Computer Systems Science and Engineering

Tech Science Press

DOI: 10.32604/csse.2024.042401

ARTICLE

A Hybrid Machine Learning Framework for Security Intrusion Detection

Fatimah Mudhhi Alanazi*, Bothina Abdelmeneem Elsobky and Shaimaa Aly Elmorsy

Mathematics and Computer Science Department, Faculty of Science, Alexandria University, Alexandria, Egypt

*Corresponding Author: Fatimah Mudhhi Alanazi. Email: Fatmah9-6-1413@hotmail.com

Received: 29 May 2023 Accepted: 08 January 2024 Published: 20 May 2024

ABSTRACT

Proliferation of technology, coupled with networking growth, has catapulted cybersecurity to the forefront of modern security concerns. In this landscape, the precise detection of cyberattacks and anomalies within networks is crucial, necessitating the development of efficient intrusion detection systems (IDS). This article introduces a framework utilizing the fusion of fuzzy sets with support vector machines (SVM), named FSVM. The core strategy of FSVM lies in calculating the significance of network features to determine their relative importance. Features with minimal significance are prudently disregarded, a method akin to feature selection. This process not only curtails the computational burden of the classification algorithm but also ensures the preservation of high accuracy levels. To ascertain the efficacy of the FSVM model, we have employed a publicly available dataset from Kaggle, which encompasses two distinct decision labels. Our evaluation methodology involves a comprehensive comparison of the classification accuracy of the processed dataset against four contemporary models in the field. Key performance metrics scores are meticulously calculated for each model. The comparative analysis reveals that the FSVM model demonstrates a marked superiority over its counterparts, enhancing classification accuracy by a minimum of 3%. These findings underscore the FSVM model's robustness and reliability, positioning it as a highly effective tool in the realm of cybersecurity.

KEYWORDS

Cybersecurity; fuzzy sets; classification; internet of things

1 Introduction

The growing imperative for robust cybersecurity measures has become increasingly vital in the contemporary digital era, primarily driven by the swift proliferation of the Internet of Things (IoT). This heightened demand for secure cyber environments is a direct response to the evolving landscape of digital connectivity and data exchange [1]. This uptick in cyber threats, including denial-of-service (DoS) attacks, malware intrusions, and unauthorized access, poses serious risks, particularly in large network environments. These cyber incidents can lead to significant financial repercussions; for instance, a single ransomware attack in 2020 inflicted an estimated loss of $20 billion in the healthcare industry alone [2]. Consequently, the urgency for effective cybersecurity strategies and countermeasures against a diverse range of online threats is intensifying with each passing day [3].

In the field of cybersecurity, deploying a network security system is of paramount importance, and IDS are central to protecting computer networks. These systems are instrumental in defending against a range of cyber threats, ensuring the integrity and safety of networked data and infrastructure.

While tools like firewalls and encryption techniques are integral, IDS stands out for its ability to defend against external threats. Its core responsibility lies in monitoring network activities and system usage to detect and avert harmful incidents. By closely observing standard network operations, an IDS is adept at identifying potential threats, including denial-of-service (DoS) attacks. Moreover, it efficiently detects patterns associated with cyber-attacks and unauthorized activities, such as unauthorized access, modification, or destruction of system data. These capabilities significantly surpass those of traditional security measures like firewalls, highlighting the essentiality of IDS in contemporary network security frameworks [4].

Intrusion Detection Systems (IDSs) are differentiated into several categories, aligning with their specific operational methodologies, among which Host-based (HIDS) and Network-based (NIDS) are notably prevalent [5]. HIDS functions at the individual machine level, vigilantly observing critical system files for signs of unusual or harmful activities. However, it encounters challenges in identifying new, unfamiliar malicious codes. In contrast, NIDS focuses on monitoring and analyzing network traffic to detect any abnormal activities. The evolution of IDS technology has particularly focused on enhancing detection methods, with both signature-based and anomaly-based detection techniques receiving significant research and development attention.

Signature-based IDSs excel in identifying known attacks by matching data patterns to known attack signatures, but they falter in detecting new and unprecedented attacks. Anomaly-based IDSs, in contrast, focus on understanding network behavior [6]. They create a model of normal behavior and flag deviations as potential threats. This method offers an edge in detecting novel security vulnerabilities or threats, yet it may lead to elevated false alarm rates due to misclassification of unknown but benign system behaviors as threats. Given these challenges, there is a growing consensus on the need for a machine learning-based approach to enhance detection capabilities. Such an approach aims to overcome the limitations of conventional detection methods, providing a more effective strategy to identify and mitigate a wide range of cyber threats. This research primarily focuses on exploring and developing machine learning-driven strategies to fortify IDSs against evolving online threats [7].

In the contemporary landscape of data classification methodologies, researchers have put forth various approaches, including neural networks, Bayesian networks, and notably, Support Vector Machines (SVM) [8]. Among these, SVM stands out as a particularly effective technique. Its applicability and efficiency have been proven in a multitude of contexts. SVM's prowess is evident in its application to diverse fields, such as medical imaging and disease diagnosis. For instance, SVM has significantly enhanced the speed and accuracy of identifying diffuse interstitial lung diseases in computer-aided quantification processes [9]. In the realm of hepatology, SVM has been instrumental in classifying liver diseases, achieving remarkably high accuracy rates compared to alternative methods [4]. Furthermore, SVM's versatility extends to a two-stage approach involving the artificial bee colony algorithm [10]. This growing body of evidence underscores SVM's robustness and adaptability, marking it as a key player in the field of data classification.

