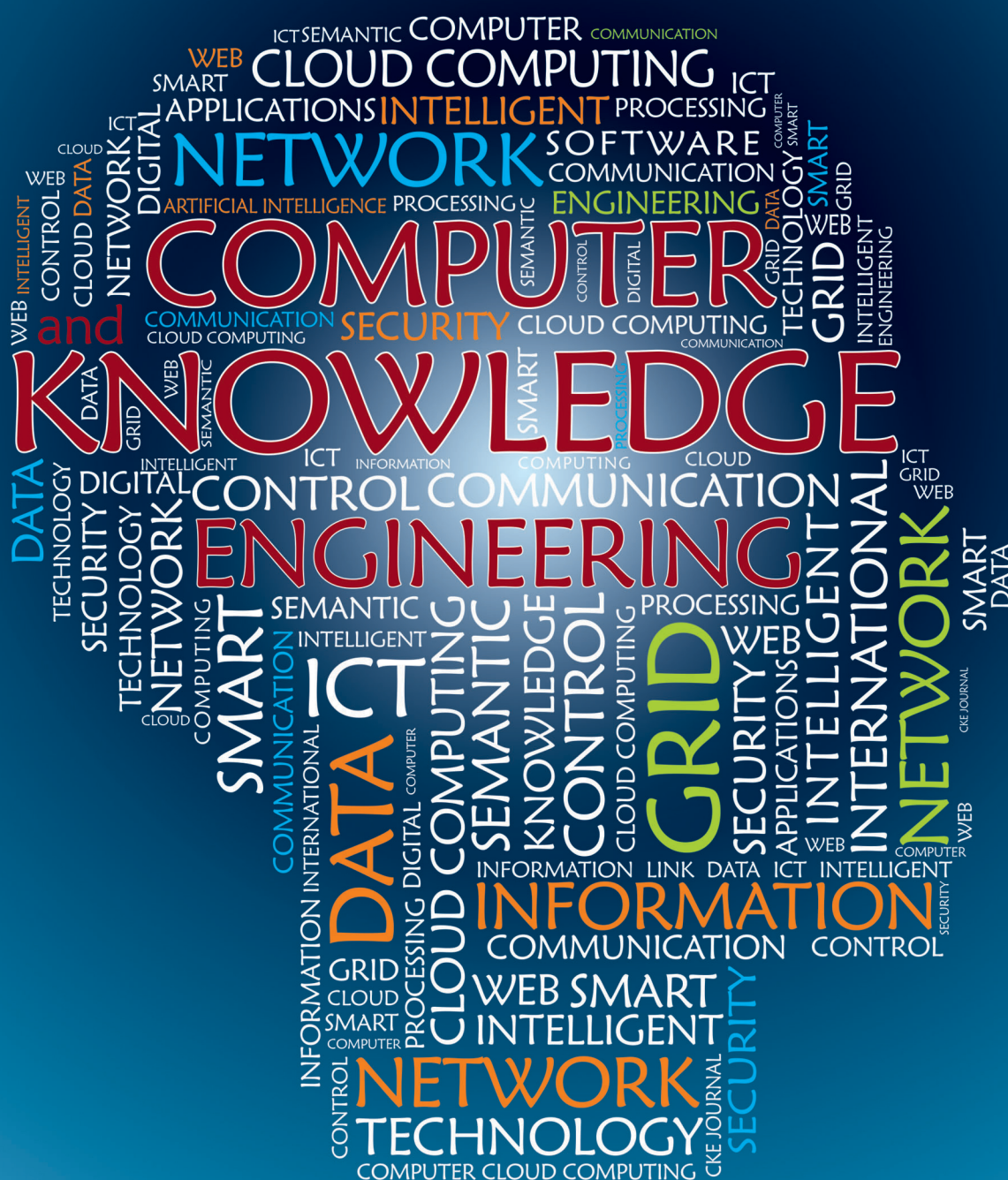


ISSN: 2717-4123





Journal of
**COMPUTER AND KNOWLEDGE
ENGINEERING**

Ferdowsi University of Mashhad

ISSN: 2717-4123

General Director: S. A. Hosseini Seno

Editor-in-Chief: M. Kahani

Publisher: Ferdowsi University of Mashhad

Editorial Board:

Mahmoud Naghibzadeh	Professor	Ferdowsi University of Mashhad, Iran
Mohammad H Yaghmaee-Moghaddam	Professor	Ferdowsi University of Mashhad, Iran
Dick H Epema	Professor	Delft Technical University, the Netherlands
Rahmat Budiarto	Professor	University Utara Malaysia, Malaysia
Mohsen Kahani	Professor	Ferdowsi University of Mashhad, Iran
Mohammad R Akbarzadeh-Tootoonchi	Professor	Ferdowsi University of Mashhad, Iran
Madjid Fathi	Professor	University of Siegen, Germany
Hossein Nezamabadi-pour	Professor	Bahonar University of Kerman, Iran
Ahmad Ghafarian	Professor	University of North Georgia, USA
Hamid Reza Pourreza	Professor	Ferdowsi University of Mashhad, Iran
Hadi Sadoghi-Yazdi	Professor	Ferdowsi University of Mashhad, Iran
Seyed Amin Hosseini Seno	Associate Professor	Ferdowsi University of Mashhad, Iran
Abedin Vahedian-Mazloun	Associate Professor	Ferdowsi University of Mashhad, Iran
Ebrahim Bagheri	Associate Professor	Ryerson University, Canada
Hossein Asadi	Associate Professor	Sharif University of Technology, Iran
Mahdi Kargahi	Associate Professor	University of Tehran, Iran
Hamid Reza Ekbia	Associate Professor	Indiana University, USA
Seyed Hassan Mirian Hosseinabadi	Associate Professor	Sharif University of Technology, Iran
Abbas Ghaemi Bafghi	Associate Professor	Ferdowsi University of Mashhad, Iran
Farhad Mahdipour	Associate Professor	Kyushu University, Japan

Administrative Director: T. Hooshmand

Journal of Computer and Knowledge Engineering

Faculty of Engineering, Ferdowsi University of Mashhad

P. O. Box. 91775-1111, Mashhad, I.R. IRAN

Tel: +98 51 38806024, Fax: +98 51 38763301, Email: cke@um.ac.ir, Site: cke.um.ac.ir

CONTENTS

Enhancing Channel Selection in 5G with Decentralized Federated Multi-Agent Deep Reinforcement Learning	1
Taghi Shahgholi- Keyhan Khamforoosh- Amir Sheikahmadi- Sadoon Azizi	
An Enhanced Sine Cosine Algorithm for Feature Selection in Network Intrusion Detection	17
Zahra Asghari Varzaneh- Soodeh Hosseini	
Evaluating Developers' Expertise in Serverless Functions by Mining Activities from Multiple Platforms	27
Aref Talebzadeh Bardsiri- Abbas Rasoolzadegan	
Description-based Post-hoc Explanation for Twitter List Recommendations	43
Havva Alizadeh Noughabi- Behshid Behkamal- Mohsen Kahani	
Machine Learning Classifiers and Data Synthesis Techniques to Tackle with Highly Imbalanced COVID-19 Data	51
Avaz Naghipour- Mohammad Reza Abbaszadeh Babil Soflaei- Mostafa Ghaderi-Zefrehei	
Anomaly Detection in IoMT Environment Based on Machine Learning: an Overview	65
Peyman Vafadoost Sabzevar- Hamidreza Rokhsati- Alireza Chamansara- Ahmad Hajipour	



Ferdowsi
University of
Mashhad

Journal of Computer and Knowledge Engineering

<https://cke.um.ac.ir>



Information and
Communication
Technology Association of
Iran

Enhancing Channel Selection in 5G with Decentralized Federated Multi-Agent Deep Reinforcement Learning*

Research Article

Taghi Shahgholi¹, Keyhan Khamforoosh² , Amir Sheikahmadi³, Sadoon Azizi⁴

DOI: [10.22067/cke.2024.88900.1119](https://doi.org/10.22067/cke.2024.88900.1119)

Abstract The increasing popularity of vehicular communication systems necessitates efficient and autonomous decision-making to address the challenges of vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications. In this paper, we present a comprehensive study on channelization in Cellular Vehicle-to-Everything (C-V2X) communication and propose a novel two-layer multi-agent approach that integrates deep reinforcement learning (DRL) and federated learning (FL) to enhance the decision-making process in channel utilization.

Our approach leverages the autonomy of each vehicle, treating it as an independent agent capable of making channel selection decisions based on its local observations in its own cluster. Simultaneously, a centralized architecture coordinates nearby vehicles to optimize overall system performance. The DRL-based decision-making model considers crucial factors, such as instantaneous channel state information and historical link selections, to dynamically allocate channels and transmission power, leading to improved system efficiency.

