International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

FALC-IDS: Securing EVCS through Network IDS Using EDTWFS and Federated Learning from Various EV Attacks

***Dr. A. Bamini[1],V. Arun Raj[2]**

Submitted:11/03/2024 **Revised**: 26/04/2024 **Accepted**: 03/05/2024

Abstract: In united states around 2.5 million electric vehicles are operating nowadays. They required the Electric vehicle charging stations (EVCS). EVCS has personal and payment information, and it run different protocols than traditional firewalls. To protect against cyber-attacks, a new Intrusion Detection Model is required. However, new vulnerabilities are still found. This work proposed a novel Federated Averaging Learning Classifier (FALC) for intrusion detection in EV charging stations (EVCS). This work uses FALC, it allows various clients to train models without sharing data. The FALC-IDS used CIC EV Charger Attack Dataset 2024 (CICEVSE2024). It contains network traffic, hardware performance, and kernel event data. The dataset is pre-processed to remove unwanted content, such as missing /duplicate values and unwanted columns. An Enhanced Dynamic Threshold Whale Optimization based Feature Selection (EDTWFS) is used to choose best features (SF) from the dataset. Also, Set of Novel features (NF) are created using statistical methods. The FALC model is compared with CNN, SVM and ANN. Experimental results show that the FALC model outperforms the other models in terms of accuracy and F1-score. On the SF dataset, it has an accuracy of 99.98% and an F1-score of 99.77%. On the NF dataset, it has an accuracy of 99.86% and an F1-score of 99.73%.

Keywords: *Electric Vehicle Charging Stations (EVCS); Network Intrusion Detection; Federated Learning; Cybersecurity; CIC EV Charger Attack Dataset, SVM, RF, ANN, CNN, EDTWFS, FALC.*

1. Introduction

The rapid growth of electric vehicles (EVs) was noticed at the 27th Conference of Parties in Egypt in November 2022 [1]. This growth shows the need for a safe and easy-to-use system for charging EVs. Internet-connected electric vehicle charging stations (EVCS) make it easier for people to identifies the stations and charge their cars and also add extra services to the electric vehicle charging station system (EVCSS). However, as more EVCS are made, both physical attacks and cyberattacks have increased. A physical attack is when someone tries to hurt the electric vehicle charging station (EVCS) by damaging it or changing it. This could start a fire or make other safety problems. Another type of attack is Network-Intrusion attack. Someone gets into the EVCS network without permission and messes with it. This could include the attacker communication between the charging station and the electric vehicle. The attacker overwhelms the network to prevent authorized users from accessing the charging station. While physical attacks are not as bad as cyberattacks, they can still cause problems.

Due to security concerns, having a special model for finding intrusions in EVCS's is necessary. It is important to have strong security measures to stop attacks that could cause outages or security risks. Electric vehicle charging stations (EVCS) are very important parts of our critical energy infrastructure.

They are needed to help us move to more sustainable ways of transportation. Many modern EVCSs are connected to the internet and other communication networks, making them vulnerable to various online threats. Intrusion detection systems (IDS) [19,20] are important for finding and stopping different kinds of threats. IDS tries to solve this problem by classifying different kinds of malicious activity and using multi-class classification to tell the difference between normal and harmful data streams. Renewable energy can now be used to charge electric vehicles (EVs). The technology called vehicle-to-grid (V2G) lets EVs also send energy back to the power grid. Now a day many peoples use EVs, there are more risks to the online network. To protect multiple (EVCS) from attacks, with the help of Intrusion Detection System (IDS). The collection of information about users also raises concerns about privacy. This need of IDS is shown in Figure 1.

[1] *Head & Assistant Professor, Department of computer applications, The Standard Fireworks Rajaratnam College for Women, Sivakasi, Tamilnadu, India. ORCID: 0000-0001-7559-6943*
** Corresponding Author Email: drbaminia@gmail.com,*
[2]*Assistant Professor (Sr. Grade), Electronics and Communication Engineering, Mepco Schlenk Engineering College, Sivakasi, Tamilnadu, India. arunraj@mepcoeng.ac.in ORCID: 0000-0003-0950-809X*

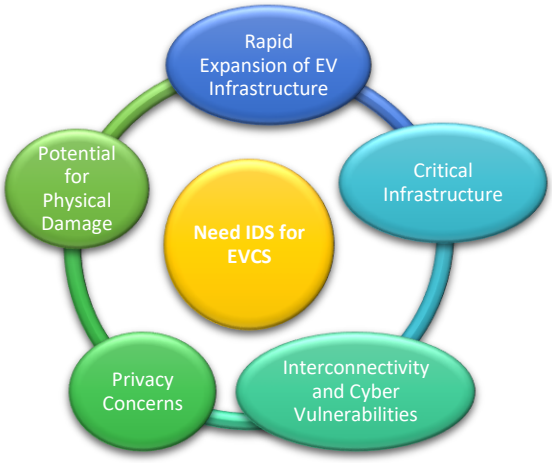

Fig 1: Need IDS for EVCS

The vulnerabilities can be handled by an IDS model created specifically for EV charging stations. This proposed model monitors logs and network traffic to detect cyber threats and stop them before they harm clients. Also, the federated learning strategy lets different clients train the system together while keeping their data private. This helps solve the problem of exposing sensitive user information. With the highest level of security and privacy for EV charging ecosystems, this method ensures that the intrusion detection system can adapt to new threats as they emerge.

1.1 Research Contribution

The primary aim of the proposed work is to build a Federated Averaging Learning Classifier (FALC) for detecting intrusions in EVCS. This FALC model uses federated learning, which keeps privacy, it allowing clients to train the model without sharing data. As the EVCS growing continues, there is a need for reliable and private IDS to prevent it from attack. This proposed approach meets that requirement for preventing IDS. Here is a brief summary of the key research contributions:

- FALC, a state-of-the-art federated learning-based intrusion detection model specifically designed for the EVCS environment.
- extensive feature engineering and data preprocessing methods, such as the creation of novel features (NF) and the application of a whale optimization algorithm to select relevant selected features (SF).
- thorough evaluation and comparison of the FALC model on SF and NF datasets with other well-known machine learning models, including SVM, RF, ANN and CNN.
- The FALC model outperforms other models on the CICEVSE2024 dataset and shows superior performance in terms of accuracy, precision, recall and F1-score.
- the use of federated learning approaches to address the critical need for privacy-friendly intrusion detection in EVCS ecosystems.

1.2 Motivation & Problem Statement

The rapid growth of electric vehicles and their charging stations has led to new cyber security challenges. These charging stations can be attacked by cyber criminals and malicious tampering. Such attacks could put users' privacy at risk and disrupt the power grid. They could also make it difficult for people to charge their vehicles. As cyber-attacks are constantly evolving, traditional centralized systems for detecting them might not work as effectively anymore. Additionally, centralized systems that collect & process data from multiple charging stations. It makes user data more vulnerable to being stolen or accessed by unauthorized individuals. This raises concerns about privacy. To address these issues, this work has a new system called FALC-IDS. This system helps protect sensitive data while still detecting cyberattacks. FALC-IDS allows multiple charging stations (called clients) to work together to train the system. In doing so, each charging station only sends a small amount of data to the others, keeping things private. At the same time, the FALC-IDS system protects each charging station's data by using a method called federated learning. This method keeps all information secure and does not move it between stations. This way, even if one charging station is attacked, the others are still safe and can keep helping to protect the whole system.

1.2.1 Problem Statement

Let X = {x1, x2, ..., xn} is an input feature vectors, where xi ∈ Rm represents an instance of network traffic and kernel events. Let Y = {y1, y2, ..., yn} is a set of labels, where yi ∈ {0, 1, 2, ..., C} with 0 denoting normal (benign) behavior and 1, 2, ..., C representing types intrusions. The goal is to learn a function f: Rm → {0, 1, 2, ..., C} that can accurately map input feature vectors X to their corresponding class labels Y, while preserving data privacy and maintaining high predictive performance. The objective is to determine the ideal values of θ for the function f(x; θ) in order to minimize the loss function θ* = argmin L(f(X; θ), Y), where L(·) represents an appropriate loss function (e.g. G. sensitive data is kept safe and locally stored by performing the minimization within the limitations set by federated learning (e.g., cross-entropy loss for multi-class classification). Furthermore, maximizing the metrics Accuracy, Precision, Recall, and F1-Score, the learned function f(x; θ*) should demonstrate strong predictive performance. Create an intrusion detection model with a high level of accuracy, precision, recall and F1 score that can distinguish between malicious and legitimate activity, including different types of cyberattacks. If a charging station has a virus, it can spread the virus to other vehicles that use it. This can harm those vehicles and make them stop working. This can be

dangerous for drivers, passengers, people nearby, and property. To do this, use a comprehensive data set that includes network traffic, hardware performance counters and kernel events from EVCS environments.

