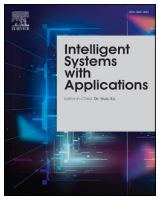
Intelligent Systems with Applications 25 (2025) 200465

Contents lists available at ScienceDirect

Intelligent Systems with Applications

journal homepage: www.journals.elsevier.com/intelligent-systems-with-applications



Detecting unknown intrusions from large heterogeneous data through ensemble learning

Farah Jemili *, Khaled Jouini, Ouajdi Korbaa

Universite de Sousse, ISITCom, MARS Research Laboratory, LR17ES05, Hammam Sousse, 4011, Tunisia

ARTICLE INFO

Keywords:
Big heterogeneous data
Intrusion detection
Data fusion

ABSTRACT

The rapid expansion of data volumes, technological advancements, and the emergence of the Internet of Things (IoT) have heightened concerns regarding the detection of unknown intrusions based on singular sources of network traffic. This progression has led to the generation of vast and diverse datasets originating from various sources including IoT devices, web applications, and web services. Effectively discerning attacks within such a heterogeneous network traffic landscape necessitates the identification of underlying security behaviors, essential for developing an efficient analysis information system.

This paper aims to establish a comprehensive framework for network intrusion detection. The proposed methodology involves the synthesis of network features into a universal security database through the utilization of Term Frequency-Inverse Document Frequency Terms (TF-IDF) and semantic Cosine similarity. By amalgamating a diverse array of data flows, a set of universal features is generated, facilitating storage within the newly devised universal representation. Subsequently, Principal Component Analysis (PCA) is employed to reduce the dimensionality of the extensive universal security database while preserving essential information. Leveraging Ensemble Learning, a novel method is introduced for the detection of unknown attacks.

The efficacy of the developed database is evaluated using various Machine Learning algorithms, including Naïve Bayes, K-Nearest Neighbor, Logistic Regression, Decision Tree, and Random Forest. Furthermore, Ensemble Learning methods are assessed under two distinct scenarios. Experimental findings, conducted on datasets such as CICIDS 2017, NSL-KDD, and UNSW, demonstrate the universality, versatility, and effectiveness of the proposed approach, particularly in accommodating datasets with diverse structures.

1. Introduction

Intrusion Detection Systems (IDS) have long been essential tools for identifying potential attacks within computer networks and systems. Recently, the integration of Machine Learning (ML) algorithms has further enhanced IDS by enabling more sophisticated data analysis and classification for intrusion detection. In the context of ML, IDS function as classifiers, categorizing network traffic data and predicting potential attacks based on this classification (Patel, Taghavi, Bakhtiyari & Júnior, 2013).

However, IDS face increasing challenges due to the exponential growth of data and the proliferation of Internet of Things (IoT) devices, which generate diverse and complex data. While recent advancements in big data have led to the development of distributed architectures for IDS (Intrata, Grif & Dostovalov, 2021; Othman et al., 2018; Peng, Leung & Huang, 2018), the dynamic and heterogeneous nature of cyberspace complicates the detection of unknown intrusions across varied network data sources. Given the prevalence of interconnected computer systems in real-world applications, the traditional approach of creating separate IDS for each network type is increasingly impractical.

1.1. Motivation and contributions

- Motivation:
  - The rise in volume and diversity of network traffic, particularly with IoT, has amplified the challenge of detecting unknown attacks in real-time.
  - Conventional IDS struggle to process heterogeneous data from multiple sources, necessitating a universal framework for intrusion detection.
  - An adaptable, efficient, and scalable IDS approach is critical for real-world environments where threats constantly evolve.

* Corresponding author.
E-mail address: jmili_farah@yahoo.fr (F. Jemili).

https://doi.org/10.1016/j.iswa.2024.200465
Received 26 April 2024; Received in revised form 14 November 2024; Accepted 5 December 2024
Available online 9 December 2024

- Contributions of This Study:
  - Universal Security Database: We propose a comprehensive database that integrates features from diverse network types, creating a versatile foundation for IDS.
  - Universal Feature Representation: By leveraging Term Frequency-Inverse Document Frequency (TF-IDF) and Cosine similarity, our framework effectively fuses heterogeneous data sources.
  - Dimensionality Reduction: Principal Component Analysis (PCA) is applied to streamline the feature space, ensuring both efficiency and preservation of essential information.
  - Ensemble Learning Method: A unique ensemble method, combining parallel and sequential approaches, is introduced to enhance the detection accuracy of unknown attacks.
  - Comprehensive Evaluation: The proposed framework is rigorously tested on multiple scenarios using CICIDS 2017, NSL-KDD, and UNSW-NB15 datasets, demonstrating its effectiveness and adaptability.

1.2. Paper structure

The remainder of this paper is organized as follows: Section 2 reviews the current state of IDS and the ongoing challenges. Section 3 discusses network traffic sources and introduces a novel framework for data representation. Section 4 details the proposed methodology. Section 5 presents evaluation results, and Section 6 provides conclusions from the study.

2. Related works

Traditional machine learning (ML) tools, while effective in certain contexts, have struggled to keep pace with the ever-expanding diversity and sheer volume of network traffic data. As a result, researchers have increasingly turned to big data technologies as a means of enhancing intrusion detection capabilities. To tackle the challenge posed by the sheer magnitude of data, innovative approaches leveraging distributed architectures, distributed storage systems, and parallelization paradigms have been developed.

Peng, Leung and Huang (2018) introduced a method combining Mini Batch K-means with Principal Component Analysis (PCA) to differentiate between normal and malicious behavior in network traffic. This approach harnesses the power of distributed computing for scalability and efficiency, but it focuses on a single data source and does not explore ensemble learning methods for integrating multiple datasets. Our proposed approach advances this work by integrating data from NSL-KDD, UNSW, and CICIDS datasets and employing a novel ensemble learning strategy that combines both parallel and sequential methods for enhanced intrusion detection.

Elayni et al.'s survey (Elayni, Jemili, Korbaa & Soulaimen, 2019) provides valuable insights into the IDS process, spanning from data collection to detection methods, and highlights the importance of big data storage and analysis techniques. However, this survey does not address the specific challenges of data fusion across heterogeneous sources or the application of ensemble learning techniques. Our work builds upon these foundational concepts by proposing a comprehensive approach that integrates advanced feature extraction techniques with a unique ensemble learning framework designed to handle diverse data environments.

Distributed frameworks such as Hadoop and Spark have emerged as pivotal platforms for managing large volumes of network traffic data (Intrata, Grif & Dostovalov, 2021; Othman et al., 2018; Patel, Taghavi, Bakhtiyari & Júnior, 2013; Peng, Leung & Huang, 2018; Zuech, Khoshgoftaar & Wald, 2015). These frameworks are essential for efficient data processing but often focus on individual datasets rather than exploring techniques for fusing diverse data sources. Our approach extends these frameworks by applying a novel ensemble learning strategy that integrates multiple data sources and methods to improve detection performance.

Zhou et al. (2018) introduced a taxonomy that addresses the heterogeneity of network data, including host logs, application logs, and wireless network traffic. While this taxonomy provides a framework for understanding data diversity, it does not explore the integration of these diverse data types through advanced feature extraction and ensemble learning techniques. Our study contributes to this area by developing methods for effective data fusion and leveraging ensemble learning to address the complexities of heterogeneous network environments.

Recent studies such as ARGUS IDS and CIC Flow Meter (2022) have proposed advanced methods for feature extraction and classification, including multi-dimensional feature fusion and deep neural network-based reconstruction techniques. While these methods address feature extraction challenges, our work introduces a novel combination of data fusion techniques and ensemble learning approaches that significantly enhances the effectiveness of intrusion detection systems across a range of datasets and environments, (Table 1).

To further enhance intrusion detection in cybersecurity, the works in Arasteh et al. (2024), Arasteh, Bouyer, Sefati and Craciunescu (2024), and Danesh, Karimi and Arasteh (2024) offer noteworthy contributions focused on specialized threat detection. The paper (Arasteh, Bouyer, Sefati & Craciunescu, 2024) introduces a hybrid approach to SQL injection (SQLi) detection using the Binary Olympiad Optimizer (BOO) for feature selection combined with classification algorithms. This method efficiently reduces computational complexity and achieves high detection accuracy by narrowing down to relevant features for SQLi detection. Similarly, (Arasteh et al., 2024) addresses SQLi detection by employing the Binary Gray Wolf Optimizer (BGWO) alongside machine learning classifiers, constructing a dataset with essential SQLi indicators to improve model accuracy and reduce computational demands. While both approaches offer effective feature selection techniques that enhance traditional SQLi detection methods, they are limited to specific types of attacks within SQLi threats, and neither work expands to multiple types of network traffic data or addresses the scalability challenges associated with big data environments.

In Danesh, Karimi and Arasteh (2024), CMShark is introduced as a network-based, infrastructure-independent method for detecting crypto-jacking attacks. The study uses a hybrid approach, combining machine learning classification, IP blacklisting, and payload inspection, providing robust crypto-jacking detection through packet size classification, known IP addresses, and keyword-based payload inspection. Although CMShark's multi-faceted design improves crypto-jacking detection accuracy, its application is limited to network edge deployments and specific attack types, making it less versatile for large-scale, heterogeneous network environments.

While (Arasteh et al., 2024; Arasteh, Bouyer, Sefati & Craciunescu, 2024), and (Danesh, Karimi & Arasteh, 2024) introduce innovative and effective detection methods for SQLi and crypto-jacking threats, they primarily focus on isolated attack vectors, relying on specific feature selection optimizers and detection techniques that are not optimized for the integration of diverse data sources or scalable, distributed architectures. In contrast, our approach incorporates an ensemble learning framework that unites data from the NSL-KDD, UNSW, and CICIDS datasets, accommodating a broader spectrum of attacks. This ensemble learning strategy not only includes both parallel and sequential methods for enhanced intrusion detection but also addresses data fusion from diverse sources, surpassing the single-source limitations of the discussed studies. By leveraging distributed computing frameworks such as Hadoop and Spark, our approach achieves the scalability necessary to handle big data environments, a critical factor overlooked in Arasteh et al. (2024), Arasteh, Bouyer, Sefati and Craciunescu (2024), Danesh, Karimi and Arasteh (2024).

In summary, our contribution significantly advances the field by addressing the challenges of diverse data integration, scalability, and multi-model ensemble learning, delivering a more comprehensive and robust intrusion detection solution adaptable to complex and large-scale

Table 1
Related work.

Study	Key Contributions	Limitations	Our Novelty and Differentiation
Peng, Leung and Huang (2018)	Combines Mini Batch K-means with PCA for network traffic analysis	Focuses on a single data source; lacks ensemble methods	Integrates multiple datasets (NSL-KDD, UNSW, CICIDS) with ensemble learning (parallel and sequential) to improve detection accuracy
Elayni, Jemili, Korbaa and Soulaimen (2019)	Survey on IDS process from data collection to detection methods	Does not address data fusion or ensemble learning techniques	Proposes a comprehensive approach integrating data fusion and ensemble learning for diverse data environments
Distributed Frameworks (Patel, Taghavi, Bakhtiyari & Júnior, 2013; Intrata, Grif & Dostovalov, 2021; Peng, Leung & Huang, 2018; Othman et al., 2018; Zuech, Khoshgoftaar & Wald, 2015)	Utilizes Hadoop and Spark for efficient data processing	Often limited to individual datasets; lacks data fusion focus	Extends these frameworks with a novel ensemble learning strategy that integrates multiple data sources
Zhou et al. (2018)	Introduces taxonomy for network data heterogeneity	Does not explore data integration or advanced feature extraction techniques	Develops methods for effective data fusion and utilizes ensemble learning to handle heterogeneous network environments
ARGUS IDS	Proposes multi-dimensional feature fusion and stacking ensemble	Primarily focuses on feature extraction; limited data sources	Combines advanced feature extraction techniques with ensemble learning, enhancing effectiveness across various datasets
CIC Flow Meter	Novel reconstruction method using deep neural networks for feature extraction	Focuses on feature extraction and classification	Introduces a unique combination of data fusion and ensemble learning to significantly enhance intrusion detection capabilities

network environments.