By incorporating federated learning, we enable knowledge sharing and synchronization among the decentralized vehicular agents. This collaborative approach harnesses the collective intelligence of the network, empowering each agent to gain insights into the broader network dynamics beyond its limited observations. The results of our extensive simulations demonstrate the superiority of the proposed approach over existing methods, as it achieves higher data rates, success rates, and superior interference mitigation.

Keyword C-V2X Optimization, Multi-Agent Learning, DRL-based Channel Access, Federated Learning Integration

1. Introduction

The Fifth Generation (5G) network has successfully transitioned into the commercial stage and is currently being swiftly deployed worldwide. Simultaneously, the proliferation of mobile devices and interactive services has resulted in a substantial surge in data traffic and user demands. Alongside human-centric communications, the prevalence of machine-to-machine (M2M) terminals is expected to escalate significantly, nearing saturation by the year 2030. Projections indicate that the number of cellular M2M terminals will reach a staggering 100 billion in 2030, approximately 10 times the figure in 2022, including more than 900 million connected cars [1].

In recent decades, the exponential increase in the number of vehicles has given rise to a range of critical issues, including traffic safety, urban congestion, and environmental pollution. In response to these challenges, there is a growing focus on establishing a transportation ecosystem that is safe, efficient, and sustainable with utilizing technologies such as 5G, autonomous and connected vehicle.

For connected vehicle, several technical solutions have been put forward where Cellular-V2X or C-V2X stands out for its ability to provide superior coverage and quality-of-service (QoS) compared to other alternatives [2]. Furthermore, by integrating advanced technologies such as millimeter-wave communication and nonorthogonal multiple access the performance of cellular V2X can be further enhanced [3]-[5]. Ensuring real-time and reliable communication for safety-critical messages poses challenges for existing centralized resource allocation in cellular networks, mainly due to diverse quality-of-service

* Manuscript received: 2024 July 11, Revised, 2024 October 6, Accepted, 2024 October 21.

¹ Ph.D. Student Department of Computer Engineering, Sanandaj Branch, Islamic Azad University, Sanandaj, Iran.

² Corresponding Author, Assistant Professor, Department of Computer Engineering, Sanandaj Branch, Islamic Azad University, Sanandaj, Iran, **Email:** k.khamforoosh@iausdj.ac.ir

³ Associate Professor, Department of Computer Engineering, Sanandaj Branch, Islamic Azad University, Sanandaj, Iran.

⁴ Associate Professor, Department of Computer Engineering, University of Kurdistan, Sanandaj, Iran.

(QoS) requirements such as ultra-reliability and low-latency.

To address these challenges, 3GPP⁵ has investigated advanced resource allocation approaches for Cellular Vehicle-to-Everything (C-V2X). These approaches involve assigning independent packet priority levels to vehicular applications based on their latency and reliability requirements. Additionally, sensing-based decentralized methods have been proposed to select resource blocks with lower interference for transmission. However, it is important to note that these approaches primarily focus on dedicated resource pools and may overlook potential interference between Vehicle-to-Infrastructure (V2I) and Vehicle-to-Vehicle (V2V) communications within shared resource pools [2],[6],[7].

Machine learning has gained attention for improving C-V2X by addressing resource allocation problems and optimizing channel utilization [8]-[11]. In this paper, we have introduced a novel decentralized approach with combining federated learning and reinforcement learning for channel selection in 5G NR C-V2X connected vehicle environments enables efficient utilization of resources, enhancing system performance in terms of latency, reliability, and spectral efficiency. Our approach addresses the challenge of selecting reliable and interference-free channels, even with a high volume of vehicles. We evaluate our approach in simulation and show that it outperforms existing approaches.

The contributions of this paper that distinguish it from past works are listed below:

- Two-layer multi-agent approach: individual vehicles as agents and clusters of vehicles as higher-level coordinators
- Deep reinforcement learning (DRL) for decision-making, considering factors like CSI, queue backlog, interference, and historical selections
- Federated learning (FL) for knowledge sharing and synchronization among decentralized vehicular agents

Advantages of the Proposed Approach are Real-time adaptation and optimization of actions, utilization of collective intelligence for more informed decisions and Efficient channel utilization, minimized interference, and enhanced performance and reliability.

The subsequent sections of the paper are organized as follows: Section II provides an in-depth exploration of the background and related work, setting the foundation for our research. In Section III, we present the system model, outlining the key components and architecture of our proposed approach. Section IV delves into the problem formulation and defining the objectives of our research. The simulation and results are detailed in Section V, where we present the outcomes of our results and evaluate the performance of our proposed approach. Finally, in Section VI, we draw conclusions based on our findings, highlighting the significance of our approach.