1.3 Paper Organization

The paper is organized as follows: The 2nd section Review of literature provides an analysis of the literature on intrusion detection in EVCS. In the 3rd section, covers the data acquisition, pre-processing, novel feature generation, feature selection and classification. The 4th section describes the experimental setup, including configurations, datasets, and evaluation metrics. It compares and evaluates the performance of the FALC model with other models. The 5th section, Comparative Analysis, compares the proposed FALC model with existing literature. Finally, the 6th section, Conclusions, presents the main findings, contributions, and implications of the study.

2. REVIEW LITERATURE

Using a variety of machine learning techniques and native IoT dataset, ElKashlan, M. [1] addresses IoT security issues in EVCS by detecting fraudulent traffic in EVCS, a task that has never been done before. To detect Distributed Denial of Service (DDoS) attacks in the EVCS network environment, he compares different machine learning classification algorithms. The model proposed by Zhaoliang Lin et al [2] uses autoencoder stacking to build a layered unsupervised representation learning algorithm. Batch normalization of the hidden layers accelerates the convergence of the model to prevent overfitting and local optima. It also adds an attention mechanism to highlight important aspects of sequences consisting of data vectors to increase accuracy rates. EVCP is allowed to keep local data for model training and submit the model parameters to the central server for building new global models to reduce the possibility of user privacy leaks on the central server. In this paper [3], IoT vulnerabilities are predicted, along with their causes and methods for detection. Using the case study SILEX, a known malware targeting IoT devices, we show what lessons can be learned.

The Collaborative Anomaly Detection System for Charging Stations, or CADS4CS, was proposed by Cumplido et al [4] as an optimization measure. Using a tuning algorithm, the central analysis unit of CADS4CS combines several different anomaly detection systems to achieve higher accuracy. In addition, CADS4CS offers the possibility to regularly re-train machine learning models to ensure that they are adapted to the data distribution. Various use cases and real-world studies are being investigated to validate the methodology and demonstrate the effectiveness and efficiency of the solution.

The study by Samrat Acharya et al [5] deals with demand-side cyberattacks on power grids using high-power EV charging stations (EVCS) connected to the Internet. In these attacks, the voltages and frequencies of the power grid are distorted. The detailed charging behavior of an electric vehicle is considered by Dustin Kern et al [6], and we analyze and discuss various design options. We consider models that combine both novelty- and classification-based anomaly detection approaches as well as an ensemble approach. We perform evaluations by using simulated attacks and data from actual EV charging sessions. Our results demonstrate that for attacks that affect individual reports during a charging session, regression-based prediction provides a significant improvement in detection performance.

An analysis of the cybersecurity risks of the EVCS network is presented by Hamdare S. et al [7]. First, the background of the research area is presented based on recent developments in the EVCS network, recent trends in electric vehicle customization, and electric charging use cases. Second, potential cyber-attack scenarios and infrastructure and protocol-related vulnerabilities have been considered in presenting the cybersecurity aspects of EVCS. Third, real-time data-centric analysis of EV charging sessions has confirmed the threats in EVCS. ElKashlan proposes [8] a classification algorithm that utilises machine learning to identify malicious traffic in an Internet-of-Things environment. The proposed system uses a real IoT dataset derived from actual IoT traffic.

Bidirectional communication, automation, remote control, distributed and embedded intelligence, and intelligent resource management are enabled in power grids by integrating the open communication layer with the physical layer, according to M. Basnet and M. Hasan Ali [9]. Confidentiality, Integrity, and Availability (CIA) of network resources can be jeopardized by cybersecurity risks inherent in the open communication layer. Due to the increasing popularity and use of electric vehicles (EVs), a reliable electric vehicle charging station (EVCS) infrastructure must be widely deployed. The DNN- and LSTM-based IDS both achieved a detection accuracy of more than 99 %, according to the results.

The researchers Vinod Miskin et al. [10] suggest using deep learning and machine learning techniques to create an intrusion detection system (IDS). They tested a variety of models with CICIDS 2017 dataset, including Naive Bayes, SVM, Long Short-Term Memory (LSTM) networks & DNN. Balakrishna et al [11] suggest that the best way to detect fraud in an IoT network is by using a ML approach for classification. This work uses real-time data from actual IoT traffic. Implementing ML in the IDS engine of the EVCS can help to stop attacks. An intrusion detection system (IDS) is explained by N. Singh and R. Agarwal [12]. It finds hostile attacks that target EVs specifically. The goal

of this work is to detect and prevent potential attacks. The network traffic between different EV parts is managed by it.

2.1 Research Gap & Challenge's

A study from the review literature, some challenges are captured as follows

1 **Detecting Distributed Denial of Service (DDoS)** attacks: Several studies [1], [3], [9] discuss the significance of identifying DDoS attacks aimed at Electric Vehicle Charging Station (EVCS) networks. These attack can disrupt charging services and instability in the power grid.

2 **Privacy concerns**: Zhaoliang Lin et.al [2] warn that sharing private information with a central server for training machine learning models can be risky. They explain that it's important to find a way to protect users' privacy while still letting people work together to find unusual things.

3 **Predicting anomalies during Charging**: Dustin et.al [6] emphasized the importance of looking at charging behavior during a single session and finding the best methods for detecting anomalies. They compared different types of models, such as classification-based ones.

4 **Infrastructure and protocol-centric vulnerabilities**: Hamdare. S et.al. [7] present cybersecurity risks in the EVCS network, considering infrastructure and protocol-centric vulnerabilities, and the need for real-time data-centric analysis of EV charging sessions.

5 **Evaluating and comparing machine learning algorithms**: Several studies [1], [3], [8], [10], [11] compare different machine learning algorithms, such as Support Vector Machines, Naive Bayes, Deep Neural Networks, and Long Short-Term Memory networks, for detecting intrusions and cyberattacks in the EVCS environment.

Previous studies have mainly focused on machine learning methods, but these have not been able to adequately address the evolution and propagation of new network threats. Therefore, advanced intrusion detection techniques are urgently needed to ensure the security and integrity of the EVCS ecosystem. Common difficulties include identifying different forms of cyberattacks, dealing with privacy issues, evaluating and selecting appropriate machine learning models, and developing powerful intrusion detection systems specifically designed for EVCS and EV environments, taking into account the unique characteristics and vulnerabilities of these systems.

3. PROPOSED MODEL

Intrusion Detection System's (IDS) federated learning-based workflow illustrates in Figure 2. Gathering the data sets is the process's initial step, i.e. e. to gather data from various sources (EV1, EV2, etc.) regarding normal and attack data, kernel and HPC (High-Performance Computing) data, and network traffic. which could stand for various networks or environments. The pre-processing step of data processing involves encoding the data, eliminating superfluous columns, and eliminating duplicate and missing values after it has been captured. This stage of data cleansing is known as "data cleansing.".

The processed data is fed into an EDTWFS algorithm (*Enhanced Dynamic Threshold Whale Optimization based Feature Selection*) in the following stage of feature processing. This algorithm runs two components: a "Novel Feature Generator" and a "Best Feature Selection." Next, the data set is split into test and training subsets. The training dataset is used to train multiple EV clients through a federated learning process, in which the models cooperate and share knowledge without directly exchanging data. To make judgments and classify various attack types (Attack1, Attack2, Attack3, Attack4, .. AttackN), federated learning models are employed. An alert system that alerts users to detected network intrusions or attacks is the end result.