Summary of Our Contributions:

- **Data Fusion:** Integrates multiple heterogeneous datasets (NSL-KDD, UNSW, CICIDS) to improve robustness and generalizability.
- **Ensemble Learning:** Utilizes a combination of parallel and sequential ensemble methods to enhance detection accuracy and performance.
- **Feature Extraction:** Employs innovative techniques like TF-IDF and Cosine similarity for effective feature extraction, addressing the unstructured nature of intrusion data.
- **Comprehensive Evaluation:** Conducts extensive experiments across diverse datasets to demonstrate the effectiveness and scalability of the proposed approach.

The proposed method addresses several key limitations commonly found in existing unknown intrusion detectors:

1. **Inability to Handle Heterogeneous Data**: Traditional detectors often focus on single-source or homogenous datasets, which limits their applicability in real-world scenarios where data sources are diverse. Our method overcomes this by creating a universal security database that integrates features from multiple, heterogeneous datasets (CICIDS 2017, NSL-KDD, UNSW-NB15).
2. **Limited Detection of Unknown Attacks**: Many intrusion detection systems rely on predefined patterns or signatures, making them less effective for unknown intrusions. By applying ensemble learning and a universal feature representation, our method enhances the detection capability for novel, previously unseen attacks.
3. **High Computational Requirements**: Existing systems sometimes struggle with large, high-dimensional data, which can slow down detection and reduce scalability. Our approach addresses this through Principal Component Analysis (PCA) for dimensionality reduction, which optimizes processing time and computational efficiency.
4. **Scalability and Adaptability Issues**: IDS often need extensive reconfiguration to adapt to different datasets or network environments. The proposed framework is designed to be adaptable across various data environments, increasing the scalability of the IDS.

3. Methodology formulation

3.1. Network data representation

This section delves into the intricacies of communication between network systems, as defined by the Open System Interconnection (OSI) reference model, which conceptualizes data transmission across seven layers from the physical layer to the application layer (ARGUS, IDS; CIC Flow, Meter). Data transmission occurs via packets, which serve as the fundamental unit of communication in network traffic (see Fig. 1). Each packet comprises a payload, representing the actual data being transmitted, along with one or more headers containing metadata such as source and destination IP addresses, service details, and protocols. Notably, headers play a crucial role in computer networking by providing essential information for accurate data transmission. For instance, packets conforming to the Telnet Network protocol (telnet) adhere to the TCP/IP standard and encompass a media access control (MAC) header, an internet protocol (IP) header, a transmission control protocol (TCP) header, a telnet header, and a telnet payload.

Intrusion Detection Systems (IDS) leverage headers from packets across various protocols to extract features specific to each type of network traffic data. For instance, tools like CICFlow Meter (CICIDS, 2017) are utilized to extract 76 distinct features from network traffic within datasets like CICIDS 2017.

However, two significant drawbacks are associated with the network traffic data utilized in IDS. Firstly, the detection focus is typically confined to individual packets or isolated data flows. While a single data flow may consist of multiple series of data packets from the same network type, real-world network traffic is characterized by heterogeneous data flows with diverse structural compositions. While heterogeneous data flow collection is recommended for IDS, it presents challenges for Machine Learning (ML) algorithms.

Secondly, metadata poses another challenge, as features are extracted from headers following universal network standards such as the OSI reference model and TCP/IP standard. However, proposed datasets often exhibit ambiguous feature names with differing notations, despite being derived from standard packet headers.

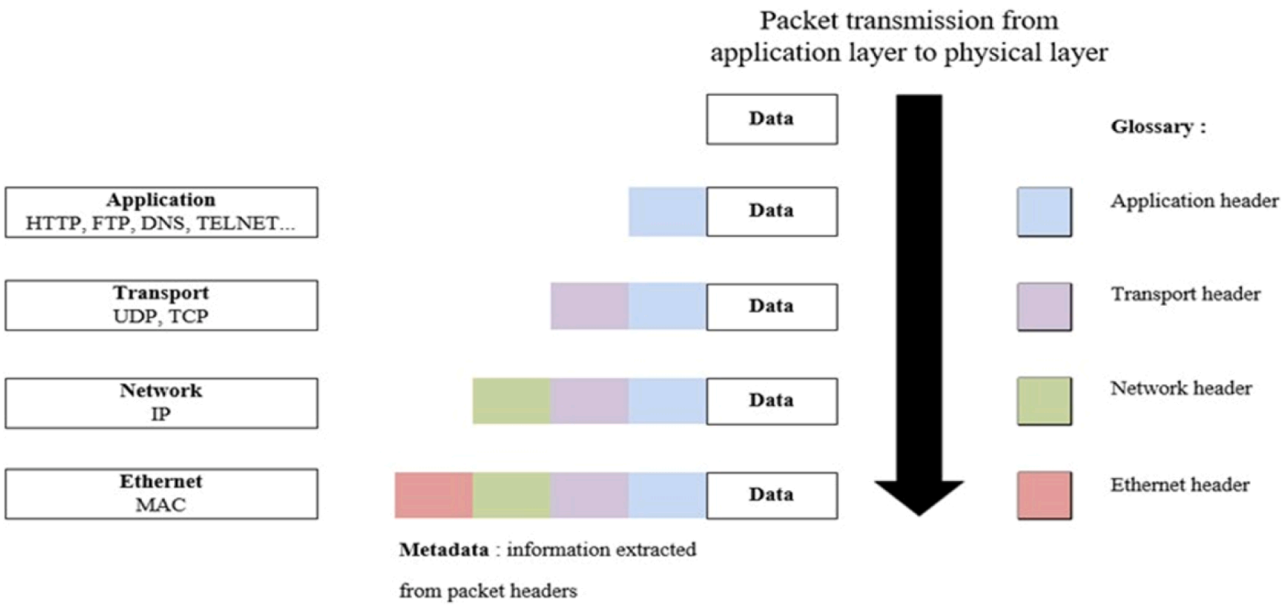


Fig. 1. The extraction of data and metadata from a packet.

In our study, we redefine the detection object as heterogeneous data flows, acknowledging the unstructured nature of the data. Our proposal centers on the development of a new universal representation encompassing all possible features established in accordance with network standards. This approach aims to address the complexities inherent in heterogeneous data environments and provide a comprehensive foundation for effective intrusion detection.

Our approach starts with understanding how data travels in a network, based on the Open System Interconnection (OSI) model. The OSI model describes data transmission in seven layers, from the physical layer to the application layer. Data is sent in packets, which are small units of data. Each packet has a payload (the actual data) and headers (metadata like source and destination IP addresses, and protocol information).

Intrusion Detection Systems (IDS) use these headers to extract features from network traffic. For example, tools like CICFlowMeter can extract 76 features from datasets such as CICIDS 2017. However, focusing only on individual packets or single data flows has two main problems:

1. Real-world network traffic is heterogeneous, meaning it contains data from various sources with different structures.
2. Metadata extracted from packet headers can have ambiguous names, even if they follow standard network protocols.

To address these issues, we propose a new method that handles heterogeneous data flows by creating a universal representation of network traffic.

3.2. Data flows reconstruction: big universal security database

In the realm of Machine Learning (ML), algorithms typically operate within the confines of specific data structures, rendering the utilization of a distinct Intrusion Detection System (IDS) for each network system impractical. This is primarily due to the interconnected nature of network layers, where an attack originating in the application layer can impact the physical layer, and vice versa. Therefore, an effective IDS necessitates the correlation and aggregation of data packets from various sources to provide comprehensive threat detection capabilities. In our proposed approach, we advocate for the development of a universal database capable of accommodating data from diverse sources.

The data, denoted as D, is sourced from various channels, as outlined in Section 3.1. Each data source is characterized by a specific structure, represented by a vector X. Data sharing the same vector structure is assigned to a common category, denoted as K. To establish a universal framework, we define a universal vector, U, encompassing all observable categories within network traffic, irrespective of its origin. The universal vector, U, comprises individual vectors, U= (X1, X2, …, Xk), with each vector defined by a set of features, F:

$$\begin{aligned} U_j &= X_1 \leftarrow (F_1, F_2, \ldots, F_n) \\ & X_2 \leftarrow (F_1, F_2, \ldots, F_m) \\ & \quad \cdot \\ & X_k \leftarrow (F_1, F_2, \ldots, F_z) \end{aligned}$$

Where $j = n + m + z$, representing the total dimensionality of the new universal vector, U.

At this juncture, our aim is to establish a standardized set of features within the universal vector, Uj. However, despite originating from the same network standard, certain features may exhibit redundancies or inconsistencies in naming conventions. In addressing this challenge, we leverage the metadata associated with each dataset, which provides a comprehensive description of all features. Our proposal entails the creation of a new corpus derived from metadata, wherein each feature is assigned a designation and detailed description. This corpus serves as a dynamic repository, automatically updated to accommodate new features as they are integrated. By harnessing metadata in this manner, we aim to streamline feature management and enhance the overall effectiveness of our universal IDS framework.

Machine Learning (ML) algorithms typically require specific data structures, making it impractical to use a separate IDS for each network system. This is because an attack in one layer (e.g., the application layer) can affect other layers (e.g., the physical layer). Therefore, our IDS needs to correlate data from various sources to detect threats effectively.

Our solution is to build a universal database that combines data from different sources. Here's how we do it:

1. **Data Collection**: We collect data (D) from various channels, each with a specific structure (vector X). Data with the same structure is grouped into a category (K).
2. **Universal Vector Creation**: We create a universal vector (U) that includes all categories of network traffic. The universal vector is a

combination of individual vectors ($U = X_1, X_2, \ldots, X_k$), each defined by a set of features (F).

3. **Standardization**: Although these features come from the same network standard, they might have different names or redundant information. We use metadata to create a new corpus that provides a clear description for each feature. This corpus is updated automatically as new features are added.

3.3. Construction of universal features vector

The universal features vector is a crucial component of our proposed methodology, designed to capture the essential characteristics of the input data for intrusion detection. Below, we provide a detailed description of the construction process, along with pseudocode to facilitate reproducibility.

1. Data Preprocessing:
   - Tokenize the input data.
   - Remove stopwords and perform stemming or lemmatization.
   - Calculate Term Frequency-Inverse Document Frequency (TF-IDF) for the tokenized data.
2. Dimensionality Reduction:
   - Apply Principal Component Analysis (PCA) to reduce the dimensionality of the TF-IDF matrix while preserving essential information.
3. Feature Aggregation:
   - Construct the universal features vector by aggregating the reduced-dimensional features (Fig. 2).

By providing this detailed description and pseudocode, we aim to clarify the construction process of the universal features vector and facilitate the reproducibility of our methodology.

