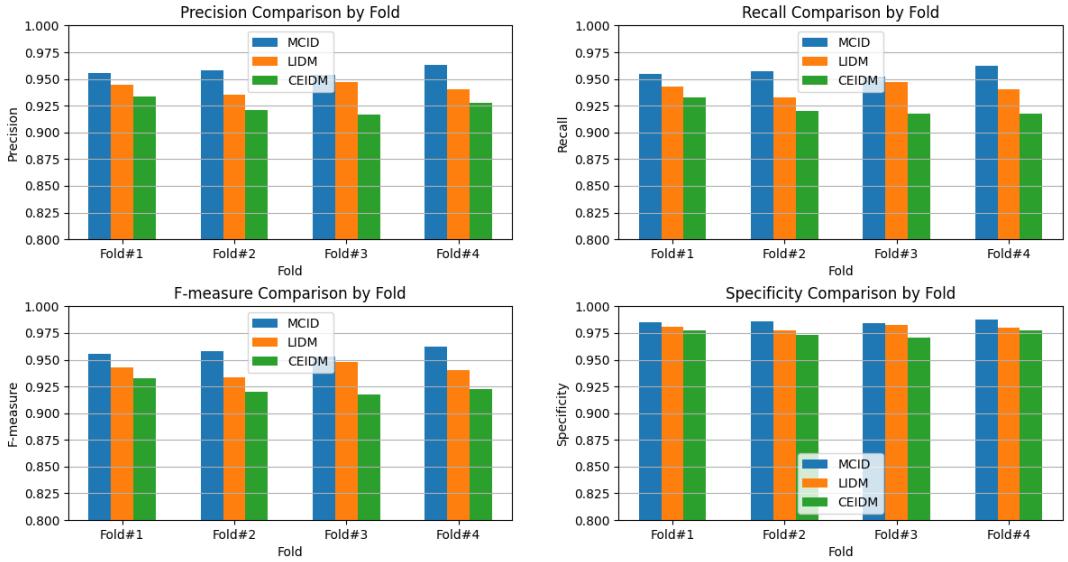

Fig. 4.4: Comparative analysis of precision, recall, f-measure, and specificity

consistently delivers reliable and robust performance metrics across the four evaluation folds.

4.2. Comparative Analysis. This section provides evaluation comparing the proposed MCID model with its contemporaries, LIDM and CEIDM. Through a meticulous examination of performance across vital metrics such as precision, recall, F-measure, specificity, and accuracy, along with their macro counterparts, insights are drawn regarding the inherent strengths, potential pitfalls, and distinguishing characteristics of each model. This analysis seeks to offer a comprehensive perspective on the standing of MCID within the contemporary landscape.

According to comparative analysis presented in figure 4.4, MCID consistently demonstrates commendable performance across all metrics and folds, indicating its robustness and generalizability. Notably, it exhibits high specificity, especially in the initial folds, ensuring a low false positive rate which is crucial in many applications. Additionally, its balanced results across precision, recall, and F-measure highlight its capability to provide reliable classification without significant trade-offs. Even in scenarios where it doesn't lead, MCID remains a close competitor, showcasing its potential as an effective and versatile method in comparison to LIDM and CEIDM.

The bar graphs in figure 4.5 provide insight into the performance of three methods (MCID, LIDM, and CEIDM) across four different folds. At a glance, MCID consistently offers the highest accuracy across all folds, with FOLD#4 being its best at 0.9813. While LIDM and CEIDM often hover close in metrics, LIDM slightly outperforms CEIDM in accuracy for most folds. However, in terms of Macro Precision, Recall, and F-measure, MCID again generally surpasses the other two, solidifying its position as the most effective model among the three, especially in FOLD#4. For Macro Specificity, all methods perform quite commendably, with minimal variance across folds. Yet, MCID again edges out with the highest value in FOLD#4. Overall, the MCID method proves superior, with FOLD#4 consistently representing the highest metric values for this method.