Lai et al. [11] introduced the Support Vector Machine (SVM) as a groundbreaking approach to address pattern recognition challenges. The core mechanism of SVM involves projecting data points into a high-dimensional feature space, followed by the determination of an optimal separating hyperplane. This hyperplane is strategically positioned to maximize the margin between two distinct

classes of data. A unique feature of SVM is its formulation as a quadratic programming (QP) problem, which not only guarantees a unique solution but also promotes sparsity, leading to excellent generalization capabilities. Despite these strengths, SVM faces challenges when dealing with real-world datasets that often contain noisy samples and outliers. The presence of these outliers and noise can significantly skew the results of a conventional SVM. To address these challenges, the innovative concept of Fuzzy Support Vector Machine (FSVM) was introduced [12]. It augments a conventional SVM framework by attributing a specific fuzzy membership value to each data point. This membership serves as an indicator of the sample's association with a particular class, thereby reflecting the importance of the sample in shaping the decision surface. By incorporating these fuzzy memberships, FSVM effectively mitigates the impact of noise and outliers, offering a more robust and reliable solution in complex, real-world data scenarios.

In the methodology presented within this study, we commence with an evaluation and prioritization of diverse security features, essential for effective intrusion detection modeling. The initial step involves meticulously assessing the features' importance in cybersecurity, thereby establishing a hierarchy of relevance. Subsequent to this feature selection process, our approach entails the construction of a tree-based, generalized model for intrusion detection. This model is intricately designed using the training security dataset, ensuring a comprehensive representation of potential cyber threats. The robustness and reliability of the model are further ascertained through rigorous validation using a separate set of test data. One of the merits of the proposed method is mitigating the risk of overfitting during the modeling phase. Additionally, by actively reducing the dimensions of features in the development phase, the computational complexity of the model is significantly lowered. These strategic measures collectively contribute to enhancing the accuracy of the model, particularly in predicting outcomes for unobserved test instances. This dual focus on minimizing complexity and maximizing predictive accuracy positions our proposed model as a proficient and reliable tool in the realm of cybersecurity.

The novelty of the research lies in our approach to solving the problem of constructing an effective, data-driven intelligent intrusion detection system (IDS) while mitigating the challenges of high dimensionality, computational cost, and training time inherent in handling contemporary security datasets.

The primary innovative aspects of our work are:

- We propose the FSVM, a framework integrating fuzzy logic SVM. Unlike conventional SVM, our FSVM-based model accounts for outliers and noise in the data. This ensures the robustness of our model and enhances its ability to generalize beyond the training data.
- Our model uniquely prioritizes security features based on their significance. We introduce a new process for security feature ranking using a criterion, presumably a metric like the Gini Index or information gain. This aspect of our approach facilitates an optimal selection of features, which reduces the feature dimensions and thus decreases the computational complexity.
- Based on the ranked security features, we develop a decision tree model to handle the reduced, yet most significant features. The tree-based approach is known for its efficacy in predictive analytics, providing an intuitive way to represent decisions and decision-making.
- Our model is designed to deliver enhanced prediction accuracy for unobserved test cases. By reducing overfitting through an optimized feature selection and utilizing FSVM, our model effectively handles both known and unforeseen cyber threats.

- Finally, we present a comprehensive evaluation of our model's performance by comparing it with traditional machine learning techniques. We provide a thorough comparative assessment of our model's efficiency. This comparison is rarely as extensive in existing literature.

Therefore, our research advances the field of cybersecurity by addressing several critical issues in IDS modeling and demonstrating an effective, comprehensive, and novel solution.

2 Related Work

Plethora of innovative approaches being proposed and implemented by various researchers. A selection of notable contributions is summarized as follows.

Verma et al. [13] were instrumental in developing ELNIDS to enhance security of IPv6 Routing Protocol in Lossy Networks. This system, which incorporates ensemble machine learning classifiers like Boosted Trees and RUSBoosted Trees, demonstrated its effectiveness in detecting routing attacks using the RPL-NIDDS17 dataset. Saharkhizan et al. [14] integrated a DL with LSTM modules with decision trees, achieving an impressive accuracy rate above 99% in detecting cyberattacks on IoT systems. Megantara et al. [15], in 2021, introduced a hybrid model blending supervised learning for feature selection with unsupervised learning for data reduction, showing high accuracy on the UNSW-NB15 dataset for various attack types. Abou El Houda et al. [16] developed a model to interpret IoT-related Intrusion Detection Systems (IDSs) decisions using deep neural networks and techniques like LIME and SHAP. Nguyen et al. [17] in 2022 unveiled Realguard, an DNN-based NIDS capable of identifying various cyberattacks in real-time with minimal computational resources. Muthanna et al. [18] proposed cuLSTMGRU for IoT, noted for its accuracy and efficiency. Sokkalingam et al. [19] recommended an ML technique, showing good performance on cybersecurity dataset. Xu et al. [20] introduced a new IDS based on a logarithmic autoencoder (LogAE) and XGBoost, surpassing many existing intrusion detection algorithms. Umair et al. [21] proposed framework named NAC using DL. Kaushik et al. [22] presented framework integrating IoT and deep learning. Nagarajan et al. [23] proposed a new anomaly detection method combining CNN with KF-based GMM for Cyber-Physical Systems (CPSs). Ramana et al. [24] emphasized the need for robust security in IoT networks, proposing an Ambient Approach IDS in Wireless Sensor Networks (WSNs) using RL-DQN. Finally, Devarajan et al. [25] discussed the significance of Fog computing in the expansion of IoT, presenting a BSFLVN for addressing privacy concerns and network congestion.

These studies collectively represent the dynamic and evolving nature of cybersecurity, with each offering unique solutions to the challenges posed by the ever-expanding digital landscape.