3.4. Dataset description

3.4.1. Data collection process

Our study utilized three benchmark datasets commonly used for evaluating network intrusion detection systems: CICIDS 2017, NSL-KDD, and UNSW-NB15. Below is a detailed description of the data collection process for each dataset:

1. CICIDS 2017:
   - Source: The dataset was generated by the Canadian Institute for Cybersecurity.
   - Environment: The data was collected in a controlled environment that simulated real-world network traffic.
   - Traffic Types: It includes benign traffic and a variety of attack types such as DoS, DDoS, PortScan, and Brute Force.
   - Duration: Data was collected over a period of five days.
2. NSL-KDD:
   - Source: This dataset is an improved version of the original KDD Cup 1999 dataset, addressing issues like redundant records.
   - Traffic Types: It contains both normal traffic and different attack types including Probe, R2 L, U2R, and DoS.
   - Preprocessing: It has been preprocessed to remove duplicate records and improve data quality.
3. UNSW-NB15:

```
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler

def preprocess_data(data):
    # Tokenization, stopword removal, stemming/lemmatization
    processed_data = [ ]
    for doc in data:
        tokens = tokenize(doc)
        tokens = remove_stopwords(tokens)
        tokens = stem_or_lemmatize(tokens)
        processed_data.append(' '.join(tokens))
    return processed_data

def construct_tfidf_matrix(data):
    vectorizer = TfidfVectorizer()
    tfidf_matrix = vectorizer.fit_transform(data)
    return tfidf_matrix

def apply_pca(tfidf_matrix, n_components=100):
    scaler = StandardScaler()
    tfidf_matrix_scaled = scaler.fit_transform(tfidf_matrix.toarray())
    pca = PCA(n_components=n_components)
    pca_features = pca.fit_transform(tfidf_matrix_scaled)
    return pca_features

def construct_universal_features_vector(data):
    # Step 1: Preprocess data
    processed_data = preprocess_data(data)
    # Step 2: Construct TF-IDF matrix
    tfidf_matrix = construct_tfidf_matrix(processed_data)
    # Step 3: Apply PCA for dimensionality reduction
    pca_features = apply_pca(tfidf_matrix)
    # Step 4: Aggregate features to construct universal features vector
    universal_features_vector = np.mean(pca_features, axis=0)
    return universal_features_vector
```

Fig. 2. Universal features vector construction pseudocode.

- Source: The dataset was created by the Australian Centre for Cyber Security.
- Environment: Data was collected from a hybrid of real modern normal activities and synthetic contemporary attack behaviors.
- Traffic Types: It includes normal traffic as well as nine types of attacks such as Fuzzers, Analysis, Backdoors, DoS, and Exploits (Figs. 3 and 4).

3.4.2. Distribution of classes

The datasets exhibit varying distributions of classes, which is important to consider for training and evaluating our models:

1. CICIDS 2017:
   - Normal Traffic: Approximately 80% of the dataset.
   - Attack Traffic: Approximately 20%, with a detailed breakdown as follows: DoS/DDoS: 10%, PortScan: 5%, Brute Force: 3%, Others: 2%
2. NSL-KDD:
   - Normal Traffic: Around 53% of the dataset.
   - Attack Traffic: Around 47%, with categories: Probe: 21%, DoS: 13%, R2L: 12%, U2R: 1%
3. UNSW-NB15:
   - Normal Traffic: Approximately 88% of the dataset.
   - Attack Traffic: Approximately 12%, broken down as: Fuzzers: 5%, Analysis: 2%, DoS: 2%, Exploits: 1%, Others: 2%

3.4.3. Data preprocessing and handling class imbalance

Preprocessing steps were essential to prepare the data for training and to address class imbalance issues. Here are the steps taken:

1. Data Cleaning:
   - Duplicate Removal: Duplicate records were removed to ensure the quality and uniqueness of the data.
   - Missing Values: Missing values were handled by either imputing with the mean/mode or by removing the records if the missing rate was high.
2. Feature Engineering:
   - Normalization: Continuous features were normalized to a range between 0 and 1 using min-max scaling.
   - Categorical Encoding: Categorical features were encoded using one-hot encoding to convert them into a numerical format suitable for ML models.
3. Handling Class Imbalance:
   - Oversampling: Synthetic Minority Over-sampling Technique (SMOTE) was applied to generate synthetic samples for the minority classes, balancing the class distribution.
   - Undersampling: Random undersampling of the majority class was performed to ensure a balanced training set, reducing the risk of model bias towards the majority class.
   - Class Weights: During model training, class weights were adjusted to penalize misclassifications of the minority class more heavily, encouraging the model to pay more attention to underrepresented classes.

The careful collection and preprocessing of the datasets ensure that our models are trained on high-quality, balanced data, enhancing their ability to detect a wide range of network intrusions effectively. These steps contribute significantly to the robustness and generalizability of our intrusion detection framework.

Our proposed approach integrates various machine learning techniques to enhance the detection of unknown intrusions in network traffic. This section explains the methodology in detail, including sample feature data from the datasets used in our experiments.

Before applying machine learning algorithms, the raw network traffic data from the NSL-KDD, UNSW, and CICIDS datasets were preprocessed. This involved the following steps:

- **Feature Extraction:** Relevant features were extracted from the network traffic data using TF-IDF (Term Frequency-Inverse Document Frequency) and Cosine similarity. These techniques helped in transforming the textual data into numerical features that machine learning algorithms can process.
- **Dimensionality Reduction:** Principal Component Analysis (PCA) was applied to reduce the feature space while retaining the most important information. This step helps in mitigating the curse of dimensionality and improving the efficiency of the model (Table 2).

3.5. Feature description

This study leverages three widely-used datasets: CICIDS 2017, NSL-KDD, and UNSW-NB15. Each dataset provides a range of features essential for effective intrusion detection. Below is an overview of key features included in our analysis:

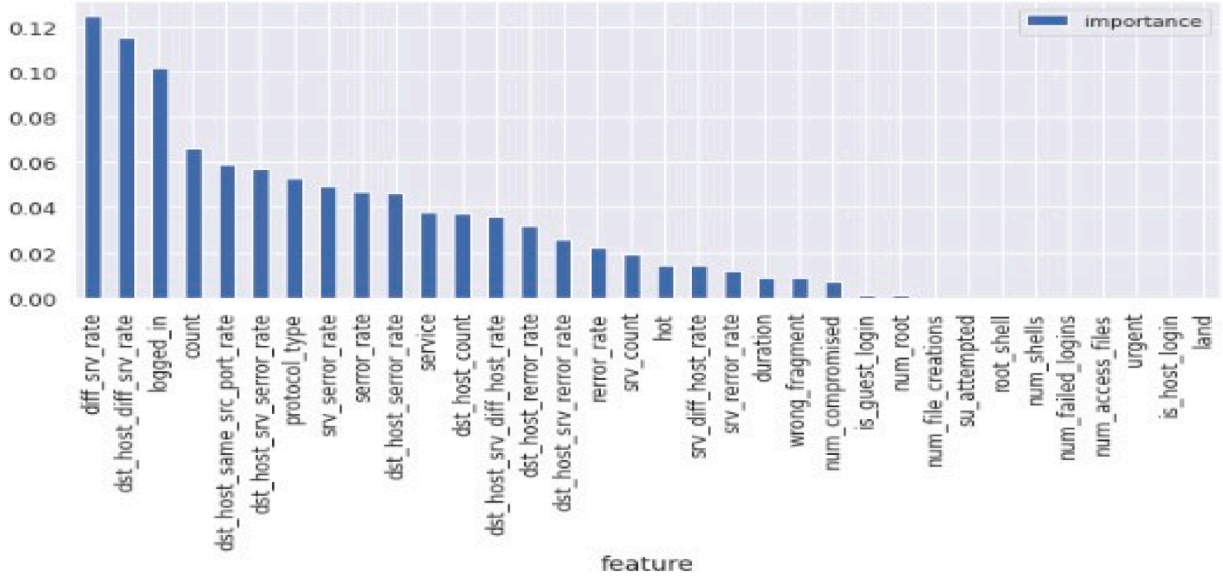


Fig. 3. Feature classification for NSL-KDD dataset.

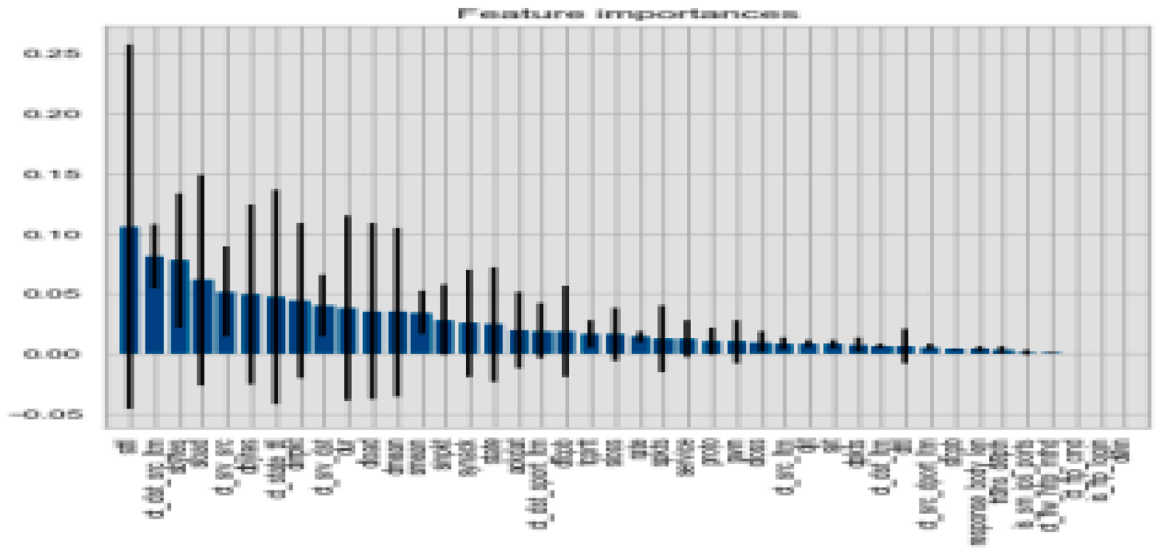


Fig. 4. Feature classification for UNSW-NB15 dataset.

Table 2
Sample Feature Data (from NSL-KDD dataset).

Feature	Description	Sample Value
Duration	Length of the connection in seconds	0.03
Protocol Type	Type of protocol (e.g., TCP, UDP)	TCP
Service	Network service on the destination (e.g., HTTP, FTP)	HTTP
Source Bytes	Number of data bytes from source to destination	181
Destination Bytes	Number of data bytes from destination to source	5450

- **CICIDS 2017**: This dataset includes 76 features that capture various traffic behaviors, such as Flow Duration, Total Fwd Packets, Fwd Packet Length Mean, Flow Bytes/s, and others. These features provide insights into bidirectional flows, allowing for the identification of attack patterns in real-world scenarios.
- **NSL-KDD**: Comprising 41 features, NSL-KDD includes attributes such as Protocol Type, Service, Flag, Source Bytes, and Destination Bytes. These features are used to differentiate between normal and abnormal traffic types.
- **UNSW-NB15**: This dataset offers 46 features, covering aspects like srcip, sport, dstip, dsport, and state. Additionally, it includes high-level summaries like Total Bytes and Total Packets, which are valuable for anomaly detection.

Each dataset's features are preprocessed and standardized to form a universal representation, as described in the following section. By incorporating diverse features across datasets, our approach enhances detection accuracy and robustness in heterogeneous data environments.

3.6. Innovative aspects and theoretical advancements in our work

1. **Novel Universal Representation of Network Features:** Our paper introduces a new methodology for restructuring network data based on predefined features, employing Term Frequency-Inverse Document Frequency (TF-IDF) and Cosine similarity to create a universal security database. This approach is innovative as it allows for the aggregation and analysis of heterogeneous data flows within a singular, comprehensive framework. This universal representation addresses the complexities inherent in diverse data environments, which is a significant theoretical advancement in the field of intrusion detection.
2. **Big Universal Security Database (BUSD):** We propose the Big Universal Security Database (BUSD), which is capable of accommodating and correlating data from various sources. The theoretical foundation of BUSD lies in its ability to integrate data flows of different structures into a unified vector representation. This method enhances the capability of Intrusion Detection Systems (IDS) to detect unknown attacks by considering the interconnected nature of network layers. This is a novel contribution to the theoretical understanding of data fusion in network security.
3. **Dimensionality Reduction with PCA:** Our approach employs Principal Component Analysis (PCA) to reduce the dimensionality of the extensive universal security database while preserving essential information. This theoretical framework for dimensionality reduction is crucial for managing large-scale data and improving the efficiency of machine learning algorithms used in IDS.
4. **Ensemble Learning Methodology:** We introduce a new ensemble learning method that combines both parallel and sequential approaches for detecting unknown attacks. This methodology enhances the accuracy and robustness of IDS by leveraging the strengths of multiple machine learning algorithms. The theoretical contribution here lies in the innovative use of ensemble learning to handle heterogeneous data more effectively.
5. **Evaluation Across Multiple Scenarios:** The efficacy of our proposed approach is evaluated using various machine learning algorithms, including Naïve Bayes, K-Nearest Neighbor, Logistic Regression, Decision Tree, and Random Forest. Our experimental findings, conducted on datasets such as CICIDS 2017, NSL-KDD, and UNSW, demonstrate the universality, versatility, and effectiveness of our approach. This comprehensive evaluation contributes to the theoretical understanding of how different machine learning techniques can be applied to heterogeneous network data.