2. Background and Related work

This paper focuses on the application of federated learning and deep reinforcement learning for channel utilization in 5G NR C-V2X. Firstly, we will provide an overview of the

3GPP standard and the channel access mechanism in 5G NR C-V2X. Next, we will delve into the use of deep reinforcement learning for resource allocation. Additionally, we will explore a novel federated learning approach and its application in channel utilization. Given the significant interest in machine learning and its application across various technologies, we conducted an extensive background and related work research to identify existing gaps in the field. By thoroughly examining the literature, we aimed to gain a comprehensive understanding of the current state-of-the-art and identify areas where further research and contributions are needed.

In Release 12, 3GPP introduced direct Device-to-Device (D2D) communications for proximity services (ProSe) using cellular technologies [12]. LTE V2X, based on the LTE air interface, was developed under Release 14 (Rel. 14) and further enhanced in Release 15 (Rel. 15). The 5G NR (New Radio) air interface served as the foundation for the development of a new cellular V2X standard under Release 16 (Rel. 16) [13].

The 5G NR standard, developed under Rel. 15, did not include sidelink (SL) aspects, which refer to direct communication between terminal nodes or User Equipment (UEs) without involving the network. However, Rel. 16 introduced V2X communications, including SL communications, based on the 5G NR air interface. This marked the availability of the first 5G NR V2X standard, focusing on connected and automated driving use cases. The goal of NR V2X SL is to support enhanced V2X (eV2X) use cases that have specific requirements not fulfilled by the LTE V2X standard.

In Release 12, two modes were defined for UE (User Equipment) transmission scheduling in V2X communications: Mode 1 and Mode 2. In Mode 1, when the UE is within the coverage of the eNB (evolved NodeB), centralized scheduling occurs at the eNB. On the other hand, in Mode 2, for D2D communication scheduling, the UE selects a radio resource from a pool configured by the cellular network or pre-configured in the UE itself to use the PC5 interface for direct communication.

Both Mode 1 and Mode 2 have a similar resource allocation structure. The data transmission is scheduled in a period called the Sidelink control period, which consists of two sets of sub-frames: Physical Sidelink Shared Channel (PSSCH) and Physical Sidelink Control Channel (PSCCH). The PSCCH is always transmitted before the PSSCH transmission to inform the receiver about the occupation of the PSSCH radio resources. This information is included in a PSCCH scheduling assignment called Sidelink Control Information (SCI). These mechanisms were designed considering the battery life of mobile devices.

However, for connected vehicle communications, different requirements need to be considered, such as latency, which D2D ProSe (Proximity Services) could not meet. Therefore, in Release 14, 3GPP introduced two new modes, Mode 3 and Mode 4, for C-V2X (Cellular Vehicle-to-Everything) to improve D2D ProSe performance.

⁵ the 3rd Generation Partnership Project

In Mode 3, similar to Mode 1, provides centralized scheduling, ensuring efficient resource utilization, but it requires vehicles to be within network coverage and introduces cellular uplink and downlink signaling overhead. Mode 4, on the other hand, enables vehicles to operate outside network coverage and make independent sub-channel selections using the sensing based SPS scheme where vehicles utilize sensing techniques to identify and select suitable sub-channels for transmission [16].

However, the PSCCH and PSSCH allocation in Mode 3 and 4 are completely different from Mode 1 and 2. In Mode 3 and 4, the resources are divided into sub-channels, and the first resource blocks are PSCCH pools, while the rest of the resource blocks are PSSCH for data transmission (Transport Blocks).

In Mode 4, UEs can select their sub-channels using a new mechanism called sensing-based semi-persistent scheduling, which significantly improves the estimation of available sub-channels.

For the access mechanism in C-V2X, it supports 10 and 20 MHz channels with Single Carrier Frequency Division Multiple Access (SC-FDMA). The channels are divided into Resource Blocks (RBs), sub-channels, and sub-frames. Resource blocks are 180 kHz wide in frequency and consist of 12 sub-carriers of 15 kHz. The sub-frames are defined as 1ms long, and a sub-channel is a group of resource blocks in the same sub-frame. In C-V2X, there are two sub-channelization schemes: Adjacent PSCCH + PSSCH and Nonadjacent PSCCH + PSSCH. In the adjacent scheme, the Sidelink Control Information (SCI) and its associated Transport Block (TB) are in adjacent resource blocks. The first two resource blocks of the first sub-channel are used for the SCI, while the transport block occupies several sub-channels in the next resource blocks. In the nonadjacent scheme, resource blocks are divided into pools, with a dedicated pool for transmission of the SCIs and other pools used for TBs transmissions. In Sidelink communications, each vehicle selects a transmission resource block without communicating with the Base Station (BS) and directly sends data to other vehicles.