3 The Proposed Method

We propose the FSDT, for cyber security in this section. This involved a number of carefully designed steps, including investigating datasets, evaluating features ranking and relevance and generating the model that uses DT in its core. We will briefly go through each of these stages in order to accomplish our aim in the sections that follow.

3.1 Data Preprocessing

In this paper, the data preprocessing involves label encoding and normalization to the features included in the dataset.

1. Label encoding: In numerous machine learning and data science endeavors, data sets often encompass categorical or textual values. A prime example is a 'color' attribute that might include entries like red, orange, blue, and white. While a handful of specialized algorithms like CATBOOST can natively process categorical values, the majority of contemporary algorithms are optimized for numerical inputs to yield optimal outcomes. Consequently, a pivotal challenge for analysts is converting these categorical or textual inputs into numerical formats without losing their inherent significance in the context of the algorithm or model being developed. There are various methodologies for transforming categorical values into numerical ones, each with its own set of advantages and implications for the feature set. In this study, we have employed a commonly used technique known as label encoding. This approach is relatively straightforward, translating each unique value in a categorical column into a distinct numerical identifier. To illustrate, consider a dataset with a 'color names' column containing the values outlined in the first column of Table 1. The label encoded version of this column would be its numerical equivalent, as shown in the second column of the same table. This transformation facilitates the integration of categorical data into algorithms that are designed to work with numerical inputs, thereby broadening the applicability of these algorithms to a wider range of datasets.

Table 1: The numerical conversion for text dataset

Color name (categorical)	Color name (numerical)
Short	1
Very short	2
Tall	3
Very tall	4

2. Min-max normalization: Consider a dataset that includes two attributes: 'years' (y1) representing an individual's age, which varies between 0 to 80 years, and 'earnings' (y2) representing an individual's annual income, which ranges from 0 to $20,000. Given the disparity in their scales, where 'earnings' are approximately 1,000 times greater than 'years', the 'earnings' attribute would inherently exert a more significant influence on the results of any subsequent analyses, such as multivariate linear regression. This disproportionate influence occurs despite the fact that a higher value does not necessarily equate to being a more effective predictor. To address this imbalance and ensure a fair comparison between these two variables, normalization of the data is essential. This is given by

$$x' = \frac{x - x_{min}}{x_{max} - x_{\min}} \tag{1}$$

where x_{max} and x_{min} are the column max and min values.

Relations between the original data values are preserved through min-max normalization. Moreover, min-max reduces standard deviations as a result of this restricted range, which can also reduce the impact of outliers.

3.2 Feature Significance and Ranking

In the scope of this research, following the completion of data preparation and exploration, we embark on a critical phase: Calculating the significance score for each security feature. This process is integral to establishing a hierarchy of features based on their relative importance, enabling the selection of the most pertinent features and the exclusion of those deemed superfluous. Essentially, the concept of feature significance assigns a quantifiable score to each feature within a cybersecurity dataset, reflecting its discriminative power in differentiating between decision labels.

The underlying principle is to measure how each feature contributes to reducing impurity in the decision-making nodes, with the probability of reaching a particular node playing a pivotal role. The probability of a node is derived by proportioning the number of objects against those reaching that node. A higher probability score indicates a more reliable measure of feature significance. The values of feature significance range between 0 and 1, where 0 denotes no dependency, and 1 signifies a direct and complete dependency.

One widely used metric to evaluate the purity of a node is the Gini Index, a statistical tool frequently utilized in data mining. It essentially quantifies the likelihood of incorrect classification of a randomly chosen element based on the class distribution within the dataset. The Gini Index values vary from 0 to 1, where a value of 0 symbolizes perfect categorization purity (all elements pertain to a single class), and 1 indicates a completely random distribution of elements.

The Gini Index G of a node n for a binary split is given as follows:

$$G = 1 - \sum_{i=1}^{n} P_i^2, \tag{2}$$

3.3 Hybrid Fuzzy-SVM Classifier

In classical Support Vector Machine (SVM) models, every data point is assigned equal weight and the same punitive parameter is used. However, in numerous scenarios, not every sample point is equally crucial to the decision boundary. Particularly, outliers or noisy data points may not be accurately categorized into distinct groups. To address this challenge, the concept of Fuzzy Support Vector Machine (FSVM) was introduced. FSVM enhances classical SVM by assigning unique fuzzy membership. This modification allows for differential contributions from various sample points in the formation of the decision surface, acknowledging the varying significance of each point in the dataset.

Given a dataset $D = (X, Y, m)$, where $X = \{x, x_2, \ldots, x_n\}$ is the set of n training samples, $Y = \{d_1, d_2, \ldots, d_l\}$ is the set of class labels in the dataset and m is the fuzzy membership function. The following is a consideration for the classification quadratic optimization problem:

$$\min \frac{1}{2} \boldsymbol{W}^T \boldsymbol{W} + \mathrm{C} \sum_{i-1}^{l} m_i \zeta_i \tag{3}$$

where $\mathrm{x_i}\left(\mathrm{W^T W + b}\right) \geq 1 - \zeta, \zeta \geq 0, i = 1, 2, \ldots, l$, b is a bias term, $\boldsymbol{W}$ is a normal vector of the separating hyperplane, and C is a value that must be chosen in advance to balance the classification margin and the cost of misclassification mistake.