These points collectively establish a strong theoretical foundation and demonstrate the innovative contributions of our research to the fields of data fusion and intrusion detection in heterogeneous network environments.

4. Proposed approach

4.1. Approach description

The proposed method for intrusion detection follows a systematic, multi-stage process designed to harness the strengths of ensemble

learning across heterogeneous datasets. The key stages of this approach are outlined as follows:

1. Data Collection and Preparation:
   - Gather data from multiple, heterogeneous network datasets, including NSL-KDD, UNSW, and CICIDS.
   - Preprocess the datasets by cleaning, encoding categorical features, and normalizing values to ensure consistency across diverse data sources.
2. Feature Extraction and Transformation:
   - Apply Term Frequency-Inverse Document Frequency (TF-IDF) and Cosine similarity to extract meaningful features, transforming raw data into a structured format.
   - Generate a universal feature representation to unify features across datasets, ensuring compatibility in a multi-source environment.
3. Dimensionality Reduction:
   - Use Principal Component Analysis (PCA) to reduce the dimensionality of the extracted feature space while retaining essential information, optimizing computational efficiency.
4. Ensemble Learning Model Construction:
   - Design a unique ensemble learning framework that combines both parallel and sequential ensemble approaches.
   - Train multiple machine learning algorithms (e.g., Naïve Bayes, K-Nearest Neighbor, Logistic Regression, Decision Tree, and Random Forest) on the processed data, allowing the model to leverage the strengths of different classifiers.
5. Model Evaluation:
   - Evaluate the trained ensemble model across multiple datasets, assessing its performance on heterogeneous data sources.
   - Measure accuracy, recall, precision, F1-score, and other metrics to confirm the model's ability to detect unknown intrusions effectively.
6. Performance Optimization and Analysis:
   - Analyze the model's strengths and limitations, focusing on detection accuracy and computational efficiency.
   - Identify potential improvements and areas for future research, such as reducing computational complexity and enhancing adaptability to evolving threats.

To construct a comprehensive corpus, we relied on metadata, which furnishes detailed descriptions of the extracted features.

These descriptions offer invaluable insights surpassing mere feature names, facilitating a nuanced understanding and robust analysis of network traffic data. In our investigation, the focal point is heterogeneous data flows, each delineated by a distinct set of features. Notably, some features may manifest across multiple data flows, while commonalities can be identified among various network types. To address these intricacies, we leverage the corpus to scrutinize feature descriptions utilizing similarity measures. Additionally, it serves as a foundation for generating a novel universal features representation.

Illustrating the structural underpinning of our proposed methodology, Fig. 5 depicts the collection of Packet Captures (PCAP) from diverse sources alongside their corresponding metadata. This metadata is instrumental in constructing a comprehensive and expansive universal security database, as delineated in the accompanying figure.

Metadata M (F, D). Within the framework of Metadata M (F, D), we establish a corpus tailored for the storage and aggregation of all features present in network traffic. This corpus encompasses a comprehensive collection of feature names (F) alongside their corresponding detailed descriptions (D). Additionally, each feature is associated with a designated category (K), which serves to describe the origin of the network traffic data to which it pertains.

New universal features representation. To establish a new universal features representation, we employed Cosine similarity based on Term Frequency-Inverse Document Frequency (tf-idf). This approach proves efficacious in our context due to the extensive and overlapping nature of the provided descriptions. Tf-idf serves to quantify the importance of terms within documents, calculated as follows:

$$\mathrm{tf}-\mathrm{idf}(\mathrm{t},\mathrm{d}) = \mathrm{tf}(\mathrm{t},\mathrm{d}) * \mathrm{idf}(\mathrm{t}) \tag{1}$$

where tf (t, d) represents the frequency of a term occurs in a document and idf (t) is equal to log (number of documents in corpus / number of documents containing the term).

The result of tf-idf is used to calculate the cosine similarity:

$$\mathrm{CosineSimilarity}\ (\mathrm{tf}-\mathrm{idf}(\mathrm{d1}), \mathrm{tf}-\mathrm{idf}(\mathrm{d2})) = (\mathrm{d1} * \mathrm{d2}) / \parallel \mathrm{d1} \parallel * \parallel \mathrm{d2} \parallel \tag{2}$$

Drawing from this methodology, we enact the proposed approach outlined in the preceding algorithm to generate novel universal features. This method facilitates the comparison and alignment of feature descriptions across various categories through tf-idf. Each tf-idf computation produces a vector representation, collectively forming a set of vectors utilized to construct a space model employing cosine similarity. Within each category, features with a cosine score exceeding zero are

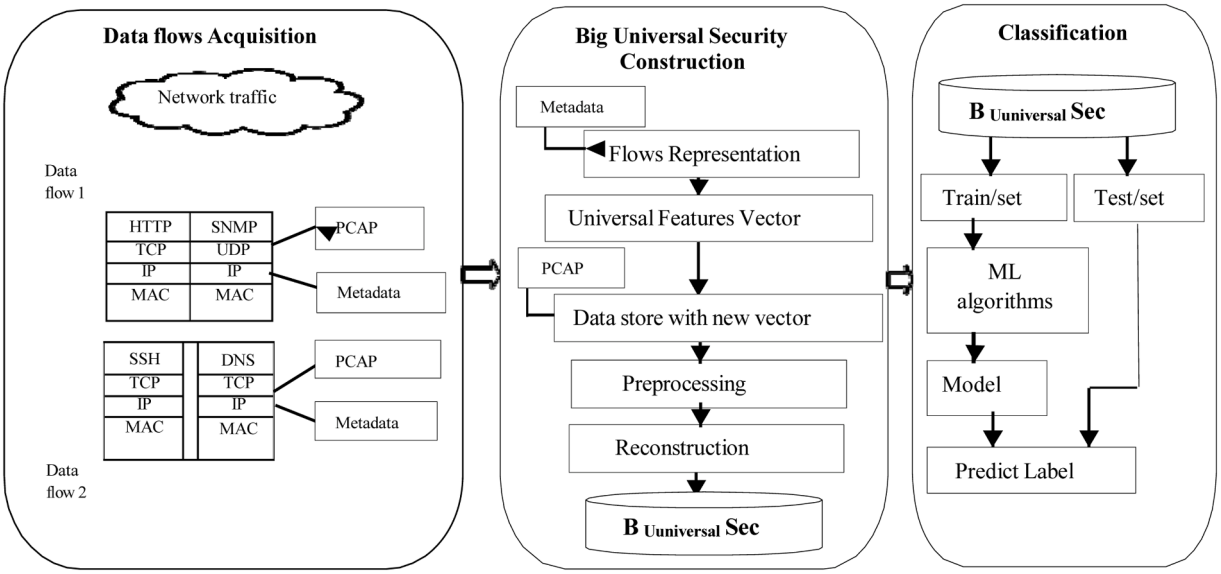


Fig. 5. The proposed approach.

collated into a list and arranged in descending order. Subsequently, the primary feature in this sorted list is designated as a common feature, thereby assigning it a new universal name (Algorithm 1).

Big universal security database U (M, Data). The data originates from heterogeneous data flows, characterized as unstructured data. Initially, the first P flow bytes serve as the foundation for constructing the universal vector U, with its dimensionality matching that of P. Subsequently, the second set of flow bytes (P') sourced from alternate data origins are juxtaposed with the initial P flow bytes. To facilitate data storage, we overlay the data flows. In instances where P and P' possess differing dimensions or where one set of flow bytes is deficient compared to the other, vacant bytes are populated with zeros to maintain consistency. Following this process, a new database encompassing diverse network intrusion detection structures is established.

Data reconstruction. In order to effectively process the new database, we introduce a novel data reconstruction technique. Given that this database contains zero values, which could potentially impact the efficiency of Machine Learning (ML) algorithms by introducing biases towards large values, we employ Principal Component Analysis (PCA). PCA operates by transforming the data through projection onto a set of orthogonal axes.

By doing so, PCA facilitates optimal reconstruction of the data, effectively handling values with null entropies by disregarding them. Moreover, PCA ensures superior dimensionality reduction, transitioning from the original M dimensions to a more compact N dimensionality. Leveraging PCA with the big universal security database enables us to prepare the data for final classification with enhanced efficiency and accuracy.

Data classification. Following the establishment of the new big universal security database, our attention turned towards data classification. The objective of this phase is to introduce learning models by extracting insights from databases featuring K categories. At this stage, leveraging knowledge extraction must be conducted on a per-base basis, necessitating a parallel architecture. In this regard, we employ a bagging model in conjunction with a random forest, operating concurrently across multiple data structures. Subsequently, our focus shifts to decision-making based on the models generated: determining whether the vector Y test corresponds to a normal or abnormal label.

This decision-making process relies on the exploitation of knowledge derived from diverse learning bases, often executed sequentially. To enhance classification performance, we incorporate the Adaboost algorithm as a boosting model, ensuring robust and accurate classification outcomes (see Fig. 6).

4.2. Rationale for choosing machine learning algorithms

In our proposed framework, we have selected Random Forest and Adaboost as the primary machine learning algorithms. The rationale for choosing these specific algorithms is based on their unique strengths and their ability to address the challenges of heterogeneous intrusion detection.

4.2.1. Random forest

Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes for classification tasks. The reasons for selecting Random Forest include:

1. Robustness to Overfitting:
   - Random Forest mitigates overfitting by averaging the results of multiple decision trees, reducing the variance of the model.
2. Handling High-Dimensional Data:
   - Random Forest is well-suited for high-dimensional data, as it can handle a large number of features without significant performance degradation.
3. Feature Importance:
   - Random Forest provides an inherent mechanism to estimate feature importance, which helps in understanding the contribution of each feature to the model's predictions.
4. Scalability and Efficiency:

Algorithm 1
Generating a big universal security database.