A comparative analysis table 4.7 summarizes the average performance improvement of MCID over LIDM and CEIDM with respect to four key metrics. The list presents percentages (showing improvements) along with their standard deviations. Compared with LIDM and CEIDM, the model is very reliable in terms of detailing every step along the way. Every time it was tested for its ability to forecast results, on average a 0.8425 % improvement occurred when compared with LIDM and an even larger increase of 1.4975 % over that achieved by CEIDM. Furthermore, in Macro Precision it is better than LIDM by 1.685 % and CEIDM by

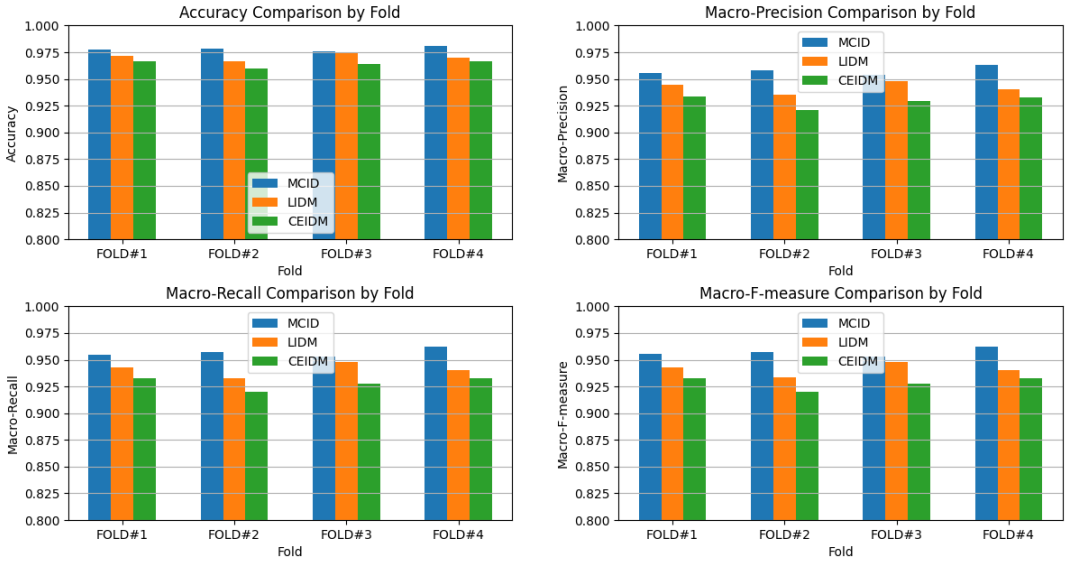

Fig. 4.5: Comparative analysis of Accuracy and macro values of the precision, recall, and f-measure

Table 4.7: comparative table of improvement rate

Metric	Improvement vs LIDM (%)	Improvement vs CEIDM (%)
Accuracy	0.8425	1.4975
Macro Precision	1.685	3.08
Macro Recall	1.735	3.1125
Macro F-measure	1.71	3.1
Macro Specificity	0.55	0.975

3.08 %, while in terms of Macro Recall the latter figure comes to as much as 1.735 % over against those for respectively approximately definitive idiomatic expressions (LIDM) and common sense fine-grained Macro F-measure's general performance Is MCID 1.71% ahead and 3.1 % higher over the two other models respectively, which suggests that this is a balanced improvement in both precision and sensitivity of classifications at its peak value for each parameter (F max). For Macro Specificity, MCID retains a small edge over LIDM (240.86P compared to 239.57 P) and an even greater margin over CEIDM at 1 point of distortion error or more—a whopping improvement in both cases! These figures together attest to MCID's strength and its prowess at helping unleash the full potential of intrusion detection in IoT fog computing environments.