In the framework of Fuzzy Support Vector Machine (FSVM), the concept of weighted error measures plays a pivotal role. In this context, the slack variables, indicative of errors, are weighted differently across various data points. This differential weighting reflects the unique attitude or relevance of each data point towards a particular class. In essence, the weighting system within FSVM

allows individual input points to influence the learning of the decision surface to varying degrees. A higher weight assigned to a data point signifies its greater importance in the decision-making process, demanding more accurate classification. Conversely, a lower weight implies lesser significance, allowing for some degree of misclassification without substantially impacting the overall model. By strategically maximizing the margin while permitting some level of misclassification for less significant points, FSVM succeeds in identifying a more stable and robust hyperplane. This approach balances the need for accuracy with the practicalities of data variability, ensuring a more effective and adaptable classification model. By adding Lagrangian multipliers α, Eq. (3) is changed to double problem to solve the FSM optimum issue:

$$\max \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i x_j, \tag{4}$$

where $\sum_{j=1}^{N} \alpha_i y_i = 0$, $0 \leq \alpha_i \leq s_i C$, and $i = 1, 2, \ldots, N$.

The upper bound of the values is the only significant variation between the aforementioned statement and the typical SVM. The same manner that in the normal SVM, and may be retrieved by solving this dual issue in Eq. (4) for optimum.

The SVM hyperplane divides the set Y into two disjoint sets by

$$y_i [wx_i + b] \geq 1, \mathrm{i} = 1, 2, \ldots, \mathrm{n} \tag{5}$$

where w_i is the weight coefficient vector for the classification surface. Therefore, the problem is an optimization problem for

$$\eta_w = \frac{||w||^2}{2} \tag{6}$$

Using the Lagrangian method, this can be transformed for maximization of

$$\mathrm{L}\,(w) = \sum_{i=1}^{n} w_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{m} w_i w_j y_j^2 x_i^T x_i \tag{7}$$

This design already offers adaptability, but when considering the detection of new and emerging types of attacks, several strategies can be employed:

1. Continuous Learning and Model Update: The dynamic nature of cybersecurity threats necessitates the continuous updating of the model. As new attack patterns emerge, labeled data from these attacks can be incorporated into the training dataset D. This ongoing integration ensures that the model remains relevant and can detect new threats.
2. Feature Significance Adaptation: The algorithm already emphasizes the significance of features (Steps 2–4). As attack patterns evolve, the significance of features can shift. By recalculating and reevaluating feature significance regularly, the model can focus on the most pertinent features for current threats.
3. Incorporate Anomaly Detection: Instead of solely relying on a supervised approach, integrating unsupervised anomaly detection can be valuable. This can help identify unusual patterns, which might be indicative of new, previously unseen attacks. Anomalous data can then be further analyzed, and if validated as a new attack type, be incorporated into the training set.

Algorithm 1: Hybrid Fuzzy-SVM classifier

Input: D = (X, Y, m) //Dataset
Output: Decision label of the input object from the trained FVS
//Feature significance and reduction:
1 F = 0/ //Feature subset
2 Compute the feature significance as per Eq. (2)
3 Sort the features with respect to the significance
4 Compute the DT accuracy of the data Acc
5 do
6 Add the most significant feature to F
7 Compute the DT accuracy of F, Acc′
8 while Acc′ < Acc;
9 Divide the dataset F into Ft and Fe, training and testing, respectively.
10 Build the Fuzzy-SVM classifier
11 Calculate the confusion matrix
12 Compute the classification accuracy

4. Feedback Loop: Establish a feedback mechanism wherein predictions from the model are reviewed by security experts. False negatives (undetected attacks) can provide insight into where the model is lacking and guide necessary adaptations.
5. Ensemble Learning: Combine the predictions of the Hybrid Fuzzy-SVM classifier with other classifiers or models. The diverse perspectives from multiple models can enhance detection rates and minimize false negatives.
6. Regular Review of Training and Testing Data Split: Given the importance of keeping the model updated with the latest attack patterns, the split between training and testing data (Step 9) should be reviewed periodically. Over time, what was previously "new" becomes "known", and this shift should be reflected in the data split.
7. Expand Data Sources: Consider incorporating additional sources of data or alternative representations of data that might offer insights into new attack patterns. This can include network logs, system behaviors, or even external threat intelligence feeds.
8. Adaptive Thresholds: Depending on the nature of the new threats, the thresholds used in the Fuzzy-SVM or in the evaluation of the model's performance might need adjustments. Adaptive thresholds, which change based on the evolving threat landscape, can be more effective than static ones.

3.4 Handling Unlabeled Datasets

Using K-Means for intrusion detection involves a systematic process of feature extraction followed by the actual clustering. Here is a detailed breakdown:

Network traffic data is in matrix format, where each row is a data instance (packet or connection record) and each column represents a feature.

Given a data instance i, the feature vector given by

$$x_i = [f_1(i), f_2(i), \ldots, f_n(i)]$$

in which x_i represents the instance i feature vector.

Normalization is crucial, ensuring each feature has equal weight in the clustering process:

$$f_{j\text{norm}}(i) = \frac{f_j(i) - \mu_j}{\sigma_j}$$

in which

- μ_j represents the instances' mean of feature j.
- σ_j represents the instants' standard deviation of feature j.

K-Means Algorithm steps are as follows:

Begin by selecting K initial centroids, corresponding to the desired number of clusters.

$$C_k = [c_{k1}, c_{k2}, \ldots, c_{kn}], \quad k = 1, 2, \ldots, K$$

where:

- C_k is the centroid for cluster k.
- c_{kj} is the value of the j^{th} feature for centroid k.

Assignment Step: Assign each data instance x_i to the nearest centroid:

$$S_k = \{x_i : \|x_i - C_k\| \leq \|x_i - C_l\| \forall 1, 1 \neq k$$

where:

- S_k is the set of all data instances closest to centroid k.
- $|.|$ denotes the Euclidean distance.