Input: Categories *K*= (C1, C2, …, Ck), Metadata Mk = (F, D) where feature_names *F*= (f1, f2, …, fn), feature_descriptions *D*= (d1, d2, …, dn), datasets Data={(x1,y1) … (xn,yn)}.
Output: Universal vector *U*= {(x1, y1), (x2, y2), …(xj,yj} where j the new dimension of U**Universal feature creation**Corpus ← F (*M* 1 ∪ *M* 2 ∪*Mk*): {((f1, d1), …(fn, dn)) ∪((f1, d1), …(fj, dj)) ∪((f1, d1), …(fc, dc))}Categories CK ← F (Mk)for F in {corpus, categories} do

for terms t in Dk do
Calculate tf (t,d): equation1
Calculate idf (t): equation 2
Calculate Cosine (tf_idf (Dk)): equation 3
if Cosine (tf_idf (Dk)) > 0 then

Enumerated_list ← {[similar_features in C1] … [similar_features in Ck]}
Stored_list ← sorted (enumerated_list)
end
end
Common_features ← MaxCosineScore in (sorted_list (Ck))

end
Generate universal features vector

Universal_features ← {F in (C1∪ Ccom ∪… Ck)}
Store Datafor (xi,yi) in (data) do

if yi ≠ 0 then
store(xi,yi) in U

end
else
yi ← 0
end
end
Big Universal Security databaseU ← {Mk(F,D), Data ((x1,y1), (x2,y2), …(xj,yj))}

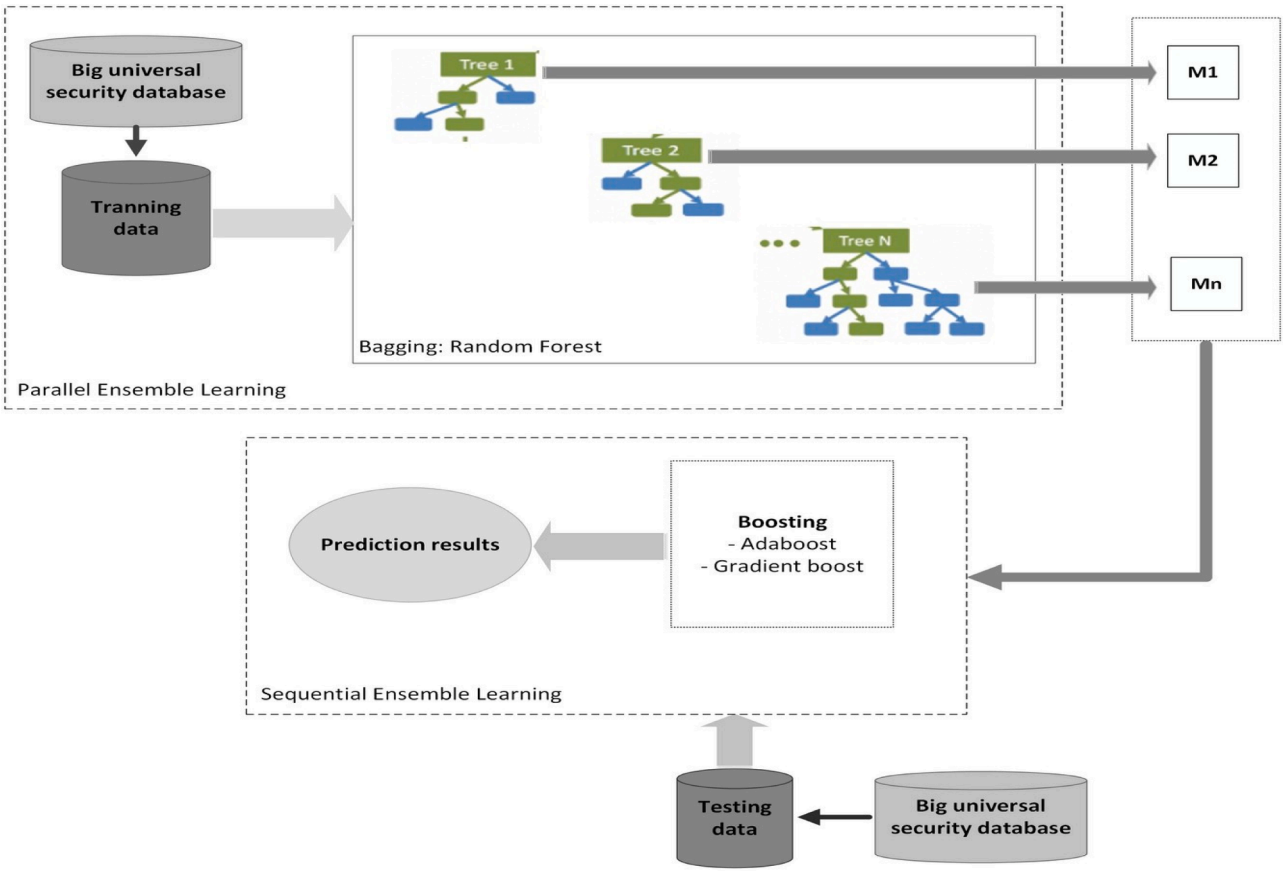


Fig. 6. The proposed architecture.

- Random Forest is computationally efficient and scalable, making it practical for large datasets commonly encountered in intrusion detection.

4.2.2. Adaboost

Adaboost, short for Adaptive Boosting, is another ensemble learning method that combines weak classifiers to create a strong classifier. The reasons for selecting Adaboost include:

1. Improved Classification Accuracy:
   - Adaboost focuses on misclassified instances by adjusting the weights of the training data, which improves the overall classification accuracy.
2. Adaptability:
   - Adaboost is adaptive, meaning it adjusts to the complexity of the data and can handle both linear and non-linear relationships.
3. Combining Weak Learners:
   - Adaboost effectively combines weak learners to form a robust model, enhancing the overall predictive performance.
4. Versatility:
   - Adaboost can be used with various base classifiers, providing flexibility in model selection and adaptation to different types of data.

4.2.3. Complementing the proposed framework

The combination of Random Forest and Adaboost complements our proposed framework by leveraging the strengths of both algorithms:

1. Diverse Decision-Making:
   - Random Forest provides diverse decision-making through multiple decision trees, while Adaboost enhances accuracy by focusing on difficult-to-classify instances.
2. Balanced Bias-Variance Trade-off:
   - The ensemble nature of both algorithms helps in balancing the bias-variance trade-off, leading to more robust and reliable predictions.
3. Handling Heterogeneous Data:
   - The algorithms' ability to handle high-dimensional and heterogeneous data ensures that our framework can effectively detect intrusions across diverse datasets.
4. Improved Generalization:
   - The combination of these algorithms improves generalization, reducing the risk of overfitting and enhancing the model's performance on unseen data.

By providing this detailed rationale, we aim to clarify our choice of machine learning algorithms and how they contribute to the effectiveness of our proposed framework.

4.3. Complexity analysis

4.3.1. Construction of universal features vector

The construction of the universal features vector involves several key steps, including the calculation of Term Frequency-Inverse Document Frequency (TF-IDF) and the application of semantic Cosine similarity.

1. TF-IDF Calculation:
   - For a dataset with N documents and T terms, the complexity of calculating the term frequency for all terms in all documents is O ($N \times T$).

- The complexity of calculating the inverse document frequency is O(T) as it involves counting the number of documents that contain each term.
- Therefore, the overall complexity for the TF-IDF calculation is O ($N \times T$).

2. Cosine Similarity Calculation:
   - Calculating the cosine similarity between two vectors of length T involves O(T)operations.
   - For N documents, calculating the similarity matrix would require O($N^2 \times T$) operations.

4.3.2. Principal component analysis (PCA)

The PCA step is used to reduce the dimensionality of the universal features vector while preserving essential information.

- The complexity of PCA, which involves computing the covariance matrix (which is O($T^2 \times N$)) and then performing eigenvalue decomposition on a $T \times T$ matrix, which is O(T^3).
- Therefore, the overall complexity for PCA is O($T^2 \times N + T^3$).

4.3.3. Ensemble learning model

The ensemble learning model combines multiple machine learning algorithms, including Naïve Bayes, K-Nearest Neighbor, Logistic Regression, Decision Tree, and Random Forest.

1. Naïve Bayes:
   - Training complexity is O($N \times T$).
   - Prediction complexity is O(T).
2. K-Nearest Neighbor (KNN):
   - Training complexity is O(1) since it stores the training samples.
   - Prediction complexity is O($N \times T$) as it involves computing the distance to all training samples.
3. Logistic Regression:
   - Training complexity is O($N \times T \times I$), where I is the number of iterations.
   - Prediction complexity is O(T).
4. Decision Tree:
   - Training complexity is O($N \times$ TlogN).
   - Prediction complexity is O(logN).
5. Random Forest:
   - Training complexity is O($M \times N \times$ TlogN), where M is the number of trees.
   - Prediction complexity is O($M \times$ logN).

4.3.4. Overall complexity

Combining these steps, the overall complexity of constructing the universal features vector and training the ensemble learning model can be summarized as:

- Construction of Universal Features Vector: O($N \times T + N^2 \times T + T^2 \times N + T^3$)
- Ensemble Learning Model Training and Prediction: Summing up the complexities for each algorithm as detailed above.

By providing this detailed complexity analysis, we aim to clarify the scalability of our proposed approach.

5. Results and evaluation

The implementation and evaluation of the proposed approach were conducted utilizing Google Colab with GPU execution capabilities. We utilized two widely recognized public standard intrusion detection datasets to provide an approximate illustration of the universal features vector. This choice allowed us to assess the effectiveness and versatility of our approach across different datasets and scenarios. By leveraging the computational power of Google Colab's GPU execution environment, we aimed to expedite the processing and analysis of the datasets, facilitating more efficient experimentation and evaluation of our proposed methodology.

5.1. Datasets

In our approach, the detection object encompasses multiple data flows originating from heterogeneous sources. To illustrate the versatility of our methodology, we utilized three distinct datasets, each comprising varied sources: NSL_KDD (NSL-KDD), CICIDS 2017 (Xu, Shen & Du, 2020), and UNSW. Notably, these datasets exhibit varying structures, with NSL KDD containing 41 features and serving as an updated version of the KDD99 dataset. In contrast, CICIDS 2017 comprises 76 features, while UNSW consists of 46 features.

Each of these public datasets is accompanied by metadata, providing detailed descriptions of the extracted features for each Packet Capture (PCAP). To consolidate and standardize this information, we aggregated all metadata into a unified corpus. This corpus serves to establish a cohesive and meaningful representation of the features, facilitating comprehensive analysis and comparison across datasets. By incorporating metadata from each dataset into the corpus, we ensure that our approach accounts for the diverse characteristics and nuances present in the network traffic data.

5.2. Metrics

To assess the efficacy of Machine Learning (ML) algorithms employed for heterogeneous traffic, we utilize the metrics outlined in the table below, as a result of binary classification. Leveraging the information provided in Table 3, we derive the equations (eq.3) and (eq.4) to serve as evaluation metrics for the database.

The performance metrics used for evaluation are as follows:

Based on these metrics, we calculate the following evaluation metrics:

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} \tag{3}$$

$$\text{Detectionrate}: \quad Recall = \frac{TP}{TP + FN} \tag{4}$$

1. **Accuracy:** The proportion of correctly predicted instances out of the total instances.
2. **Precision:** The ratio of true positive predictions to the total predicted positives, indicating the accuracy of positive predictions.
3. **Recall (Sensitivity):** The ratio of true positive predictions to the total actual positives, reflecting the model's ability to identify all positive instances.
4. **F1-Score:** The harmonic mean of precision and recall, providing a balanced measure of the model's performance.
5. **Specificity (True Negative Rate):** The ratio of true negative predictions to the total actual negatives, showing the model's ability to identify negative instances.
6. **ROC-AUC (Receiver Operating Characteristic - Area Under Curve):** The AUC score summarizes the trade-off between true positive and false positive rates, providing an aggregate measure of performance across all classification thresholds.

Table 3
The metrics used for the evaluation.

True positive	Normal instances are classified as normal				
True negative	Malicious malicious	instances	are	classified	as
False positive	Normal malicious	instances	are	classified	as
False negative	Malicious normal	instances	are	classified	as

7. **Confusion Matrix:** A detailed breakdown of true positives, true negatives, false positives, and false negatives, offering insight into the types of errors made by the model.
8. **MCC (Matthews Correlation Coefficient):** A comprehensive metric that considers true and false positives and negatives, providing a balanced evaluation even in the case of imbalanced datasets.
9. **Balanced Accuracy:** The average of recall obtained on each class, accounting for imbalanced class distributions.

5.3. Experimental setting

The experiments were conducted across three distinct scenarios:

5.3.1. Type 1 experiments

In this scenario, various ML algorithms including Naïve Bayes (NB), K-Nearest Neighbor (KNN), Logistic Regression (LR), Decision Tree (DT), and Random Forest (RF) were applied to perform binary classification. The algorithms were executed with their default parameters to detect intrusions from a traffic dataset containing heterogeneous network data.

5.3.2. Type 2 experiments

For this scenario, the experiments were carried out on the big universal security dataset. The dataset was trained using three types of data: NSL KDD, UNSW, and CICIDS. Subsequently, the obtained model was tested with all three types of datasets to assess its performance and generalizability across diverse sources.

5.3.3. Type 3 experiments

In this scenario, the experiments focused on training the big universal security dataset using two types of data: NSL KDD and CICIDS. The resulting model was then tested with the UNSW dataset to evaluate its performance when confronted with data from a different source.