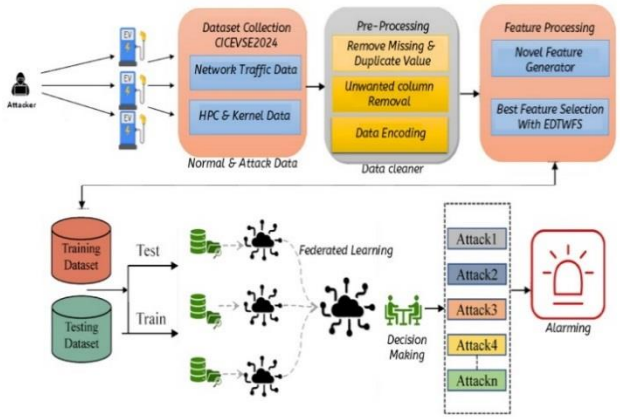

Fig 2: Proposed work Flow

3.1 Data Acquisition

This work is based on the CIC EV Charger Attack Dataset 2024 (CICEVSE2024). It has 8,474 records and 915 futures, totaling 26.3 MB. The testbed uses an operational Level charging station called EVSE-A, along with various communication devices. To create the dataset, several components were involved: The Electric Vehicle Communication Controller (EVCC), another EVSE called EVSE-B, a Power Monitor, and a Local Charging Station Monitoring System (CSMS).

3.2 Data Cleaner

An essential part for analyzing and modeling data is to clean it. The purpose of cleaning data is to make it more accurate, and better quality. Several different techniques are used to find and fix any missing information, remove duplicates, remove columns that aren't important, and

encode the data. This makes sure that outcomes from the data are reliable.

3.2.1 Missing Value Removal

Missing values can make a dataset less reliable and accurate. When a dataset has missing values, cleaning it before using it is very important. When dealing with missing values in a dataset, think about these things:

- **Data Collection Issues**: Errors or problems that occur during data collection, such as malfunctioning sensors, network outages, or human error in manual data entry, can result in missing values.
- **Inherent Data Characteristics**: The data may naturally contain missing values in some domains. As an illustration, not all patients may have results from all tests or measurements in healthcare datasets.
- **Data Privacy and Anonymization**: Sometimes, in order to preserve personal information or anonymize sensitive data, missing values are purposefully added.

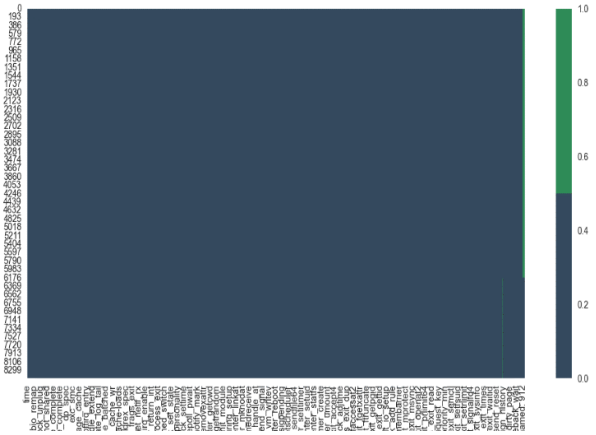

Fig 3: Missing Values

Figure 3 displays the value that is missing. 31602 missing cells (0.04 % of the total cells) were the result of the preprocessing stage removing the entire row containing a missing cell.

3.2.2 Duplication Removal

Datasets may contain duplicate data for various reasons, such as incorrect data entry, merging data from multiple sources, or features present in the original data. If replication is not done correctly, not all duplicates may be found or removed. This can be due to a variety of factors, such as incorrect assumptions, errors, inefficient algorithms that cannot handle large datasets, or certain extreme cases. Even if the %age of duplicate rows in a dataset is small (less than 1%), it is desirable to remove duplicates to maintain data consistency and integrity.

3.2.3 Unwanted Column Removal

Removing columns that contain only zero values is an important step in a Data Cleaner, as these columns do not contribute any valuable information to the analysis or modeling process. Having zeroes in some columns of a dataset can make the dimensions bigger and make the information repeat itself, which can make models prediction as worse.

3.2.4 Data Encoding

Data encoding is one of the phase in the Data Cleaner phase, particularly when dealing with categorical variables which is not suitable for AI models. The feature 'Status' has two distinct categories 'Charging' and 'Idle'. Using a one hot encoding for each category can be assigned a numerical value (e.g., 0 or 1).

Table 1: Status

Charging	1
Idle	0

There are four categories in the "Scenario" field, such as "Benign," "Crypto Jack," "DoS," and "Recon." The one-hot encoding is a simple way to turn a multi-class categorical variable into multiple binary columns. This is useful for making it easier to understand and work with the data.

Table 2: Scenario

Benign	0
Crypto Jack	1
DoS	2
Recon	3

The 'Attack Status' variables are 'Attack' and 'Normal'. Using the label encoding for this binary categorical variable to convert into numeric one.

Table 3: Attack Status

Attack	*1*
Normal	0

After pre-processing, the dataset was cleaned and had 8465 rows, 154 columns, and a size of 8.16 MB. This well-prepared dataset is now ready for use in modelling or further analysis.

Input: Raw dataset D

Output: Cleaned dataset D'

Algorithm: Data Cleaning

1. Find columns (c) with missing values (MV) in D
2. For each c with MV:
    a. If the MV is available:
        i. Drop the c from D
3. Find and delete duplicate rows from D
4. Find columns with all zero values

5. Delete the columns from D
6. Identify categorical columns in D
    a. For each categorical column c:
        i. If c has two categories:
            1. Apply label encoding (e.g., 0 and 1)
        ii. Else:
            1. Apply one-hot encoding to create binary columns for each category
7. Create a new dataset D' with the cleaned and processed data
8. Return Cleaned Dataset D'

End Algorithm

Algorithm 1: Data Cleaning

3.3 Novel Feature (NF) Generator

In the world of AI prediction, making new features from old dataset is very important. There are many reasons why this is useful. These reasons, along with the many ways these new features can be used, show why they are so important.

1. **Capturing non-linear relationships**: The features in the given data set have relationships that are not simple. It can be difficult for models that use only linear to accurately predict complex patterns. To handle these complexities, artificial intelligence (AI) models must nee new features. Novel feature generation makes a new feature from the old ones, which helps them predict better when there are non-linear relationships between the features.
2. **Enhancing feature relevance**: A dataset's original features may not be the best for a certain task. It needs to find more relevant information in the data and make novel features from that. This can help to improve how well a model works and make it easier to understand.
3. **Dimensionality reduction**: Reducing the features of voluminous data is helps for better understanding. It stops the overfitting of the models and makes it more understandable

The complex relationship and the patterns in the EVCS system attack data are not thoroughly described by the original feature set. Making new features can help show hidden patterns that could be useful for finding out. A brief explanation of every novel feature is provided below:

A) Ratio Features:

The computation of ratios among various features is explained as following:

i) Branch Miss Ratio:

This feature keeps track of the ratio of the number of branch instructions that are carried out to the number of incorrect predictions. Performance is significantly impacted by the effectiveness of the branch prediction mechanism, which is disclosed by this information.

$$branch\_miss\_ratio = branch - misses \,/\, branch - loads$$

ii) Bus Access Read Ratio

This metrics is calculated using the busaccess_rd divided by the bus_access which is help to acces bus access read ratio.

$$bus\_access\_read\_ratio = bus\_access\_rd \,/\, bus\_access$$

iii) Cache Miss Ratio

This feature measures how well the cache subsystem responds to memory requests by capturing the ratio of cache misses to total cache references.

$$cache\_miss\_ratio = cache - misses \,/\, cache - references$$

iv) DTLB Miss Ratio

This ratio calculates the total number of retired or speculated instructions divided by the total number of misses in the load and store operations data Translation Lookaside Buffer (TLB). Because the page table must be walked, TLB misses can result in severe performance penalties.

$$dtlb\_miss\_ratio = (dTLB - load - misses + dTLB - store - misses) \,/\, (inst\_retired + inst\_spec)$$

v) ITLB Miss Ratio

This feature measures the proportion of instruction TLB misses to the total number of retired instructions, much like the dtlb_miss_ratio but tailored to the instruction TLB.

$$itlb\_miss\_ratio = iTLB - load - misses \,/\, inst\_retired$$

vi) L1D Miss Ratio

This ratio assesses the efficiency of the Level 1 data cache by dividing the total number of Level 1 data cache loads by the number of misses in the cache.

$$L1D\_miss\_ratio = L1 - dcache - load - misses \,/\, L1 - dcache - loads$$

vii) L1I Miss Ratio

This feature records the ratio of L1 instruction cache misses to the total number of L1 instruction cache loads, much like l1d_miss_ratio but for the Level 1 instruction cache.