Update Step: Calculate the new centroids for each cluster:

$$c_{kj} = \frac{1}{|S_k|} \sum_{x_i \in S_k} f_{j\text{norm}}(i)$$

Continue the assignment and update processes until the centroids stabilize or a predetermined number of iterations are completed.

3.5 Complexity Analysis

To analyze the time complexity of this Hybrid FuzzySVM classifier algorithm, we need to look at each of the main steps and determine their time complexity:

1. Compute the feature significance: The time complexity of this operation can depend on the method used to calculate the feature significance. If we assume that it uses the Gini impurity (as hinted by Eq. (2)), it generally requires iterating over all features for all examples, which results in a time complexity of $O(n \times m)$.
2. Sort the features with respect to the significance: Sorting has a time complexity of $O(m\log m)$.
3. Compute the DT accuracy of the data: Decision Tree predictions for all instances have a time complexity of $O(m\log m)$. This is under the assumption that the decision tree is balanced.
4. Iterative Addition of features to the subset: This loop can potentially run m times in the worst-case scenario. In each iteration, the algorithm computes the DT accuracy, which has a time complexity of $O(m\log m)$.
5. Splitting data into training and testing subsets: This operation has a time complexity of $O(n)$.

Adding up these complexities gives us a total time complexity for the algorithm. The dominating term here would be the training of the Hybrid Fuzzy-SVM classifier, which has a complexity of $O(n^2)$, and the iterative addition of features, which has a complexity of $O(m \times n\log m)$.

4 Experimental Work

For the purpose of validating the proposed model, we implemented it using Python programming language and conducted the experiments on a PC equipped with Windows 11, operating with 3.1 GHz, and supported by 8 GB of RAM. The experimental evaluation was structured around two principal objectives. The initial goal was to compare the effectiveness of our model against current algorithms in same field.

This comparative analysis aimed to assess the relative efficacy of our model in terms of its capability to handle multidimensional data and its overall algorithmic efficiency. The second objective was to illustrate the proficiency of our model, particularly in its ability to effectively reduce the dimensionality of Multivariate Data (MVD) while simultaneously preserving, or even enhancing, the accuracy of classification outcomes. This aspect of the experiment was crucial to demonstrate the model's utility in handling complex datasets without compromising on the quality of the analytical results it yields.

4.1 Dataset

In the critical domain of cybersecurity, developing a data-centric intrusion detection model demands an in-depth comprehension of raw security data's characteristics and trends. Our study leveraged an intrusion dataset sourced from Kaggle, a preeminent machine learning and data science platform. This dataset integrates both typical and atypical class variables, comprising 41 diverse security attributes. It includes three qualitative attributes, like protocol type and response latency, and 38 quantitative ones, encompassing metrics such as session duration, login status, server error rates, host count, and data transmission volumes. A detailed presentation of these features and their types is available in Table 1. The dataset, featuring over 25,000 instances, was curated from simulated cyberattacks within a military network setting, engineered to replicate a standard US Air Force local area network. This simulation, inclusive of comprehensive TCP/IP dump data, was subjected to various cyberattacks to foster a genuine cybersecurity environment. Fig. 1 illustrates the uneven data distribution across features such as 'duration' and 'destination bytes.' Before initiating our machine learning-based intrusion detection model, we undertook extensive preparation and refinement of the raw dataset, focusing on the effective organization and prioritization of these security features. Additionally, our study incorporated the UNSW-NB15 dataset.

4.2 Classification

In the realm of machine learning, the K-fold cross-validation technique serves as a pivotal method for evaluating model performance. Each fold is subsequently used once as a testing set while K-1 folds collectively serve for training phase. This process ensures that each data sample is used for training multiple times (K-1 times) but tested only once. In this research, we implement the ten-fold cross-validation approach, applying it specifically to our samples.

The dataset is then subject to the 10-fold cross-validation process, where crucial feature selection occurs during each fold of training. The subsequent testing phase employs an unknown dataset to assess the efficacy of the trained classifier model. The performance of this testing is quantitatively evaluated using various performance metrics. A detailed representation of these results is presented in Table 2, which includes the confusion matrix.

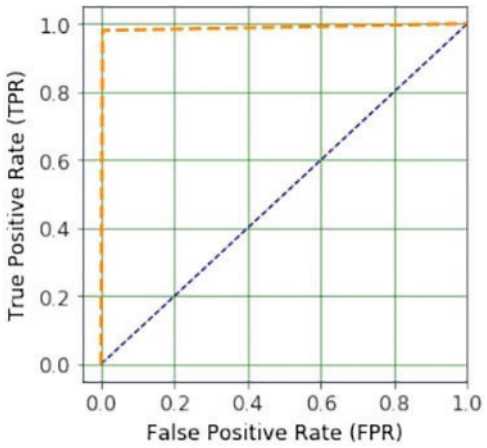

Figure 1: The relation between TPR and FPR; resulting in ROC curve

Table 2: Classification comparative analysis for our proposed model and recent competitors

Method	Accuracy	Sensitivity	Specifcity	Ref.	Result status
Hybrid ML	0.92	0.80	0.75	[15]	Copied
IntruDTree	0.96	0.82	0.79	[26]	Generated
ANN	0.98	0.91	0.90	[19]	Generated
SVM-HHO-PSO	0.96	0.88	0.85	[27]	Copied/Generated
Fuzzy-SVM-DT	**0.99**	**0.93**	**0.91**	–	–

The decision to use this particular classifier is twofold. The dataset generated by the proposed method is optimally suited to support the characteristics of the decision tree method. It contains a minimal yet essential collection of features and data, exhibits a heterogeneity in the data of each feature, and is devoid of any duplicate or redundant data. The newly produced dataset thus aligns seamlessly with the operational dynamics of the decision tree classifier, facilitating an efficient classification process.