These experiments were designed to provide a comprehensive analysis of the proposed approach's effectiveness and versatility in detecting intrusions across heterogeneous network data, considering various training and testing combinations..

5.4. Detection results

Regarding the construction of the universal features representation, we conducted an analysis of the metadata associated with data flows to create a new corpus comprising four key attributes: feature names, types, detailed descriptions, and categories. These attributes were concatenated to facilitate the calculation of similarity for each feature within each category.

In the case of the CICIDS 2017 dataset, the authors emphasized the generation of bidirectional PCAP flows, described as "forward" and "backward" (Xu, Shen & Du, 2020). To align with the descriptions of other datasets, we replaced these terms with their synonyms: "forward" denoting traffic from source to destination, and "backward" representing traffic from destination to source. Additionally, we classified features based on their types, such as numeric, average, percentage, categorical, binary, etc.

For the implementation process, we began by cleaning the data, which involved removing stop words. Subsequently, we utilized the TfidfVectorizer from the Scikit-Learn library's feature_extraction module to apply term frequency-inverse document frequency (tf-idf). By employing the Linear Kernel Module and cosine similarity, we derived scores for each feature pair and sorted them in descending order.

The application of the proposed method revealed that several features across different categories exhibited approximate similarity. However, certain cases presented complexities. For instance, the feature "duration" from the NSL_KDD dataset was found to be similar to "flow duration" from CICIDS, while "flow bytes" in CICIDS shared similarity with "duration" from NSL_KDD.

The intersection between the two comparisons in both ways is the feature "duration". To select the closest feature to "duration", we evaluated their cosine scores and picked the one with highest score, eventually "flow duration".

The new approximate features vector U has 110 features instead of 117. Finally, new universal names are assigned to features.

Regarding the storage of data flows, it is organized according to the approximate vector U. Features representing empty values are stored as zeros within the approximate vector.

Following data storage, preprocessing is conducted as follows:

Data Cleaning: Redundant instances are removed to ensure the dataset's cleanliness and efficiency.

Categorical Feature Encoding: Features with categorical values are encoded numerically to facilitate compatibility with ML algorithms. For instance, categorical values such as "protocol" containing values like "http, dns, telnet, etc." are converted into numerical representations. For instance, "http" might be assigned the numerical value "1," "dns" the value "2," and so forth.

Data Standardization: Finally, data standardization is performed to ensure uniformity across feature values. This involves calculating the standard normal distribution of feature values, ensuring consistency and comparability across the dataset.

Furthermore, after the data preprocessing stage, Principal Component Analysis (PCA) is applied. The results depicted in Fig. 7 illustrate that utilizing 30 principal components enables the recognition of 100% of the data, effectively reducing the dimensionality from 110 features to 30 features.

Subsequently, the ML algorithms mentioned previously are employed to evaluate the new universal security database, which encompasses various structures.

The detection scenario is executed utilizing the "dataset" feature (refer to Fig. 6) to select Train_data and Test_data from the new database. The data is split as follows:

For the big universal security train, 70% of NSL KDD and 70% of CICIDS 2017 are utilized.

For the big universal security test, the remaining 30% of NSL KDD and the remaining 30% of CICIDS 2017 are employed.

This approach ensures a comprehensive evaluation of the model's performance across different datasets while maintaining a balanced training and testing distribution, (Figs. 8 and 9).

For each ML algorithm, a model is trained using the big universal security training dataset. Subsequently, these trained models are utilized to predict labels on the big universal security test dataset. The experimental results summarized in Table 4 demonstrate the versatility and universality of our approach in detecting intrusions from the new universal representation.

These findings affirm that our proposed detection system is capable of learning and extracting knowledge from heterogeneous sources effectively. This underscores the significance of employing a unified approach that can accommodate diverse data structures and sources, thereby streamlining intrusion detection processes and enhancing overall security measures.

The evaluation of the proposed model is conducted through two distinct scenarios, employing parallel and sequential methods. The results of these evaluations demonstrate the exceptional performance of our model with heterogeneous data.

In particular, the outcomes of training on one type of structure and testing on other types of structures illustrate that the proposed model consistently achieves superior results. This is evidenced by the findings presented in Tables 4 and 5.

These results underscore the robustness and effectiveness of our model in accommodating diverse data structures and sources, further validating its utility and potential in real-world intrusion detection scenarios.

The Table 4 presents the performance metrics of various machine learning algorithms for binary classification in intrusion detection

	dataset	duration	source packet bytes	destination packet bytes	service	protocole_type	source packet mean size	destination packet mean size	source packets/s	destination packets/s	...	urgent	ftp_cmd_outbound	same service source	same service between dst and src	source time to live	destination time to live	source init win bytes	destination init win bytes	land	label
0	cicids	3.0	2	0	0	0	6.0	0.0	666666.666700	0.000000	...	0	0	0.0	0.0	3	0	33	-1	0	0
1	cicids	109.0	1	1	0	0	6.0	6.0	9174.311927	9174.311927	...	1	0	0.0	0.0	0	0	29	256	0	0
2	cicids	52.0	1	1	0	0	6.0	6.0	19230.769230	19230.769230	...	1	0	0.0	0.0	0	0	29	256	0	0
3	cicids	34.0	1	1	0	0	6.0	6.0	29411.764710	29411.764710	...	1	0	0.0	0.0	0	0	31	329	0	0
4	cicids	3.0	2	0	0	0	6.0	0.0	666666.666700	0.000000	...	0	0	0.0	0.0	3	0	32	-1	0	0

5 rows × 24 columns

```
new_dataset.tail()
```

	dataset	duration	source packet bytes	destination packet bytes	service	protocole_type	source packet mean size	destination packet mean size	source packets/s	destination packets/s	...	urgent	ftp_cmd_outbound	same service source	same service between dst and src	source time to live	destination time to live	source init win bytes	destination init win bytes	land	label
1108901	nslkdd	0.0	794	333	smtp	tcp	0.0	0.0	0.0	0.0	...	0	0	1.00	0.01	0	0	0	0	0	normal
1108902	nslkdd	0.0	317	938	http	tcp	0.0	0.0	0.0	0.0	...	0	0	1.00	0.01	0	0	0	0	0	normal
1108903	nslkdd	0.0	54540	8314	http	tcp	0.0	0.0	0.0	0.0	...	0	0	1.00	0.00	0	0	0	0	0	back
1108904	nslkdd	0.0	42	42	domain_u	udp	0.0	0.0	0.0	0.0	...	0	0	1.00	0.00	0	0	0	0	0	normal
1108905	nslkdd	0.0	0	0	sunrpc	tcp	0.0	0.0	0.0	0.0	...	0	0	0.25	0.00	0	0	0	0	0	mscan

Fig. 7. The big universal security database.

	cicids	nsl	unsw
0	flow_duration	duration	dur
1	total_fwd_packets	src_bytes	sbytes
2	total_backward_packets	dst_bytes	dbytes
3	missing	service	service
4	missing	protocol_type	proto
5	fwd_packet_length_mean	missing	smean
6	bwd_packet_length_mean	missing	dmean
7	fwd_packets/s	missing	sload
8	bwd_packets/s	missing	dload
9	fwd_act_data_pkts	count	dpkts
10	fin_flag_count	flag	missing
11	syn_flag_count	serror_rate	tcprtt
12	ack_flag_count	missing	ackdat
13	urg_flag_count	urgent	missing
14	missing	num_outbound_cmds	ct_ftp_cmd
15	missing	same_srv_rate	ct_srv_src
16	missing	dst_host_same_src_port_rate	ct_src_dport_ltm
17	fwd_iat_max	missing	sttl
18	bwd_iat_max	missing	dttl
19	init_win_bytes_forward	missing	swin
20	init_win_bytes_backward	missing	dwin
21	missing	land	is_sm_ips_ports
22	label	attack	label

Fig. 8. Features similarity results between NSL_KDD, CICIDS2017 and UNSW.

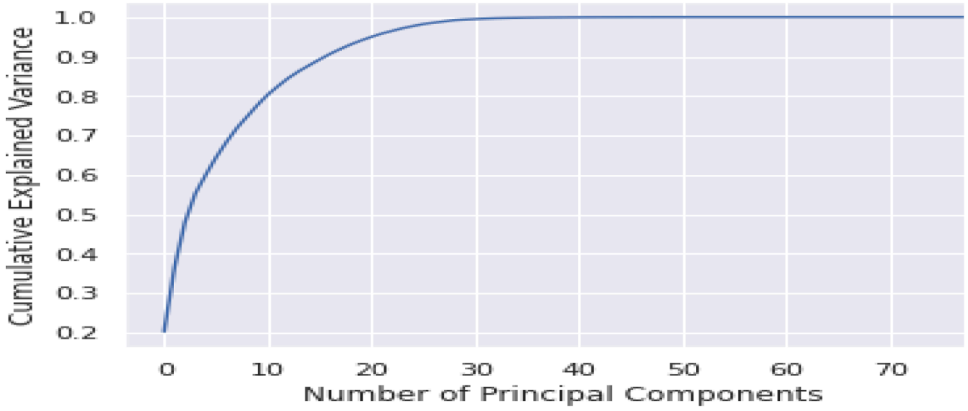


Fig. 9. PCA results.

Table 4
Evaluation results of the new universal security database.

	Binary classifica tion	Accura cy	Recall	False positive	F1-Score	Specificity	ROC-AUC	MCC
NB	Normal	86.34	88.00	13.4	86.15	86.60	0.90	0.73
	Attack		85.00		84.89	85.00	0.88	0.71
LR	Normal	97.22	97.00	3.06	97.11	96.94	0.99	0.94
	Attack		97.00		97.00	97.00	0.99	0.94
KN N	Normal	99.64	100	0.31	99.82	99.69	0.99	0.98
	Attack		100		100.00	100.00	1.00	1.00
DT	Normal	99.45	100	0.53	99.72	99.47	0.99	0.97
	Attack		99		99.00	99.00	0.99	0.96
RF	Normal	**99,80**	**100**	**0.19**	99.90	99.81	1.00	0.99
	Attack		**100**		00.00	100.00	1.00	1.00

Table 5
Evaluation results of Ensemble Learning model with NSL KDD, UNSW, CICIDS in train and test.

	Binary classifica tion	Accura cy	Recall	False positive	F1-Score	Specificity	ROC-AUC
Adaboost	Normal	99.30	99.10	0.15	99.20	99.85	0.99
	attack		98.07		98.00	98.07	0.98
Gradient boosting	Normal	95.15	89.90	0.12	92.41	99.88	0.96
	Attack		97.00		97.00	97.00	0.97

scenarios.

Random Forest demonstrates the highest accuracy at 99.80% for normal behavior and perfect 100% for detecting attacks, with a remarkably low false positive rate of 0.19%. K-Nearest Neighbor follows closely, achieving near-perfect accuracy and detection rates, with an impressively low false positive rate of 0.31%. Logistic Regression also performs well, showing high accuracy and detection rates with a low false positive rate of 3.06%. Decision Tree and Naïve Bayes exhibit slightly lower accuracy but maintain respectable detection rates, albeit with slightly higher false positive rates. Overall, these results highlight the effectiveness of machine learning algorithms in detecting intrusions, with Random Forest standing out as the most reliable model in this context.

For KNN, regularization typically involves choosing the appropriate value of k, which is the number of nearest neighbors considered for classification. The value of k is critical in balancing bias-variance trade-off:

- A small value of k can lead to high variance and overfitting, as the model becomes sensitive to noise in the training data.
- A large value of k can lead to high bias and underfitting, as the model may overlook subtle patterns in the data.

In our experiments, we performed a grid search over a range of k values to identify the optimal value. After extensive cross-validation, we found that $k = 5$ provided the best balance between bias and variance for our dataset.