$$L1I\_miss\_ratio = L1 - icache - load - misses \,/\, L1 - icache - loads$$

viii) L2D Miss Ratio

The effectiveness of the L2 cache can be inferred from this ratio, which shows the misses in the Level 2 data cache in relation to the total number of Level 2 data cache accesses (reads and writes).

$$l2d\_miss\_ratio = l2d\_cache\_refill \,/\, (l2d\_cache\_rd + l2d\_cache\_wr)$$

b) Composite Features:

A single composite value is produced by combining several related metrics using these features.

i) Branch Efficiency

This feature calculates the ratio of correctly predicted branches to all predicted branches in order to assess the effectiveness of the branch prediction mechanism.

$$branch\_efficiency = (br\_pred - br\_mis\_pred) \,/\, br\_pred$$

ii) Memory Pressure

By combining cache misses and page faults, this composite feature shows how much strain the memory subsystem is under overall.

$$memory\_pressure = cache - misses + page - faults$$

iii) TLB Pressure

The overall pressure on the TLB subsystem is represented by this feature, which aggregates the misses in the data and instruction TLBs.

$$tlb\_pressure = dTLB - load - misses + dTLB - store - misses + iTLB - load - misses$$

iv) Cache Writeback Pressure

The writebacks from the L1 and L2 caches are combined in this feature, which shows the strain on the cache writeback mechanism and possible memory hierarchy bottlenecks.

$$cache\_writeback\_pressure = l1d\_cache\_wb + l2d\_cache\_wb$$

v) Memory Access Intensity

This feature calculates the total memory access intensity by adding the read and write memory accesses.

$$memory\_access\_intensity = mem\_access\_rd + mem\_access\_wr$$

vi) Context Switch Rate

This feature provides information about the overhead related to task switching and possible bottlenecks by capturing the rate of context switches in relation to the total number of CPU cycles.

$$context\_switch\_rate = context - switches \,/\, cpu\_cycles$$

vii) Sycall Intensity

Summing up the different system call entry events and normalizing by the total number of instructions executed is how this composite feature calculates the intensity of system call invocations.

$$syscall\_intensity = sum(syscalls\_sys\_enter\_*) \,/\, instructions$$

viii) Disk IO Intensity

This feature gives an overall measure of disk I/O activity by combining the disk I/O requests that have been initiated and completed.

$$disk\_io\_intensity = mmc\_mmc\_request\_done + mmc\_mmc\_request\_start$$

Table 4: Sample set of Novel Features

dtlb_miss_ratio	itlb_miss_ratio	l1d_miss_ratio	l1i_miss_ratio	l2d_miss_ratio	tlb_pressure
0.003404	0.00038	0.041087	0.01088	0.244415	1.56E+08
0.003115	0.000214	0.049504	0.008321	0.263711	1.4E+08
0.003018	0.000307	0.043058	0.009607	0.231759	1.36E+08
0.003377	0.000464	0.042723	0.012196	0.232323	1.47E+08
0.003297	0.000404	0.038692	0.013482	0.241647	1.47E+08

3.4 Enhanced Dynamic Threshold Whale Optimization based Feature Selection (EDTWFS)

The improved Whale Optimization Algorithm (WOA) with dynamic thresholding offers a fresh approach to tackling the complexities of feature selection in datasets with many dimensions. By incorporating a way to change the threshold used to turn continuous whale positions into binary numbers, this EDTWS improves the WOA. This makes the balance between exploring information and speeds up finding with the best solution. It makes the process of choosing features which is more accurate. The threshold is set based on how well the algorithm is doing

and the nature of the search area. The initial threshold is set to make sense based on the range of feature values in the dataset.

By using a flexible method for how the computer sees the whale positions, the algorithm change how it shows features. It can explore a feature subsets of potentially profitable solutions. This adaptability helps the algorithm avoid ending at solutions that are not the best ones. This makes it more adaptable and gives it better performance and understanding of the problem.

Input: X=x1, x2..,xn : Dataset with 'n' distance and d features. y=y1,y 2,..,yn: Corresponding labels

N= Population Size, T= max. No. of iterations, b=Constant, lb=lower bound, ub= upper bound

Output: S, Selected Feature

Algorithm EDTWFS

1. Initialize whale population X with N whales of dimension d: Xi=(x_{i1},x_{i2}..,x_{id}), $x_{ij} \in [lb_j, ub_j]$
2. Update dynamic threshold '$thres_0$' based on the range of feature values.
3. For each iteration t= 1, 2,..T:
   - Update $thres_t$ based on convergence.
   - Update whale positions with following equations:
   - Encircling prey:

     $$\vec{D} = |\vec{C}.\vec{X}^t - \vec{X}_i^{(t)}|$$

     $$\vec{X}_i^{(t)} = \vec{X}^t - \vec{A}.\vec{D}$$

   - Bubble-net attacking:

     $$\vec{X}_i^{(t+1)} = \vec{X} * t + \vec{D}^l.e^{bl}.\cos(2\pi l)$$

     - Search for Prey:

     $$\vec{D} = \left|\vec{C}.\vec{X}_{rand}^{(t)} - \vec{X}_i^{(t)}\right|$$

     $$\vec{X}_i^{(t)} = \vec{X}_{rand}^{(t)} - \vec{A}.\vec{D}$$

   - Update whale positions $X_i^{t+1)}$ to binary vectors $X_{bin}^i(t+1)$ using $thres_t$.
   - Evaluate fitness of updated whale positions $X_{bin}^i(t+1)$
   - Update best whale X∗ if a better solution is found.
4. Repeat steps 1-5 until reaching the maximum iterations T.
5. Select the better Solution related Features S and return it.

End Algorithm

Algorithm 2: Enhanced Dynamic Threshold Whale Optimization based Feature Selection

To solve the feature selection problem, the **EDTWFS** creating a group of whales. each whale representing a potential solution. The whales' positions are based on the values, selects randomly. The idea of the **EDTWFS** is called "dynamic thresholding." It changes the whale positions from being continuous to being either 0 or 1. The threshold is adjusted at each step to help the algorithm find the best solution. Then, the **EDTWFS** uses a series of process to move the whales around. These allow the whales to explore parts of the search space, it also move whales in the direction of the best solution found so far. This way, the algorithm finds the best solution to the feature selection.

Based on the best solution, the location of the whales is updated by moving prey. To explore more of the search space around the best solution, used the bubble-net attack. To look deeper into the search space using the whales search for prey. The whales' updated positions are turned into binary vectors with a threshold. Then, a predefined objective function is used to evaluate the fitness of these binary vectors. If a better solution is found, the best solution is updated.

Until the maximum number of iterations has been reached, the process repeats itself. The algorithm searches through the search space and finds the best feature subset by slowly changing the positions of the whales and checking how well they fit. This goes on until the right combination of features is found, which makes the objective function reach its highest point.

The feature importance of the features can be seen in the figure 4, arranged according to their ranks. Approximately one-third of the features in the dataset were chosen as the best ones. Most of these chosen features were related to the most important ones in the dataset. This means that the most important features were found and the less important ones were left out. This shows that the EDTWFS algorithm worked well and found the right features. The EDTWFS algorithm can make problems simpler and make the optimization process faster. It does this by only picking the most important features. Because of this, the algorithm can be used to solve feature selection problems and make them easier to understand.

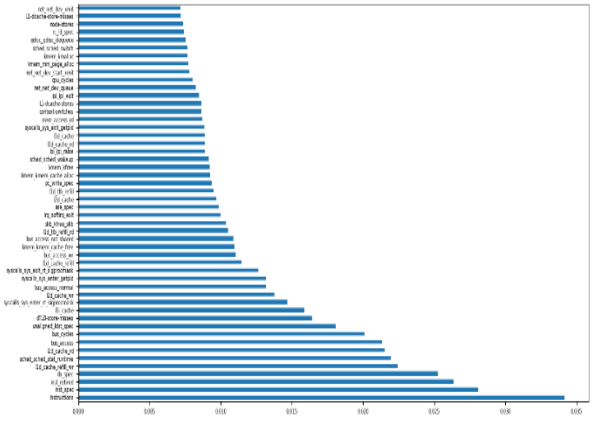

Fig 4: Feature Importance

3.5 Intrusion Predictor

Intelligently intrusion predictors enhance the security. This system distinguishes between attack and normal data by combining AI algorithms with anomaly detection techniques. Three main parts of this work includes, the dataset splitting, Federated Averaging Learning Classifier (FALC) algorithm, and class labelling. When combined, these components create a strong and proactive approach to network security and IDS.