Accuracy, sensitivity, specificity, false alarm rate, and computing time are the parameters used as per Eqs. (8)–(10). The computing time is calculated by calculating the calculation time from the first to final technique.

1. Confusion matrix: It is made up of two projected classes and two actual classes.
2. Accuracy: The formula in Eq. (8) can be used to calculate the accuracy value.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP} \tag{8}$$

3. Sensitivity: The formula in Eq. (9) can be used to compute the sensitivity value.

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{9}$$

4. Specificity: The formula in Eq. (10) is used to compute it.

$$Specifcity = \frac{TN}{TN + FP} \tag{10}$$

To ensure unbiased outcomes, models were trained using the selected features across ten trial runs, with the results' average serving to mitigate potential biases. Additionally, by conducting nearly five iterations of the algorithms, any feature selected more than eight times was deemed significant for inclusion. Table 3 presents a comparative analysis of these models. This evaluation suggests that integrating the distinct features of two algorithms into a hybridized form leads to enhanced performance. This synergistic approach, which combines the unique strengths of individual algorithms, offers a more robust and effective solution.

Table 3: Performance result of our FSVM

Class	Accuracy	Recall	F1-score	Precision
Safe	0.98	0.98	0.96	0.97
Up normal	0.99	0.97	0.97	0.98

The IDS model performance outperforms conventional algorithms. An additional facet of this investigation focuses on ascertaining the efficacy of the model across various decision classes. To this end, the dataset was partitioned such that 80% was chosen for training. The outcomes of this experimental setup are meticulously documented in Table 4. Upon scrutinizing Table 4, it becomes evident that our model demonstrates commendable performance across each decision class. This is evident through the four metrics, collectively underscoring the model's proficiency in accurately predicting anomalies or intrusions. Furthermore, the robustness of the model is visually represented in Fig. 1 through an ROC (Receiver Operating Characteristic) curve. This curve juxtaposes the model's TPR against its FPR, offering a holistic view of its performance. Fig. 1 distinctly illustrates that the model achieves a high TPR, signifying a substantial proportion of correctly categorized cases, which approaches the optimal value of 1. Conversely, the FPR remains low, indicating a minimal incidence of incorrectly classified cases. This balance between high TPR and low FPR manifests the model's adeptness in effectively discerning between anomalies and normal classes based on their patterns within the security dataset. The cumulative experimental results, as presented in Table 3 and Fig. 1, unequivocally affirm that our model not only demonstrates significant accuracy in detecting anomalies and normal classes but also exhibits a high degree of reliability in yielding substantial results for unseen test cases. This comprehensive evaluation showcases the model's utility and effectiveness in the realm of cybersecurity, particularly in intrusion detection scenarios.

Table 4: The features with the corresponding significance and rank

#	Feature	Significance	Rank
1	duration	0.580	4
2	protocol_type	0.448	17
3	service	0.045	37
4	flag	0.236	28

(Continued)

Table 4 (continued)

#	Feature	Significance	Rank
5	src_bytes	0.492	9
6	dst_bytes	0.580	5
7	land	0.285	21
8	wrong_fragment	0.117	33
9	urgent	0.281	22
10	hot	0.463	14
11	num_failed_logins	0.457	16
12	logged_in	0.280	23
13	num_compromised	0.266	25
14	root_shell	0.077	35
15	su_attempted	0.033	39
16	num_root	0.446	18
17	num_file_creations	0.668	1
18	num_shells	0.192	29
19	num_access_files	0.158	30
20	num_outbound_cmds	0.237	27
21	is_host_login	0.642	2
22	is_guest_login	0.305	20
23	count	0.139	32
24	srv_count	0.056	36
25	serror_rate	0.262	26
26	srv_serror_rate	0.568	6
27	rerror_rate	0.463	15
28	srv_rerror_rate	0.035	38
29	same_srv_rate	0.481	10
30	diff_srv_rate	0.480	11
31	srv_diff_host_rate	0.276	24
32	dst_host_count	0.543	7
33	dst_host_srv_count	0.440	19
34	dst_host_same_srv_rate	0.585	3
35	dst_host_diff_srv_rate	0.087	34
36	dst_host_same_src_port_rate	0.519	8
37	dst_host_srv_diff_host_rate	0.157	31
38	dst_host_serror_rate	0.020	40
39	dst_host_srv_serror_rate	0.010	41
40	dst_host_rerror_rate	0.464	13
41	dst_host_srv_rerror_rate	0.472	1

4.3 Experiment 2: Feature Importance

This segment of the experiment is dedicated to a meticulous examination of feature importance, with the overarching objective of developing a comprehensive, data-driven security model while

concurrently pruning superfluous features from the security dataset. Central to this endeavor is the evaluation of each feature's significance and respective importance score within the dataset.

The methodology employed to ascertain feature importance involves the use of the Fuzzy-SVM-DT classifier. This approach entails assessing each feature individually to determine its contribution to classification accuracy, specifically in discriminating between target decisions. A higher accuracy score for a feature signifies its greater importance in the classification process. In practical terms, this involved conducting 80 separate classification runs, each incorporating one conditional feature alongside the decision feature.

The outcomes of this experiment are represented in Table 4, which elucidates the relevance score calculations for various variables within the security dataset. A notable observation from this is that not all features within each dataset exhibit uniform significance scores. These scores can vary significantly from one feature to another, contingent upon their respective influence on the target class.