5. Conclusion. In sum, the MCID framework represents an important step forward in IoT fog computing security. It fills the critical gaps that traditional security measures are often unprepared to fill, dealing with fog computing environments which by their nature tend to be complex and constantly evolving. With a rich set of features and sophisticated machine learning techniques, MCID is offered as an efficient solution to the problems of intrusion detection. By its detailed architecture, the system can fully exploit behavioral, temporal and anomaly characteristics, optimizing them through SVM-BFE. This leads to a streamlined feature set that improves the system's accuracy and speed. Also, using a Random Forest algorithm for multilabel classification helps MCID precisely identify various kinds of intrusion targets. Cross-validation results further show that the MCID framework has high efficacy across many different metrics, making it a potent weapon in comprehensive security. Based on the precision, recall rate. F-measure and specificity values it achieves accuracy in threat detection and classification while minimizing false positives to grant IoT devices a high level of reliable protection

under fog computing networks. In short, the design of MCID is an important step in making fog computing secure. Its novel methodology and early successes open the gateway for more research and development on this front, foreshadowing a model that is scalable as well as flexible enough to conform with changes in the threat environment created by IoT.

REFERENCES

[1] M. Lombardi, F. Pascale, and D. Santaniello, *Internet of things: A general overview between architectures, protocols and applications*, Information, 12(2), 2021, pp. 87.

[2] S. Qabil, U. Waheed, S. M. Awan, Y. Mansoor, and M. A. Khan, *A survey on emerging integration of cloud computing and internet of things*, In 2019 International Conference on Information Science and Communication Technology (ICISCT), IEEE, 2019, pp. 1-7.

[3] A. Ometov, O. L. Molua, M. Komarov, and J. Nurmi, *A survey of security in cloud, edge, and fog computing*, Sensors, 22(3), 2022, pp. 927.

[4] B. Sudqi Khater, A. W. B. A. Wahab, M. Y. I. B. Idris, M. A. Hussain, and A. A. Ibrahim, *A lightweight perceptron-based intrusion detection system for fog computing*, Applied Sciences, 9(1), 2019, pp. 178.

[5] K. Sadaf and J. Sultana, *Intrusion detection based on autoencoder and isolation forest in fog computing*, IEEE Access, 8, 2020, pp. 167059-167068.

[6] P. Kumar, G. P. Gupta, and R. Tripathi, *A distributed ensemble design based intrusion detection system using fog computing to protect the internet of things networks*, Journal of Ambient Intelligence and Humanized Computing, 12, 2021, pp. 9555-9572.

[7] K. Kalaivani and M. Chinnadurai, *A Hybrid Deep Learning Intrusion Detection Model for Fog Computing Environment*, Intelligent Automation & Soft Computing, 30(1), 2021.

[8] F. A. Zwayed, M. Anbar, Y. Sanjalawe, and S. Manickam, *Intrusion Detection Systems in Fog Computing–A Review*, In Advances in Cyber Security: Third International Conference, ACeS 2021, Penang, Malaysia, August 24–25, 2021, Revised Selected Papers 3, Springer Singapore, 2021, pp. 481-504.

[9] H. A. Afolabi and A. Aburas, *Proposed back propagation deep neural network for intrusion detection in internet of things fog computing*, Int J, 9(4), 2021, pp. 464-469.

[10] M. Ramkumar, *Support vector machine based intrusion detection system in fog computing*, ICTACT Journal on Data Science and Machine Learning, March 2021, Volume: 02, Issue: 02, pp. 16-164.

[11] V. Chang, L. Golightly, P. Modesti, Q. A. Xu, L. M. T. Doan, K. Hall, S. Boddu, and A. Kobusińska, *A survey on intrusion detection systems for fog and cloud computing*, Future Internet, 14(3), 2022, pp. 89.

[12] A. Wu, S. Tu, M. Wagas, Y. Yang, Y. Zhang, and X. Bai, *Intrusion Detection System Using a Distributed Ensemble Design Based Convolutional Neural Network in Fog Computing*, Journal of Information Hiding and Privacy Protection, 4(1), 2022, pp. 25.