3.5.1 Multi Class Labelling

The process of labelling (Class) to a group of data based on its features is called Class-Labelling. This is done using a training set, which shows the relationships and patterns between the different groups in the data. The goal of classification is to teach a model how to predict the label of new, unseen data accurately, so it can be used for supervised learning tasks.

Table 5: Class Label

Label	Scenario	Attack	Class
Normal	*Benign*	None	0
Attack	*Crypto-jacking*	Crypto-jack	1
	Recon	Aggressive-Scan	2
		OS-fingerprinting	3
		Port-scan	4
		Service-detection	5
		Syn-stealth	6
		Vulnerability-scan	7
		Tcp-flood	8
	DOS	Icmp-flood	9
		Icmp-fragmentation	10
		Push-ack-flood	11
		Syn-flood	12
		Tcp-flood	13
		Upd-flood	14
		Synonyms-ip-Flood	15

Table 5 shows a classification system for IDSs that groups them based on labels, scenarios, and specific classes. For example, "Normal" activities are labeled as "Benign" and placed in Class 0, indicating normal or non-threatening behavior. However, in the context of the "Attack" scenario, attacks such as "Crypto-jacking" are classified as Class 1 and belong to the "Crypto-jack" category. The table also lists reconnaissance (Recon) tasks with different class assignments, ranging from 2 to 8, including "Aggressive-Scan," "OS-fingerprinting," and "Port-scan.". Furthermore, the "DOS" scenario is used to classify Denial-of-Service (DOS) attacks such as "Icmp-flood," "Syn-flood," and others, with classes ranging from 9 to 15. Using a structured classification scheme, various network behaviour's and threats can be arranged and comprehended according to their characteristics and effects.

3.5.2 Dataset Splitting

One of the important steps in classifying data is to split it into two groups: a training set and a test set. The training set is used to teach the AI model, while the test set is used to measure its performance on data it has not seen before. In this work, dataset is divided at training set has 5925 rows, which is 70% of the total data, and the test set has 2540 rows, which is 30% of the total data.

3.5.3 Federated Averaging Learning Classifier (FALC)

Through the work of many clients from different places, federated learning (FL) is a way to teach machines using data that is spread out over many locations. This method doesn't require client to share their private information, which is an important part of keeping it safe. The most well-known algorithm for this kind of learning is called Federated Averaging (FedAvg). It lets machines learn from each other by sharing updates based on the latest data they've seen.

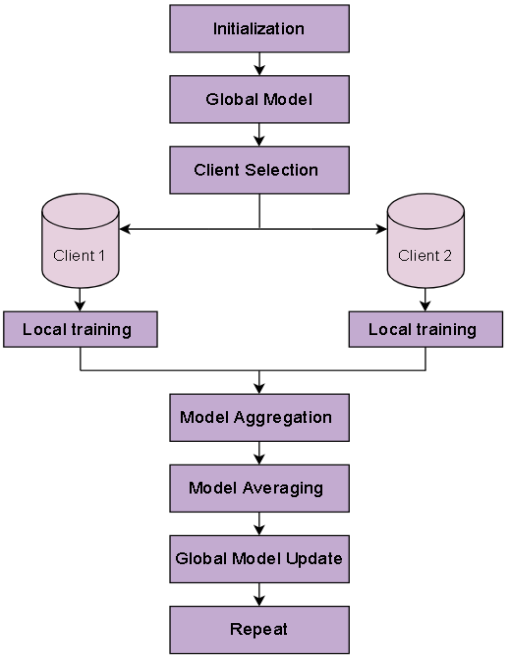

Fig 5: Flow of the Federated Learning

This proposed system is using FALC to detect the IDS to keep up with new threats. It does this by always learning from the most recent data it can get, which comes from each charging station (Clients). Because of this, the IDS can find new or emerging threats with a high degree of accuracy and stay up-to-date.

Many clients use FALC to train a model without sharing their private data. This method is best for situations where keeping data safe. The flow of the FALC in figure 5 and the steps of the FALC as follows:

1. **Initiation**: Starting with the Central Server (C_S) and global model(θ_g).
2. **Selection**: subset of clients (S_C) is chosen randomly for participate in the round.
3. **Distribution**: Replicas of the θ_g is given to the S_C.
4. **Training**: Each S_C trains the model includes several epochs with local data.
5. **Aggregation**: After local training, the updated models from each S_C are transmitted C_S.
6. **Averaging**: By averaging the models, the C_S compiles the models that it has received from S_C.

This process is repeated for multiple training rounds until the θ_g performance meets the desired criteria.

Input: K: No.of clients, R: No. of communication rounds, D_k: Local data of client k, M: Model architecture,
η: Initial learning rate, E: epochs, - B: Batch size, T: Fraction of clients, λ: Regularization parameter, α: Learning rate decay, split_ratio: Ratio of Dataset
Output: θ_g: Trained global model parameters, - test_metrics: Performance metrics

Algorithm: Federated Averaging Learning Classifier (FALC)

1. Initialize θ_g with random values
2. Split each client's data into training and testing sets:
   for k = 1 to K: D_k_train, D_k_test ← split (D_k, split_ratio)
3.For each communication round (r = 1, 2, ..., R):
   Select a subset of clients (S) to participate in this round:
   S ← random_sample(K, T)
   For each client k in S:
      for e = 1 to E:
         for batch in D_k_train:
            Update local model θ_k using optimizer opt:
            θ_k ← optimizer.update(θ_k, ∇(L(θ_k, batch) + λ * R(θ_k)), η)
            Compute model update Δθ_k:
            Δθ_k ← θ_k - θ_g
            Δθ_g ← FedAvg({Δθ_k}_{k∈S})
            Update global model θ_g: θ_g ← θ_g + Δθ_g
            η ← η * α
            global_test_set ← concatenate({D_k_test}_{k=1}^K)
            Compute performance metrics test_metrics:
            test_metrics ← evaluate (θ_g, global_test_set)
            Print performance metrics for current round r.
   for k = 1 to K:
      θ_k ← θ_g
4. Evaluate the global model (θ_g) on the combined global test set:
   global_test_set ← concatenate({D_k_test}_{k=1}^K)
   Compute performance metrics test_metrics:
   test_metrics ← evaluate (θ_g, global_test_set
End Algorithm

Algorithm 3: FALC

Federated Averaging Learning Classifier (FALC) algorithm 3 is a state-of-the-art machine learning technique that tackles the difficulties presented by decentralized data sources. FALC assists in training models by collaborative models among many clients, avoiding centralized storage and unnecessary communication overhead when shared models are frequently dispersed across multiple clients.

1.Initialization and Data Splitting

At Initial, random values are assigned to the global parameters (θ_g). The data for each client (D_k) is then split into two parts: a training & testing set, using a specific split ratio. A model with this division will not over fit to the training set. It will be unable to generalize to new data if it over fits to the training set.

2. Communication Rounds and Client Participation

Each communication round, a subset of clients (S) is randomly selected. It determines which clients participate in

each round, and how much work is required for all types of data contributions. Those clients whose training data are selected update their local models by using θ_k. To improve its convergence and stability, the learning rate is gradually decreased through iterations. Also, updates are made more efficient using an optimizer and a regularization parameter (λ).

3.Global Model Aggregation and Evaluation

The division prevents the model from overfitting to the training set and being unable to generalize. Using FedAvg and the individual model updates (Δθ_k) from each participant, the global update (Δθ_g) can be calculated. It is possible to operate on all available data using the global model (Δθ_g) using various data. Using the evaluate function, we validate the global model's performance on test data, and we then use the metrics to guide future improvements.

4.Model Synchronization and Final Evaluation

To improve cohesion and speed in local submissions, the models are synced with the global model at the end of each round. Over this step, the global model reflects the knowledge of all clients. To evaluate the global model's general success, a final analysis is conducted on the combined global test set to establish the overall capabilities of the global model. In the FALC model assessment stage, which is displayed in table 6, the model's effectiveness and generalizability are examined comprehensively across all client data to illustrate its generalizability and dependability.