This means revealing a variance in computed significance among the features in each dataset. For instance, the feature 'num file creations' emerges as the most impactful in the dataset, boasting a significance score of 0.668. Conversely, the feature 'dst host srv error rate' registers a minimal impact, with its significance score nearing zero. Additionally, the last column of Table 4 systematically presents the features in order of their impact, providing a clear hierarchy of feature importance based on their respective scores. This detailed analysis not only highlights the differential impact of each feature but also guides the prioritization of features in the context of the overall security model development.

4.4 Experiment 3: Model Superiority

This experimental endeavor is devoted to substantiating the preeminence of our model in comparison to conventional ML algorithms. Primary objective is twofold: Firstly, validation for efficacy of the Decision Tree (DT) as a Feature Selection (FS) tool, and secondly, to establish the superiority of the hybrid fuzzy-Support Vector Machine (SVM) in classifying security data.

Table 5 presents a comparative analysis between the DT and several renowned baseline FS techniques, including Principal Component Analysis (PCA), Salp Swarm Optimization (SSO), and Ant Colony Optimization (ACO). A perusal of this reveals that while DT is adept at selecting the minimal number of features, it exhibits a longer running time compared to its counterparts.

Table 5: A comparison of the number of selected features for DT and three competitors

Tool	# of selected features	Time (s)
PCA	14	0.09
SSO	12	**0.07**
ACO	12	0.08
DT	**10**	1.02

To elaborate on Hybrid Fuzzy-SVM classifier performance, we engaged in a comparative study involving several well-known baseline classifiers, namely LR, SVM, and DT. To ensure a fair and comprehensive comparison, we computed the end outcomes for each model using the same security

datasets. This approach allowed for a direct and objective evaluation of the performance of each algorithm, thereby facilitating a clearer understanding of the relative strengths and capabilities of our proposed Hybrid Fuzzy-SVM classifier in the context of cybersecurity applications.

5 Conclusions

This work developed a ML for intrusion detection security model. Initial stage focused on establishing a generalized intrusion detection model centered on chosen critical features, highlighting the varying significance of different security elements. This was achieved through the integration of fuzzy sets with Support Vector Machines (SVM), aiming to enhance the efficiency of the security model. By focusing on fewer yet significant features, we managed to improve the cost while bolstering the accuracy of test case predictions.

To elaborate on our proposed model efficiency, we conducted a set of experimental tests employing different cybersecurity datasets.

Comparative analysis of our model's performance against a range of established classical machine learning techniques revealed its superiority. Notably, our model demonstrated a faster convergence rate while substantially reducing the feature space. When these refined features were tested on simple classifiers, we observed an increase in accuracy, further substantiating the validity and relevance of the selected features.

Looking ahead, future research endeavors could extend the evaluation of our model's efficacy by harnessing larger datasets encompassing a wider array of security feature dimensions, particularly in the context of IoT security services. Such studies would contribute significantly to understanding the model's application-level efficacy within the broader domain of cybersecurity. This exploration will not only provide insights into the scalability and adaptability of the model but also offer valuable perspectives on its potential applications in real-world cybersecurity scenarios.

Acknowledgement: Not applicable.

Funding Statement: The authors received no funding for this study.

Author Contributions: Fatimah Mudhhi Alanazi: Coding, Carrying out experiment, Data manipulation, Formalization, and Writing initial draft. **Bothina Abdelmeneem Elsobky:** Supervision, Conceptualization, Formalization, Reviewing results and Editing the final version. **Shaimaa Aly Elmorsy:** Supervision, Conceptualization, Formalization, Reviewing results and Editing the final version.

Availability of Data and Materials: Code and data are available upon request.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] M. Paul, L. Maglaras, M. A. Ferrag, and I. AlMomani, "Digitization of healthcare sector: A study on privacy and security concerns," *ICT Exp.*, vol. 9, no. 4, pp. 571–588, 2023. doi: 10.1016/j.icte.2023.02.007.

[2] A. Robinson, C. Corcoran, and J. Waldo, "New risks in ransomware: Supply chain attacks and cryptocurrency," *Sci. Tech., Public Policy Program Reports*, 2022.

[3] T. Mazhar *et al.*, "Analysis of cyber security attacks and its solutions for the smart grid using machine learning and blockchain methods," *Future Internet*, vol. 15, no. 2, pp. 83, 2023. doi: 10.3390/fi15020083.