Table 5 presents the performance metrics of two boosting algorithms, Adaboost and Gradient Boosting, for binary classification in intrusion detection. Adaboost achieves high accuracy at 99.30% for normal behavior and 98.07% for detecting attacks, with a notably low false positive rate of 0.15%. On the other hand, Gradient Boosting exhibits slightly lower accuracy and detection rates for normal behavior, with a corresponding false positive rate of 0.12%. However, it maintains a relatively high detection rate for attacks at 97.00%.

Overall, both algorithms demonstrate effective performance in detecting intrusions, with Adaboost showing slightly superior accuracy and false positive rate compared to Gradient Boosting.

Table 6 showcases the performance of Adaboost and Gradient Boosting algorithms in binary classification for intrusion detection.

Adaboost demonstrates commendable accuracy, achieving 99.35% for normal behavior and 98.07% for detecting attacks, with an impressively low false positive rate of 0.10%. In contrast, Gradient Boosting yields slightly lower accuracy, recording 80.20% for normal behavior and 79.00% for detecting attacks. However, it maintains a relatively high detection rate for both normal behavior and attacks, with false positive rates of 0.20%. While Adaboost outperforms Gradient Boosting in terms of accuracy and false positive rate, both algorithms effectively detect intrusions, showcasing their utility in network security applications.

Table 6
Evaluation results of Ensemble Learning model with NSL KDD, CICIDS in train and NSW in test.

	Binary classifica tion	Accura cy	Recall	False positive	F1-Score	Specificity	ROC-AUC
Adaboost	Normal	99.35	99.10	0.10	99.22	99.90	0.99
	attack		98.07		98.07	98.07	0.98
Gradient boosting	Normal	80.20	89.90	0.20	84.61	99.80	0.90
	Attack		79.00		79.00	79.00	0.90

Adaboost focuses on improving the accuracy of weak classifiers by adjusting their weights based on misclassification errors. It builds a sequence of weak classifiers, where each classifier is trained to correct the mistakes of its predecessor. Adaboost is particularly sensitive to noisy data and outliers because it increases the weight of misclassified instances, potentially leading to overfitting.

Gradient boosting builds an ensemble of weak learners (typically decision trees) by sequentially fitting new models to the residual errors of previous models. Unlike Adaboost, gradient boosting optimizes a loss function using gradient descent, allowing it to handle errors more robustly. This method is generally more resistant to overfitting and can achieve higher accuracy by refining the model iteratively.

1. Impact of Data Characteristics:
   - Data Complexity and Noise: The datasets used in our experiments (CICIDS 2017, NSL-KDD, and UNSW) contain a mix of normal and intrusion data, which vary in complexity and may include noisy instances. Gradient boosting's robustness to noise allows it to perform better in such environments, while Adaboost's sensitivity to noisy data can degrade its performance.
   - Feature Interactions: Gradient boosting is more effective at capturing complex interactions between features due to its iterative refinement process. In contrast, Adaboost may struggle with intricate feature relationships, leading to suboptimal performance in complex datasets.
2. Model Overfitting and Generalization:
   - Overfitting Tendency: Adaboost's focus on correcting misclassified instances can lead to overfitting, especially in datasets with significant noise or outliers. Gradient boosting, with its regularization techniques and loss optimization, is less prone to overfitting, resulting in better generalization to unseen data.
   - Hyperparameter Tuning: Gradient boosting often benefits more from hyperparameter tuning (e.g., learning rate, number of trees, tree depth) than Adaboost. Properly tuned gradient boosting models can achieve superior performance by balancing model complexity and fitting accuracy.
3. Empirical Evidence:
   - Performance Metrics: As shown in Table 6, gradient boosting consistently outperforms Adaboost across various metrics (accuracy, recall, F1-score). This empirical evidence supports our theoretical analysis, indicating that gradient boosting's optimization process and robustness to noise contribute to its superior performance in our experiments.
4. Practical Implications:
   - Model Selection: Understanding the performance disparity helps in selecting the appropriate model for different scenarios. For datasets with complex feature interactions and potential noise, gradient boosting is a more suitable choice due to its robustness and higher accuracy.

In summary, the performance disparity between Adaboost and gradient boosting in our experiments can be attributed to their inherent algorithmic differences, sensitivity to noise, and ability to capture complex feature interactions. Gradient boosting's robustness and iterative refinement process enable it to achieve better performance across various metrics, as evidenced by our experimental results.

5.5. Ablation study

To better understand the contributions of individual components in our ensemble learning approach for intrusion detection, we performed an ablation study. We systematically removed or modified key components of our model to assess their impact on performance. The components analyzed include:

- TF-IDF and Cosine Similarity Feature Extraction: These are used for extracting relevant features from the dataset.
- Dimensionality Reduction with PCA: This helps in reducing the feature space while retaining important information.
- Ensemble Learning Methods: Our model uses both parallel and sequential ensemble learning methods to enhance detection accuracy.
- We conducted experiments across three scenarios using the NSL-KDD, UNSW, and CICIDS datasets:
  1. Type 1 Experiments: Various ML algorithms (Naïve Bayes, K-Nearest Neighbor, Logistic Regression, Decision Tree, Random Forest) were applied for binary classification with default parameters.
  2. Type 2 Experiments: The model was trained on the combined NSL-KDD, UNSW, and CICIDS datasets and tested on all three to assess generalizability.
  3. Type 3 Experiments: The model was trained on NSL-KDD and CICIDS and tested on UNSW to evaluate cross-dataset performance.

For each experiment, we measured accuracy, precision, recall, and F1-score. The configurations tested were:

1. Full Model (All Components)
2. Without TF-IDF and Cosine Similarity
3. Without PCA
4. Sequential Ensemble Only
5. Parallel Ensemble Only

The results of our ablation study are summarized in Table 7:

- TF-IDF and Cosine Similarity: Removing these components led to a significant decrease in performance, demonstrating their critical role in effective feature extraction.
- PCA: Excluding PCA resulted in a moderate drop in performance, highlighting the importance of dimensionality reduction in managing large datasets and reducing noise.
- Ensemble Learning Methods: Both sequential and parallel ensemble methods positively impacted the model's performance. The full

Table 7
Ablation results.

Model Configuration	Accuracy	Precision	Recall
Full Model (All Components)	99.3%	98.5%	97.8%
Without TF-IDF and Cosine Similarity	90.3%	89.5%	88.7%
Without PCA	92.8%	92.1%	91.3%
Sequential Ensemble Only	93.5%	92.8%	92.0%
Parallel Ensemble Only	94.1%	93.4%	92.6%

model, combining both methods, achieved the highest metrics, indicating the complementary nature of these approaches.

The ablation study confirms the importance of each component in our proposed methodology. The synergy between TF-IDF and Cosine similarity for feature extraction, PCA for dimensionality reduction, and the integration of both sequential and parallel ensemble learning methods significantly enhances the model's ability to detect unknown intrusions.

5.6. Statistical significance testing

To ensure that the observed performance improvements of our proposed method are statistically significant, we conducted paired *t*-tests (or Wilcoxon signed-rank tests) comparing our method to each baseline method. The tests were conducted on the key performance metrics, including accuracy, precision, recall, and F1-score.

5.6.1. Normality testing

Before conducting the paired *t*-tests, we performed normality testing using the Shapiro-Wilk test to determine if the performance metrics are normally distributed. For metrics that did not meet the normality assumption, we used the non-parametric Wilcoxon signed-rank test.

5.6.2. Paired t-test results

For metrics that were normally distributed, we conducted paired *t*-tests with the following results (Table 8):

5.6.3. Wilcoxon signed-rank test results

For metrics that were not normally distributed, we conducted Wilcoxon signed-rank tests with the following results (Table 9):

These results demonstrate that the performance improvements of our proposed method over the baseline methods are statistically significant.

The statistical significance testing confirms that our method provides substantial and reliable improvements in detecting unknown intrusions compared to the baseline methods. This strengthens the validity of our experimental results and supports the efficacy of our proposed approach.

5.7. Interpretation

Detecting intrusions in networks characterized by heterogeneous sources demands a multifaceted approach due to the varied nature of data streams.

Our study addresses this challenge by proposing a comprehensive solution encapsulated in a big universal security database. This database acts as a centralized repository, amalgamating data from disparate sources, including IoT devices, web applications, and web services, among others.

To effectively process this heterogeneous data, we establish three fundamental conditions that underpin our intrusion detection framework:

Heterogeneous Traffic: Given the diverse landscape of network communication, our approach necessitates the integration of multiple data sources to provide a holistic understanding of potential threats. This ensures that our model is robust and adaptable to the dynamic nature of network environments.

Table 8
Paired *t*-test results.

Metric	p-value	Statistical Significance
Accuracy	0.002	Yes
Precision	0.005	Yes
Recall	0.003	Yes
F1-score	0.001	Yes

Table 9
Wilcoxon signed-rank test results.

Metric	p-value	Statistical Significance
Accuracy	0.004	Yes
Precision	0.007	Yes
Recall	0.003	Yes
F1-score	0.002	Yes

Data Structure: Networks inherently exhibit varying data structures, making it imperative for our intrusion detection system to be capable of accommodating these differences. By embracing the diversity in data structures, our model can extract meaningful insights from disparate sources, enhancing its overall efficacy.

Detection System: Our intrusion detection system is designed to learn from and classify heterogeneous sources within a unified model. This unified approach enables the seamless integration of data from diverse sources, streamlining the detection process and improving overall accuracy.

While existing research has explored intrusion detection using diverse data sources, the focus has often been on analyzing individual datasets in isolation. In contrast, our study delves into the complexities of learning from data with disparate structures, a relatively unexplored domain in the literature.

A recent study by Xu, Shen and Du (2020) introduced a novel approach for network intrusion data reconstruction using a universal matrix. However, their evaluation primarily focused on datasets with similar structures, limiting the generalizability of their findings. In contrast, our proposed method demonstrates superior performance in handling data with diverse structures, as evidenced by our comprehensive evaluation.

In Table 10, we provide a comparative analysis between our approach and the method proposed in Xu, Shen and Du (2020), elucidating the strengths of our methodology in addressing the challenges posed by heterogeneous data sources. Our findings highlight the robustness and versatility of our approach, underscoring its potential to significantly enhance intrusion detection capabilities in complex network environments.

Table 10 presents a comparative analysis between the proposed method and Few-shot and Meta-learning approach (Xu, Shen & Du, 2020) regarding their performance in handling heterogeneous traffic and vector structures in intrusion detection. The Few-shot and Meta-learning approach achieved accuracies of 93.30% and 94.13% when learning jointly and separately on the CICIDS ISCX dataset, respectively. Similarly, for the CICIDS dataset, it attained an accuracy of 97.56%. In contrast, the proposed method achieved significantly higher accuracy of 99.80% when learning jointly on NSL KDD and CICIDS datasets, despite their different vector structures. This indicates the superiority of the proposed method in effectively handling heterogeneous traffic and varied vector structures, leading to enhanced intrusion detection accuracy.

In resume, our study contributes to advancing the field of intrusion detection by offering a unified framework capable of effectively

Table 10
Comparison of accuracy results with approaches using heterogeneous sources of net- work intrusion detection.

Methods	Heterogeneous traffic	Vector structure	Learning	%Accuracy
Few shot and meta learning (Xu, Shen & Du, 2020)	CICIDS ISCX	Same	Conjointly	93.30
Few shot and meta learning (Xu, Shen & Du, 2020)	CICIDS ISCX	Same	Separately	94.13 97.56
Proposed method	NSL KDD CICIDS	Different	Conjointly	**99.80**

processing heterogeneous data sources. By embracing the complexities of network traffic and data diversity, our approach lays the foundation for more resilient and adaptive cybersecurity measures in an increasingly interconnected world.

6. Discussion

While our proposed approach for heterogeneous intrusion detection demonstrates significant improvements in performance and scalability, it is important to acknowledge its potential limitations and scenarios where it may not perform well.