Table 6: Evaluation metrics over Multi Clients and Rounds

Dataset	Clients (Charging Station)	Rounds	global_acc	global_precision	global_recall	global_f1
SF	5	10	99.9880	99.9281	99.6479	99.7798
	10	10	99.9606	99.9423	99.9113	99.9556
	15	10	99.9213	99.9321	99.8225	99.9112
	5	20	99.9930	99.4573	99.5729	99.6476
	10	20	99.9606	99.9942	99.9113	99.9556
	15	20	99.9326	99.9937	99.9323	99.9956
NF	5	10	99.7637	99.5606	99.9118	99.7359
	10	10	99.8031	99.6482	99.9281	99.7798
	15	10	99.8650	99.5599	99.7354	99.6476
	5	20	99.7638	99.6479	99.8236	99.7357
	10	20	99.8367	99.5729	99.8362	99.7375
	15	20	99.6850	99.6473	99.4573	99.6473

The two types of datasets, SF and NF, are carefully tested to measure how well they perform. In these tests, clients from the SF dataset show strong and dependable behavior. They show high levels of accuracy, precision, recall, and F1 score in many rounds. The clients in the SF dataset also do very well, especially in round 5 with 20 clients, where they get an F1 score of 99.56%. Nevertheless, the NF dataset also demonstrated excellent performance, if marginally worse than the SF clients in the majority of metrics.

In spite of this, during the testing rounds, NF clients remained highly dependable and consistent. The effectiveness and stability of the clients under evaluation are generally highlighted by the SF and NF datasets, which also provide insightful information for future optimization and improvement of systems that depend on these kinds of client interactions.

4. OVERALL PERFORMANCE EVALUATION

This section assesses the suggested system's/method's overall performance under different experimental conditions and metrics. The experimental setup is described in 4 Point 1, the accuracy and error rate results are shown in 4 Point 2, the precision and recall are covered in 4 Point 3, and the F1-score is covered in 4 Point 4. Figure 6 shows the frequencies or counts of different kinds of attacks. The various attack categories are listed on the x-axis, and the counts are represented on the y-axis. The most frequent attack, with 1793 incidents reported, is crypto-jacking. 4604 attacks did not take place.

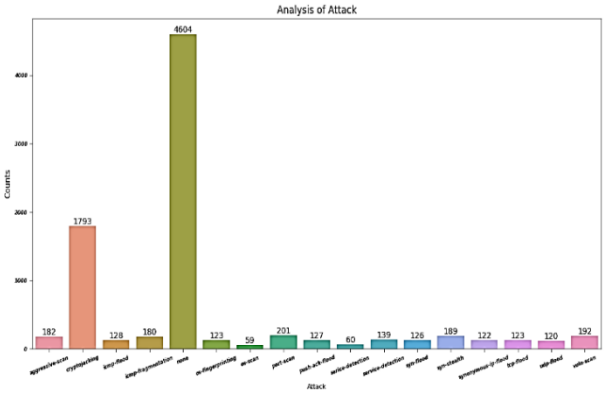

Fig 6: Attack Analysis

4.1 Experimental Setup

The work's implementation environment is designed to leverage with a 64-bit Windows operating system, an Intel Core i7-2620M Processor, 16 GB of RAM, and a 500 GB hard drive. This hardware configuration used in this work and the implementation managed by the Anaconda distribution. Anaconda provides a comprehensive Python environment with streamlined package and environment management, acting as the backbone of the system. This experiment makes use of several powerful libraries, including NumPy, Pandas, scikit-learn, Matplotlib, TensorFlow, and Keras. Two datasets were utilized in this work: Selected Features (SF) and Novel Features (NF). The SF dataset comprises a carefully chosen selection of features from the dataset, whereas the NF dataset adds new features. In terms of various machine learning techniques, the models that are evaluated are SVM, Random Forest, ANN, CNN, and FALC.

Table 7: Overall Evaluation

Data Set	Model	Accuracy	Error Rate	Precision	Recall	F1-Score
Selected Feature (SF)	SVM	89.19	10.81	90.30	89.19	89.07
	RF	92.41	7.59	92.63	92.41	92.47
	ANN	89.17	10.83	90.06	89.17	88.84
	CNN	86.85	13.15	88.07	86.85	86.24
	FALC	**99.98**	**0.02**	**99.92**	**99.64**	**99.77**
	SVM	79.71	20.29	72.68	79.71	74.62
Novel Feature (NF)	RF	88.15	11.85	87.75	88.15	87.85
	ANN	83.50	16.50	81.64	83.50	81.89
	CNN	82.22	17.78	81.93	82.28	80.46
	FALC	**99.86**	**0.14**	**99.53**	**99.64**	**99.73**

The experimental results are presented in Table 7. The performance metrics that correspond to each row in the table are displayed in the columns. The row describes a model and dataset combination. The Selected Features (SF) dataset was analysed by the Random Forest model, it has 7.59% error rate and 92.41% accuracy and SVM has 89.19 %, FALC has 99.98 % accuracy on the SF Dataset. Next, using the Novel Features (NF) dataset, the FALC model was evaluated, it has a high accuracy of 99.86 % and an F1-score of 99.73 %. The SVM model exhibited an F1-score of 74.62% and a relatively high error rate of 20.29%, despite achieving a lower accuracy of 79.71% for the NF Dataset. It is not worth that the SF and NF datasets demonstrated significant differences in the performance of certain models, such as CNN and ANN, highlighting the sensitivity of these models to feature representations.

4.2 Accuracy & Error Rate

Two key performance indicators in machine learning classification tasks are accuracy and error rate, which reveal information about how well a model can classify instances. These measurements are used as key performance indicators. Accuracy is the % of correctly classified examples among all evaluated examples; it shows how well the model predicts things. On the other hand, error rate shows what % of the dataset's instances the model incorrectly classified [21].

$$Accuracy = \frac{\sum Correct\ Prediction}{\sum Total\ Prediction} * 100\%,$$

$$ErrorRate = 1 - Accuracy * 100\%$$

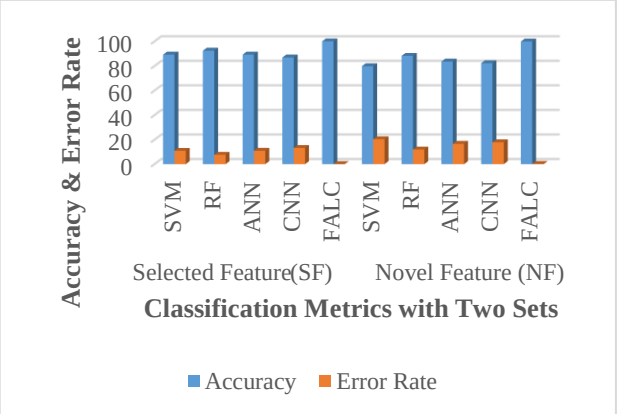

Fig 7: Accuracy & Error Rate

The "Selected Feature" data set is displayed in Figure 7. FALC produced impressive results, with an accuracy of 99.98 % and an incredibly low error rate of 0 points02 %. With an accuracy of 92.41 % and an error rate of 7.59 %, Random Forests secured the second position. CNN trailed behind with an accuracy of 86.85 % and an error rate of 13.15 %, while ANN and SVM demonstrated comparable accuracy of approximately 89 %. With 99.86 % accuracy and a mere 0.14 % error rate on the "Novel Feature" data set, FALC once again showed itself to be the best performer. Even with an increase in error rate to 11.85 %, RF's accuracy remained strong at 88.15 %. The accuracy of the neural network models, CNN and ANN, was higher at 82.22 % and 83.20 %, respectively, while the error rates were also higher. On both data sets, FALC consistently performed significantly better than the other models.

4.3 Precision & Recall

In classification tasks, precision and recall are important metrics that provide detailed information about a model's performance. Precision shows the %age of correctly identified relevant instances among all instances labeled as positive, thereby measuring the accuracy of positive predictions. Conversely, recall measures the sensitivity of the model by evaluating its capacity to extract all pertinent examples from the total number of actual positives.

$$Precision = TP\ /\ (TP\ +\ FP), Recall = TP\ /\ (TP\ +\ FN)$$

Figure 8 displays metrics for precision and recall for various machine learning models that were assessed using two sets of data. FALC obtained the highest recall of 99.64 % and the highest precision of 99.92 % for the "Selected Feature" data set. With a recall of 92.41 % and precision of 92.63 %, Random Forests fared well as well. While SVM and ANN had somewhat lower recall values, their precision scores were comparable, hovering around 90%. On this data set, CNN had the lowest recall (86.85 %) and precision (88.07 %). FALC was the best performer once more on the "Novel Feature" data set, achieving a precision of 99.53 % and a recall of 99.64 %. When compared to the "Selected Feature" set, though, its performance advantage over other models was less pronounced.