[4] Z. Turk, B. G. de Soto, B. R. Mantha, A. Maciel, and A. Georgescu, "A systemic framework for addressing cybersecurity in construction," *Autom. Constr.*, vol. 133, pp. 103988, 2022. doi: 10.1016/j.autcon.2021.103988.
[5] A. Hamed and H. Nassar, "Efficient feature selection for inconsistent heterogeneous information systems based on a grey wolf optimizer and rough set theory," *Soft Comput.*, vol. 25, no. 24, pp. 15115–15130, 2021. doi: 10.1007/s00500-021-06375-z.
[6] K. Razikin and B. Soewito, "Cybersecurity decision support model to designing information technology security system based on risk analysis and cybersecurity framework," *Egypt. Inform. J.*, vol. 23, no. 3, pp. 383–404, 2022. doi: 10.1016/j.eij.2022.03.001.
[7] O. A. Salem, F. Liu, Y. P. P. Chen, A. Hamed, and X. Chen, "Effective fuzzy joint mutual information feature selection based on uncertainty region for classification problem," *Knowl.-Based Syst.*, vol. 257, pp. 109885, 2022. doi: 10.1016/j.knosys.2022.109885.
[8] A. Hamed, M. Tahoun, and H. Nassar, "KNNHI: Resilient KNN algorithm for heterogeneous incomplete data classification and k identification using rough set theory," *J. Inf. Sci.*, vol. 49, no. 6, pp. 1631–1655, 2023. doi: 10.1177/01655515211069539.
[9] T. H. Aldhyani and H. Alkahtani, "Attacks to automatous vehicles: A deep learning algorithm for cybersecurity," *Sensors*, vol. 22, no. 1, pp. 360, 2022. doi: 10.3390/s22010360.
[10] A. Garcia-Perez, J. G. Cegarra-Navarro, M. P. Sallos, E. Martinez-Caro, and A. Chin-naswamy, "Resilience in healthcare systems: Cyber security and digital transformation," *Technovat.*, vol. 121, pp. 102583, 2023.
[11] Z. Lai, X. Chen, J. Zhang, H. Kong, and J. Wen, "Maximal margin support vector machine for feature representation and classification," *IEEE Trans. Cybern.*, vol. 53, no. 10, pp. 6700–6713, 2023. doi: 10.1109/tcyb.2022.3232800.
[12] L. D. Sharma *et al.*, "Evolutionary inspired approach for mental stress detection using EEG signal," *Expert. Syst. Appl.*, vol. 197, pp. 116634, 2022. doi: 10.1016/j.eswa.2022.116634.
[13] A. Verma and V. Ranga, "ELNIDS: Ensemble learning based network intrusion detection system for RPL based internet of things," in *2019 4th Int. Conf. Internet of Things: Smart Innov. Usage (IoT-SIU)*, Ghaziabad, India, 2019, pp. 1–6.
[14] M. Saharkhizan, A. Azmoodeh, A. Dehghantanha, K. K. R. Choo, and R. M. Parizi, "An ensemble of deep recurrent neural networks for detecting IoT cyber attacks using network traffic," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 8852–8859, 2020. doi: 10.1109/jiot.2020.2996425.
[15] A. A. Megantara and T. Ahmad, "A hybrid machine learning method for increasing the performance of network intrusion detection systems," *J. Big Data*, vol. 8, no. 1, pp. 1–19, 2021.
[16] Z. Abou El Houda, B. Brik, and L. Khoukhi, "Why should I trust your IDS?": An explainable deep learning framework for intrusion detection systems in internet of things networks," *IEEE Open J. Commun. Soc.*, vol. 3, pp. 1164–1176, 2022. doi: 10.1109/ojcoms.2022.3188750.
[17] X. H. Nguyen, X. D. Nguyen, H. H. Huynha, and K. H. Le, "Realguard: A lightweight network intrusion detection system for IoT gateways," *Sensors*, vol. 22, no. 2, pp. 432, 2022. doi: 10.3390/s22020432.
[18] M. S. A. Muthanna, R. Alkanhel, A. Muthanna, A. Rafiq, and W. A. M. Abdullah, "Towards SDN-enabled, intelligent intrusion detection system for internet of things (IoT)," *IEEE Access*, vol. 10, pp. 22756–22768, 2022. doi: 10.1109/access.2022.3153716.
[19] S. Sokkalingam and R. Ramakrishnan, "An intelligent intrusion detection system for distributed denial of service attacks: A support vector machine with hybrid optimization algorithm based approach," *Concurr. Comput.: Pract. Exp.*, vol. 34, no. 27, pp. e7334, 2022. doi: 10.1002/cpe.7334.
[20] W. Xu and Y. Fan, "Intrusion detection systems based on logarithmic autoencoder and XGBoost," *Secur. Commun. Netw.*, vol. 2022, pp. 1–8, Apr. 2022.
[21] M. B. Umair, Z. Iqbal, F. Z. Khan, M. A. Khan, and S. Kadry, "A deep learning based method for network application classification in software-defined IoT," *Int. J. Uncert., Fuzz. Knowl.-Based Syst.*, vol. 30, no. 3, pp. 463–477, 2022. doi: 10.1142/s0218488522400165.
[22] H. Kaushik *et al.*, "Screening of COVID-19 patients using deep learning and IoT framework," *Comput., Mater. Contin.*, vol. 69, no. 3, pp. 3459–3475, 2021. doi: 10.32604/cmc.2021.017337.

[23] S. M. Nagarajan, G. G. Deverajan, A. K. Bashir, R. P. Mahapatra, and M. S. Al-Numay, "IADF-CPS: Intelligent anomaly detection framework towards cyber physical systems," *Comput. Commun.*, vol. 188, pp. 81–89, 2022. doi: 10.1016/j.comcom.2022.02.022.

[24] T. Ramana, M. Thirunavukkarasan, A. S. Mohammed, G. G. Devarajan, and S. M. Nagarajan, "Ambient intelligence approach: Internet of things based decision performance analysis for intrusion detection," *Comput. Commun.*, vol. 195, pp. 315–322, 2022. doi: 10.1016/j.comcom.2022.09.007.

[25] G. G. Devarajan, M. Thirunnavukkarasan, S. I. Amanullah, T. Vignesh, and A. Sivara-Man, "An integrated security approach for vehicular networks in smart cities," *Trans. Emerg. Telecomm. Technol.*, vol. 34, no. 11, pp. e4757, 2023.

[26] I. H. Sarker, Y. B. Abushark, F. Alsolami, and A. I. Khan, "IntruDTree: A machine learning based cyber security intrusion detection model," *Symm.*, vol. 12, no. 5, pp. 754, 2020. doi: 10.3390/sym12050754.

[27] A. Aleesa, M. Younis, A. A. Mohammed, and N. Sahar, "Deep-intrusion detection system with enhanced UNSW-NB15 dataset based on deep learning techniques," *J. Eng. Sci. Tech.*, vol. 16, no. 1, pp. 711–727, 2021.