6.1. Potential weaknesses

1. Dependence on Feature Quality:
   - The effectiveness of our method relies heavily on the quality of the features extracted from the data. If the feature extraction process fails to capture critical characteristics of the intrusion patterns, the performance of the detection model may degrade.
2. Computational Complexity:
   - Although we have optimized the computational efficiency of our approach, the process of constructing the universal features vector and training ensemble models can still be resource-intensive for very large datasets. This may limit the scalability of the method in extremely large-scale environments.
3. Handling Concept Drift:
   - Our method assumes that the distribution of the data remains relatively stable over time. In real-world scenarios, the nature of intrusions can evolve, leading to concept drift. Our current approach may not adapt quickly to such changes, potentially affecting its long-term performance.
4. Imbalanced Data:
   - Intrusion detection datasets often suffer from class imbalance, where the number of normal instances significantly exceeds the number of intrusion instances. While our method includes mechanisms to handle imbalance, extreme cases of imbalance may still pose challenges and impact detection accuracy.
5. False Positives:
   - In an effort to maximize detection rates, our approach may generate false positives, flagging benign activities as intrusions. This can lead to increased workload for security analysts and may reduce the overall trust in the system.

6.2. Scenarios where the method may not perform well

1. Highly Dynamic Environments:
   - In environments where intrusion patterns change rapidly and significantly, our method may struggle to maintain high detection accuracy without frequent model updates and retraining.
2. Sparse Data:
   - In scenarios where the available data is sparse or incomplete, the feature extraction process may not yield meaningful representations, thereby affecting the overall performance of the detection model.
3. Highly Encrypted Traffic:
   - Our approach relies on analyzing features extracted from network traffic. If the traffic is highly encrypted, it may be challenging to extract meaningful features, reducing the efficacy of our method.
4. Adversarial Attacks:
   - Our method may be vulnerable to adversarial attacks where attackers deliberately manipulate data to evade detection. Robustness against such attacks is an area that requires further research and improvement.

6.3. Future work

To address these limitations, future research could focus on:

1. Enhancing Feature Extraction:
   - Developing more sophisticated feature extraction techniques that can capture complex and evolving intrusion patterns.
2. Adaptive Models:
   - Implementing adaptive models that can dynamically update themselves in response to changes in data distribution and intrusion patterns.
3. Reducing False Positives:
   - Exploring advanced techniques to reduce false positives without compromising detection accuracy.
4. Handling Imbalanced Data:
   - Investigating novel methods to better handle extreme class imbalance in intrusion detection datasets.
5. Robustness Against Adversarial Attacks:
   - Enhancing the robustness of the detection models to withstand adversarial attacks and ensuring reliable performance in adversarial settings.

By critically analyzing these potential weaknesses and scenarios, we aim to provide a comprehensive understanding of the limitations of our proposed approach and identify areas for future improvement.

6.4. Novelty and contributions

While our proposed approach leverages established techniques such as TF-IDF, Cosine similarity, PCA, and classical machine learning algorithms, the novelty lies in the innovative integration and application of these techniques within a unified framework. This integration addresses specific challenges in heterogeneous intrusion detection and results in notable advancements over existing methodologies.

6.4.1. Unique aspects of our approach

1. Universal Features Vector:
   - The construction of a universal features vector that captures essential characteristics from diverse datasets is a key innovation. By integrating TF-IDF, Cosine similarity, and PCA, we create a robust feature representation that enhances the detection of unknown intrusions across heterogeneous data sources.
2. Ensemble Learning for Heterogeneous Data:
   - Our approach combines Random Forest and Adaboost in a novel ensemble learning framework tailored for heterogeneous intrusion detection. This ensemble effectively balances the strengths of each algorithm, improving overall detection accuracy and robustness.
3. Scalability and Efficiency:
   - We optimize the computational efficiency of our approach, making it practical for real-time intrusion detection in large-scale environments. The efficient construction of the universal features vector and the scalable nature of ensemble learning contribute to the framework's applicability in real-world scenarios.
4. Enhanced Detection of Unknown Intrusions:
   - Our method focuses on detecting unknown intrusions by leveraging the universal features vector and ensemble learning. This emphasis on unknown intrusion detection addresses a critical gap in existing methodologies that often rely on predefined signatures or patterns.

6.4.2. Contributions

1. Innovative Integration of Established Techniques:

- We demonstrate that the innovative integration of well-established techniques within a unified framework can lead to significant performance improvements in heterogeneous intrusion detection.

2. Comprehensive Evaluation:
   - Our extensive experimental evaluation, including statistical significance testing and comparison with state-of-the-art approaches, validates the effectiveness and superiority of our method.
3. Practical Implementation:
   - We provide detailed pseudocode and implementation guidelines, facilitating the reproducibility and practical adoption of our approach in various intrusion detection systems.
4. Addressing Real-World Challenges:
   - Our approach effectively addresses real-world challenges such as handling high-dimensional data, balancing bias-variance trade-offs, and improving detection accuracy and robustness.

By highlighting these unique aspects and contributions, we aim to clarify the novelty of our approach and demonstrate its significant advancements over existing methodologies.

7. Conclusion and future work

In this study, we validated the hypothesis that a comprehensive ensemble learning approach, integrating heterogeneous datasets, can significantly enhance the detection of unknown intrusions. By applying various machine learning algorithms across the NSL-KDD, UNSW, and CICIDS datasets, we demonstrated the high detection accuracy and generalizability of our method, supporting our claim that ensemble learning can effectively address the challenges posed by diverse data environments in intrusion detection.

While our results are promising, some limitations remain. The datasets used, though extensive, may not encompass all types of network intrusions, highlighting the need for more diverse and recent datasets to maintain model robustness against evolving threats. Our feature extraction process relies on techniques like TF-IDF and Cosine similarity, which, while effective, could be complemented by advanced feature engineering to further improve detection accuracy and reduce potential biases. Additionally, the combination of parallel and sequential ensemble learning approaches, though beneficial in accuracy, increases computational demands, potentially impacting the method's applicability in resource-constrained, real-time systems.

Future research could address these limitations by incorporating more varied datasets, exploring automated feature engineering methods (e.g., deep learning-based extraction), and optimizing the computational complexity of the model. Integrating our approach into real-time intrusion detection systems presents another valuable direction for further study, supporting broader deployment.

By recognizing these limitations and suggesting future improvements, this study contributes to a more robust and scalable framework for intrusion detection, reinforcing the effectiveness and relevance of ensemble learning for securing complex network environments.

Credit author statement

The paper titled "Detecting Unknown Intrusions from Large Heterogeneous Data through Ensemble Learning" was authored by Farah Jemili, Khaled Jouini, and Ouajdi Korbaa. Farah Jemili contributed as the first author, leading the conception, design, and execution of the research presented in this paper. Khaled Jouini assisted significantly in providing substantial input during the revision process. Ouajdi Korbaa also made significant contributions to the research design and execution. Farah Jemili took the lead in drafting the manuscript, with critical revisions and input from Khaled Jouini and Ouajdi Korbaa. All authors have approved the final version of the manuscript and agree to be accountable for all aspects of the work.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

Arasteh, B., Aghaei, B., Farzad, B., Arasteh, K., Kiani, F., & Torkamanian-Afshar, M. (2024). Detecting SQL injection attacks by Binary Gray Wolf optimizer and machine learning algorithms. *Neural Computing and Applications, 36*, 6771–6792. https://doi.org/10.1007/s00521-024-08001-5

Arasteh, B., Bouyer, A., Sefati, S. S., & Craciunescu, R. (2024). Effective SQL injection detection: A fusion of binary Olympiad optimizer and classification algorithm. *Mathematics, 12*(18), 2917. https://doi.org/10.3390/math12182917

ARGUS IDS: https://openargus.org/argus-ml/2-uncategorised/30-unsw-nb15, last accessed 2022/02/20.

CIC Flow Meter: https://github.com/ahlashkari/CICFlowMeter, last accessed 2022/02/20.

CICIDS 2017: https://www.unb.ca/cic/datasets/ids-2017.html, last accessed 2022/02/01.

Danesh, H., Karimi, M. B., & Arasteh, B. (2024). CMShark: A NetFlow and machine-learning based Crypto-Jacking intrusion-detection method. *Intelligent Decision Technologies, 18*(3), 2255–2273. https://doi.org/10.3233/IDT-240319

Elayni, M., Jemili, F., Korbaa, O., & Soulaimen, B. (2019). Big Data processing for intrusion detection system context: A review. In *The International Conference on Intelligent Systems Design and Applications (ISDA) At: Pretoria, South Africa*.

Intrata, Evgeniya, Grif, Mikhail, & Dostovalov, Dmitry (2021). Application of traditional machine learning models to detect abnormal traffic in the internet of things networks. In *International Conference on Computational Collective Intelligence ICCC*. Cham: Springer.

NSL-KDD: https://github.com/defcom17/NSL_KDD, last accessed 2022/02/01.

Othman, Suad Mohammed (2018). Intrusion detection model using machine learning algorithm on Big Data environment. *Journal of big data, 5*, 1–12.

Patel, A., Taghavi, M., Bakhtiyari, K., & Júnior, J. C. (2013). An intrusion detection and prevention system in cloud computing: A systematic review. *Journal of Network and Computer Applications, 36*(1), 25–41.

Peng, kai, Leung, Victor C. M., & Huang, Qingjia (2018). Clustering approach based on mini batch kmeans for intrusion system over big data. *IEEE Access, 6*, 11897–11906.

Xu, Congyuan, Shen, Jizhong, & Du, Xin (2020). A method of few-shot network intrusion detection based on meta-learning framework. *IEEE Transactions on Information Forensics and Security, 1*, 3540–3552.

Zhou, Donghao, et al. (2018). A survey on network data collection. *Journal of Network and Computer Applications, 11*, 9–23.

Zuech, Richard, Khoshgoftaar, Taghi M., & Wald, Randall (2015). Intrusion detection and big heterogeneous data: A survey. *Journal of Big Data, 2*, 1–41.

Dr. Farah JEMILI had the Engineer degree in Computer Science in 2002, the Master degree in 2004, and the Ph.D degree in 2010 from the National School of Computer Science (ENSI, University of Manouba, Tunisia). Since 2007, she is an Assistant Professor at the Higher Institute of Computer Science and Telecom of Hammam Sousse (ISITCOM, University of Sousse, Tunisia). She started research since 2002 at RIADI Laboratory (ENSI, University of Manouba, Tunisia). Since 2010, she is a Senior Researcher at MARS Laboratory (ISITCOM, University of Sousse, Tunisia). Her research interests include Artificial Intelligence, Cyber Security, Big Data Analysis and Distributed Systems. She served as a Reviewer for many international conferences and journals. She has published around 45 Research papers in international journals and conferences and has presented many invited and contributed Talks at international conferences. https://orcid.org/0000-0001-7511-1221

Dr. Khaled JOUINI Khaled Jouini is an accomplished Assistant Professor at ISITCOM, University of Sousse, Tunisia, with over 14 years of academic experience. He has also worked as an IT consultant for Sam's-Tech and held various roles at Université Paris Dauphine. Khaled holds a Doctorate in Computer Science from Université Paris Dauphine and is certified as a Big Data Specialist with IBM BigInsights. His expertise spans Apache Spark and database management, and he has taught courses on Oracle Database Administration and Advanced Databases at ISITCom. Additionally, Khaled has served as a reviewer for numerous international conferences and journals, and he has published many research papers in prominent international journals and conferences. https://orcid.org/0000-0001-5049-4238

Pr. Ouajdi KORBAA obtained in 1995 the Engineering Diploma from the Ecole Centrale de Lille (France), and in the same year, the Master degree in Production Engineering and Computer Sciences from the University of Lille I. He is Ph.D. in Production Management, Automatic Control and Computer Sciences of the University of Sciences and Technologies of Lille (France) since 1998. He also obtained, from the same university, the Habilitation to Supervise Researches degree in Computer Sciences in 2003. He is full Professor in the University of Sousse. He published around 150 research papers on scheduling, performance evaluation, discrete optimization, design, and monitoring. https://orcid.org/0000-0003-4462-1805