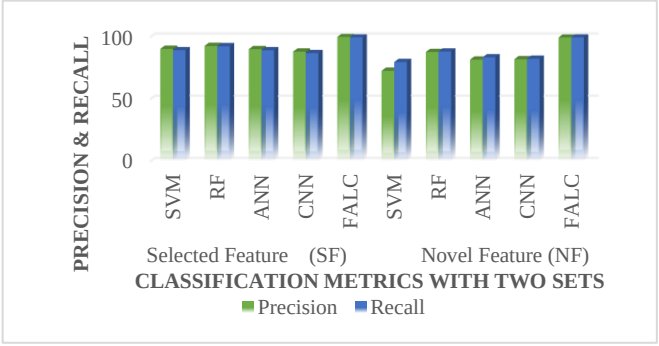

Fig 8: Precision & Recall

With a recall of 88.15 % and a precision of 87.75 %, RF demonstrated commendable performance. Lower precision and recall scores were shown by the neural network models, CNN and ANN, with CNN scoring 81.93 % and 82.28 %, respectively, and ANN scoring 81.64 % and 83.50 %, respectively.

With a precision of 72 points 68 % and a recall of 79 points 71 % on the "Novel Feature" set, SVM notably displayed a more notable decline in performance, indicating possible overfitting or sensitivity to the novel feature space.

4.4 F1-Score

The F1 score provides a thorough assessment of the performance of a classification model and is calculated as the harmonic mean of precision and recall. It gives a fair evaluation by taking recall and precision into account at the same time. The harmonic mean of precision and recall is used to compute the F1 score, which has two possible values: 1 (perfect precision and recall) and 0 (worst).

$$F1 - Score = 2X\frac{Precision\ X\ Recall}{Precision + Recall}$$

Using the "Selected Feature" data set, as shown in Figure 9, FALC outperformed the other models, earning the highest score of 99 points 77. With a score of 93.47, Random Forests (RF) came in second, followed by ANN with 88.84, SVM with 89.07, and CNN with 86.24, which was the lowest score on this set of data. Maintaining its outstanding performance, FALC scored 99 points 73 on the "Novel Feature" data set, making it the top performer once more. When compared to the "Selected Feature" set, the performance difference between FALC and the other models was smaller. With a commendable score of 81.89, RF was closely followed by ANN, who scored 81.39. SVM scored 74 points, while CNN scored 80 points, showing lower scores.

Overall, FALC proved to be robust and effective as it consistently outperformed the other models on both data sets. Artificial neural networks and random forests also performed fairly well, especially when used with the "Novel Feature" set. But in general, especially when it came to the "Novel Feature" set, the Convolutional Neural Network (CNN) and Support Vector Machine (SVM) models performed worse than the other methods assessed in this study.

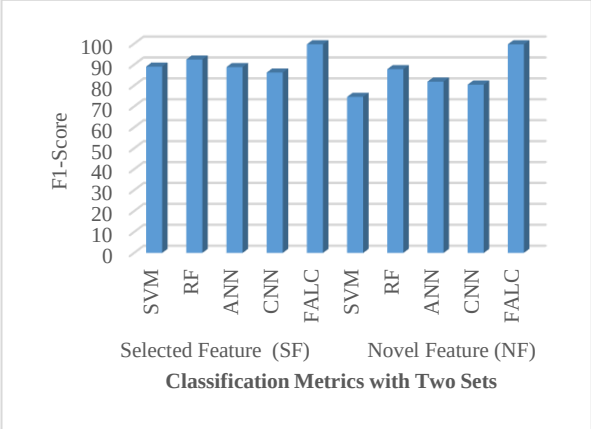

Fig 9: F1-Score

4. COMPARATIVE ANALYSIS

Recently, deep learning and machine learning have been used extensively to help identify anomalies [15, 16]. Selecting the best algorithm is where the difficulty lies. A precise solution with minimal computational overhead is the aim. This aids in the timely detection of attacks and the termination of risky communications. Machine learning classifiers need to be able to make the correct decision with the least amount of data possible, even when they have only been trained with a small number of data points. Algorithms for unsupervised learning are often employed to thwart unidentified threats (also known as zero-day attacks). Unsupervised learning algorithms do have certain disadvantages, though, including a high false alarm rate and a low detection rate.

As far as we are aware, unsupervised learning methods only achieve a maximum performance of about 90%, which is not good enough to fend off unidentified threats. On the other hand, supervised learning algorithms can be used to achieve high performance, up to 99 %. The achieved performance is similar to intrusion detection systems that rely on signatures. Regularization techniques are the answer to this conundrum and how to solve it. supervised learning algorithms can perform better for the same problem than unsupervised learning algorithms because they can be used to simplify the system and solve the overfitting issue in supervised learning [17].

EV charging stations gather and handle private user data, including payment information, charging habits, and personal data. This data must be combined from various charging stations into a central server in order to train models using traditional centralized machine learning and deep learning techniques. There are serious privacy issues with this data aggregation because it may expose or compromise sensitive user data. Since EV charging stations are spread out geographically, the data produced at each site might have particular traits or trends.

Federated learning substantially lowers the cost of training models across multiple distributed data sources by enabling the direct extraction of heterogeneous data patterns from their sources without aggregating them at a central location. The volume of generated data is growing along with the number of electric vehicle (EV) users and charging stations. Scalability becomes a critical issue for centralized machine learning and deep learning methods when handling such massive volumes of data. Federated learning, on the other hand, gets around this problem by sharing the processing load across a number of clients (charging stations), which makes large-scale model training and updates possible. Processing and managing this enormous volume of data may present scalability issues for centralized machine learning and deep learning techniques. Federated learning allows for more effective and scalable model training and updating by dividing the computational load among several clients (charging stations).

Similar to the proposed study, ElKashlan [1] uses a native IoT dataset and various machine learning algorithms to detect fraudulent traffic in order to address IoT security vulnerabilities in EVCS. In that work, machine learning techniques like Decision Tree (DT), Logistic Regression (LR), and Naive Bayes (NB) were applied. In comparison to state-of-the-art techniques like those documented in [14,18], this work [1] uses different Deep learning algorithms (highest accuracy of 97%).The Filtered Classifier (99.99%) was the binary classification algorithm with the best accuracy. The suggested work, however, seems to perform better than the published findings. This study compares different machine learning classifiers; however, the proposed work introduces Multi-class using the proposed FALC model, which achieves remarkably high precision (99.92% and 99.53%), recall (99.64% for both), accuracy (99.98% on Selected Feature and 99.86% on Novel Feature), and F1-scores (99.77% and 99.73%).

Zhaoliang Lin outlines a model in [2] that combines an autoencoder, batch normalization, and attention mechanism with an unsupervised representation learning algorithm to improve accuracy and feature representation. The current work surpasses the findings published. Despite the fact that it does not reveal information about the underlying model architecture. regarding recall, F1-score, accuracy, and precision. Not the main focus of the paper, but the proposed work specifically addresses the problem of fraudulent traffic detection in EVCS networks. ...

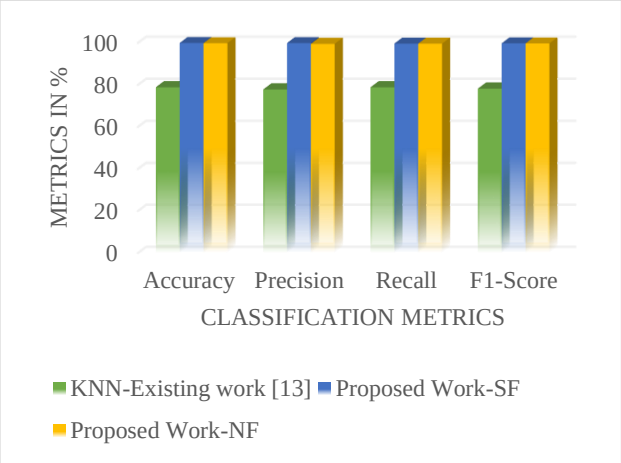

Fig 10: Comparative Analysis

The CICEVSE2024 dataset was created by the work [13], which also assessed 15 network-based attack scenarios by applying machine learning classification tasks to host and network traffic. Figure 10 displays the results: 78.88 % accuracy for multi-class classification using Random Forest and 0.91 % accuracy for binary classification using KNN.