[13] O. A. Alzubi, J. A. Alzubi, M. Alazab, A. Alrabea, A. Awajan, and I. Qiqieh, *Optimized machine learning-based intrusion detection system for fog and edge computing environment*, Electronics, 11(19), 2022, pp. 3007.

[14] J. Sajid, K. Hayawi, A. W. Malik, Z. Anwar, and Z. Trabelsi, *A Fog Computing Framework for Intrusion Detection of Energy-Based Attacks on UAV-Assisted Smart Farming*, Applied Sciences, 13(6), 2023, pp. 3857.

[15] K. Kaliyaperumal, C. Murugaiyan, D. Perumal, G. Jayaraman, and K. Samikannu, *Combined Ensemble Intrusion Detection Model using Deep learning with Feature Selection for Fog Computing Environments*, Acta Scientiarum. Technology, 45, 2023.

[16] D. Mohamed, and O. Ismael, *Enhancement of an IoT hybrid intrusion detection system based on fog-to-cloud computing*, Journal of Cloud Computing, 12(1), 2023, pp. 1-13.

[17] C. A. de Souza, C. B. Westphall, and R. B. Machado, *Intrusion detection with Machine Learning in Internet of Things and Fog Computing: problems, solutions and research*, Sociedade Brasileira de Computação, 2023.

[18] G. Zhao, Y. Wang, and J. Wang, *Lightweight Intrusion Detection Model of the Internet of Things with Hybrid Cloud-Fog Computing*, Security and Communication Networks, 2023, 2023.

[19] D. A. Pisner, and D. M. Schnyer, *Support vector machine*, In Machine learning, Academic Press, 2020, pp. 101-121.

[20] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, *Survey of intrusion detection systems: techniques, datasets and challenges*, Cybersecurity, 2(1), 2019, pp. 1-22.

[21] S. J. Rigatti, *Random forest*, Journal of Insurance Medicine, 47(1), 2017, pp. 31-39.

[22] L. Rokach and O. Maimon, *Decision trees*, Data mining and knowledge discovery handbook, 2005, pp. 165-192.

[23] R. Vishwakarma and A. K. Jain, *A survey of DDoS attacking techniques and defence mechanisms in the IoT network*, Telecommunication systems, 73(1), 2020, pp. 3-25.

[24] P. O'Kane, S. Sezer, and D. Carlin, *Evolution of ransomware*, IET Networks, 7(5), 2018, pp. 321-327.

[25] R. D. Jager, L. P. Aiello, S. C. Patel, and E. T. Cunningham Jr., *Risks of intravitreous injection: a comprehensive review*, Retina, 24(5), 2004, pp. 676-698.

[26] Y. Li, Y. Jiang, Z. Li, and S.-T. Xia, *Backdoor learning: A survey*, IEEE Transactions on Neural Networks and Learning Systems, 2022.

[27] H. F. Atlam, R. J. Walters, and G. B. Wills, *Fog computing and the internet of things: A review*, Big Data and Cognitive Computing, 2(2), 2018, pp. 10.

[28] Y.-Y. Song and L. U. Ying, *Decision tree methods: applications for classification and prediction*, Shanghai Archives of

Psychiatry, 27(2), 2015, pp. 130.
[29] T. Yu and H. Zhu, *Hyper-parameter optimization: A review of algorithms and applications*, arXiv preprint arXiv:2003.05689, 2020.
[30] N. Moustafa and J. Slay, *UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)*, In 2015 Military Communications and Information Systems Conference (MilCIS), Nov 2015, pp. 1–6.

Edited by: Anil Kumar Budati
Special issue on: Soft Computing and Artificial Intelligence for wire/wireless Human-Machine Interface
Received: Oct 4, 2023
Accepted: Jan 13, 2024