6. Conclusion

The Proposed work FALC-IDS is presented in this work for Securing EVCS. It has data preprocessing methods, such as Removal of Missing, Duplicate Value & Unwanted Column, and Data Encoding. It creates the Novel Features (NF) generation and selects the best features (SF) using EDTWFS. The proposed model helps for user privacy and security of EVCS. Using the FALC technique, this proposed collaborative model trains more clients. Compared to existing models, the FALC model has higher performance, 99.98% accuracy for the SF dataset, and 99.86% accuracy for the Novel Features (NF) dataset. The security solution for EVCS may exist in the future by combining the FALC model with security measures. In the future, transfer learning techniques adapt the FALC model for EVCS. Also, includes smart grid security or automotive cybersecurity.

Author Contributions

The corresponding author confirm sole responsibility for the following: study conception and design, data collection, analysis and interpretation of results, and manuscript preparation.

Data availability

The input dataset used is the publicly available benchmark dataset CICEVSE2024 dataset. Data is available.

Funding

This research received no external funding.

Conflicts of Interest

The authors declare no conflict of interest.

Ethics approval

Not applicable

References

[1] ElKashlan, M.; Aslan, H.; Said Elsayed, M.; Jurcut, A.D.; Azer, M.A. Intrusion Detection for Electric Vehicle Charging Systems (EVCS). *Algorithms* 2023, *16*, 75. https://doi.org/10.3390/a16020075

[2] Zhaoliang Lin, Jinguo Li, FedEVCP: Federated Learning-Based Anomalies Detection for Electric Vehicle Charging Pile, *The Computer Journal*, 2023.

[3] Mukhtar, B.I.; Elsayed, M.S.; Jurcut, A.D.; Azer, M.A. IoT Vulnerabilities and Attacks: SILEX Malware Case Study. *Symmetry* 2023, *15*, 1978. https://doi.org/10.3390/sym15111978

[4] J. Cumplido, C. Alcaraz, and J. Lopez, “Collaborative anomaly detection system for charging stations”, The 27th European Symposium on Research in Computer Security (ESORICS 2022) vol. 13555, pp. 716736, 2022. http://doi.org/https://doi.org/10.1007/978-3-031-17146-8_35

[5] Samrat Acharya, Hafiz Anwar Ullah Khan, Ramesh Karri, and Yury Dvorkin, "MaDEVIoT: Cyberattacks on EV Charging Can Disrupt Power Grid Operation", arXiv:2311.06226v1 [eess.SY] 10 Nov 2023

[5] Dustin Kern, Christoph Krauß, Matthias Hollick, "Detection of Anomalies in Electric Vehicle Charging Sessions", ACSAC '23: Annual Computer Security Applications Conference, Austin, TX, USA, December 2023, DOI: https://doi.org/10.1145/3627106.3627127

[6] Hamdare S, Kaiwartya O, Aljaidi M, Jugran M, Cao Y, Kumar S, Mahmud M, Brown D, Lloret J. Cybersecurity Risk Analysis of Electric Vehicles Charging Stations. Sensors (Basel). 2023 Jul 27;23(15):6716. doi: 10.3390/s23156716. PMID: 37571500; PMCID: PMC10422437.

[7] ElKashlan, M.; Elsayed, M.S.; Jurcut, A.D.; Azer, M. A, “Machine Learning-Based Intrusion Detection System for IoT Electric Vehicle Charging Stations (EVCSs)”, *Electronics*, *2023*, *12*, 1044. https://doi.org/10.3390/electronics12041044

[8] M. Basnet and M. Hasan Ali, "Deep Learning-based Intrusion Detection System for Electric Vehicle Charging Station," *2020 2nd International Conference on Smart Power & Internet Energy Systems (SPIES)*, Bangkok, Thailand, 2020, pp. 408-413, doi: 10.1109/SPIES48661.2020.9243152.

[9] Vinod Miskin, P. Chandaragi and U. V Wali, "Intrusion Detection System for Electric Vehicle

Charging Station," 2023 3rd International Conference on Mobile Networks and Wireless Communications (ICMNWC), Tumkur, India, 2023, pp. 1-7, doi: 10.1109/ICMNWC60182.2023.10435897.

[10] Balakrishna, A. Kumar, T. V, A. Younas, N. M. G. Kumar and R. Rastogi, "A Novel Ensembling of CNN-A-LSTM for IoT Electric Vehicle Charging Stations Based on Intrusion Detection System," 2023 International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS), Erode, India, 2023, pp. 1312-1317, doi: 10.1109/ICSSAS57918.2023.10331735.

[11] N. Singh and R. Agarwal, "Intrusion Detection System for Smart Vehicles Using Machine Learning Algorithms," 2023 International Conference on Communication, Security and Artificial Intelligence (ICCSAI), Greater Noida, India, 2023, pp. 749-753, doi: 10.1109/ICCSAI59793.2023.10421298.

[12] E. D. Buedi, A. A. Ghorbani, S. Dadkhah, and R. Ferreira. “Enhancing EV Charging Station Security Using A Multi-dimensional Dataset: CICEVSE2024”. - (Submitted to ESORICS 2024 Conference).

[13] Amanoul, S.V.; Abdulazeez, A.M. Intrusion detection system based on machine learning algorithms: A review. In Proceedings of the 2022 IEEE 18th International Colloquium on Signal Processing & Applications (CSPA), Selangor, Malaysia, 12 May 2022; pp. 79–84.

[14] Balaji, R.; Deepajothi, S.; Prabaharan, G.; Daniya, T.; Karthikeyan, P.; Velliangiri, S. Survey on intrusions detection system using deep learning in iot environment. In Proceedings of the 2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS), Erode, India, 7–9 April 2022; pp. 195–199.

[15] Zeadally, S.; Tsikerdekis, M. Securing internet of things (iot) with machine learning. Int. J. Commun. Syst. 2020, 33, e4169.

[16] Ying, X. An overview of overfitting and its solutions. In Journal of Physics: Conference Series; IOP Publishing: Bristol, UK, 2019; Volume 1168, p. 022022.

[17] Ahmad, R.; Alsmadi, I.; Alhamdani, W.; Tawalbeh, L. A comprehensive deep learning benchmark for iot ids. Computer. Security, 2022, 114, 102588.

[18] D.V. Jeyanthi and B. Indrani, "An Efficient Intrusion Detection System with Custom Features using FPA-Gradient Boost Machine Learning Algorithm", International Journal of Computer Networks & Communications (IJCNC), Vol.14, Issue.1, pp.99-115,2022.

[19] T.S. Urmila, Dr. R. Balasubramanian, "Dynamic Multi-layered Intrusion Identification and Recognition using Artificial Intelligence Framework", International Journal of Computer Science and Information Security (IJCSIS), Vol. 17, No. 2, pp.137-147, 2019.

[20] G.B. Govindaprabhu, M. Sumathi, "Ethno medicine of Indigenous Communities: Tamil Traditional Medicinal Plants Leaf detection using Deep Learning Models", Procedia Computer Science, Volume 235, 2024, Pages 1135-1144, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2024.04.108.

Authors Profiles

Dr.A.Bamini (**Corresponding Author**) is the Head and Assistant Professor of the Department of Computer Applications at The Standard Fireworks Rajaratnam College for Women in Sivakasi. She completed her Master of Computer Applications (MCA) in 2001 from SFR College, followed by a Master of Philosophy (M.Phil.) from Mother Teresa University in 2003. In 2018, she earned her doctoral degree (Ph.D.) from Karunya University. She also qualified for the National Eligibility Test (NET), demonstrating her expertise in the field of computer science. she published 10 journal papers and the presentation of 6 papers at various conferences and published 1 book. She has a total of 23 years of teaching experience. ORCID: 0000-0001-7559-6943.

Mr.V.Arun Raj, Assistant Professor (Sr.Grade) in the Department of Electronics and Communication Engineering at Mepco Schlenk Engineering College, Sivakasi. He has been a part of the faculty since June 2016. He completed his Bachelor of Engineering in Electronics and Communication Engineering in 2013 and his Master of Engineering in Communication Systems in 2015, from Anna University, Chennai. He has a teaching experience spanning over 8 years with a dedicated to imparting knowledge and fostering the academic growth of his students. ORCID: 0000-0003-0950-809X.