During the SPS process, a vehicle first senses the transmissions in its vicinity to assess the availability and quality of different sub-channels. Based on this information, the vehicle identifies candidate resources within a designated Selection Window (SW). The SW includes a range of subframes where the vehicle can find sub-channels that can accommodate its transmission. Once the candidate resources are identified, the vehicle excludes specific resources based on the sensed interference or other criteria. The remaining sub-channels within the SW are then considered for transmission. The vehicle reserves these selected sub-channels for its subsequent transmissions using the Resource Reservation Interval (RRI) included in the Sidelink Control Information (SCI). This approach allows for efficient utilization of available sub-channels and helps mitigate interference in V2X Sidelink communications [17].

While SPS is a simple and effective method, it has several limitations including lower QoS in higher density, faces challenges in handling packet collisions due to

imprecise sensing results caused by the hidden-terminal problem, the half-duplex constraint preventing the detection of other vehicles using the same resources, the increased likelihood of collisions in high-density scenarios, etc.

In this paper, we propose a novel approach to improve the resource allocation process in C-V2X Mode 4, where we integrate the clustering technique, federated learning, and multi-agent deep reinforcement learning algorithm into SPS, enabling vehicles to intelligently select radio resources and avoid resource conflicts. This approach leverages federated learning and clustering techniques to enhance resource allocation performance in C-V2X communications.

To overcome the SPS limitations, in recent years, novel resource allocation schemes in C-V2X communications has been explored. These schemes employ both centralized and decentralized approaches, with a primary focus on optimizing parameter configurations, enhancing the resource sensing process, and improving the resource allocation process.

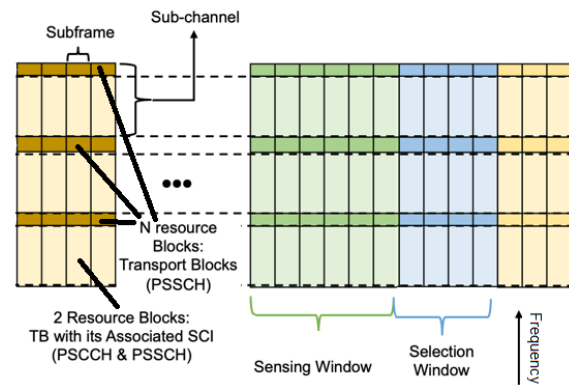


Figure 1. summarizes the channel resource management in C-V2X standard.

For centralized approach, in [15]- [18], a power control algorithm was developed based on spatiotemporal traffic patterns. This algorithm aims to satisfy the delay and reliability requirements of V2V services while reducing the overhead of periodic Channel State Information (CSI) reports to the Base Station (BS). Graph theory has also been utilized in [19]-[21] to enhance system throughput. In the study [19], the ergodic capacity of Vehicle-to-Infrastructure (V2I) communications and the reliability of V2V communications were analyzed by considering the statistics of fast fading components. Based on these analyses, the researchers proposed centralized resource allocation and power control algorithms to meet diverse QoS requirements.

In [22], the authors addressed the challenge of channel uncertainty caused by delayed Channel State Information (CSI) feedback to the Base Station (BS). They analyzed the correlation of fast-changing channels and proposed a joint channel allocation and power control algorithm. The objective was to maximize system throughput while meeting the delay and reliability requirements of each V2V link. However, these existing algorithms rely on V2V links reporting their local information, which can result in significant signaling overhead as the number of vehicles

increases. Moreover, resource allocation has been formulated as combinatorial optimization problems with nonlinear constraints, posing challenges for traditional optimization methods.

To overcome the centralized problems, recently, several studied proposed decentralized approaches. The authors in [23] introduced a decentralized resource allocation approach for vehicle-to-vehicle communications. In this approach, each V2V transmitter acted as an independent agent and made decisions autonomously based on local observations. The system was able to optimize resource allocation in a decentralized manner, considering the unique characteristics and requirements of each V2V transmitter.

In [24], the authors propose a novel semi-distributed transmission paradigm for NR V2X to achieve high reliability. It includes distributed clustering, autonomous inter-cluster resource selection, and centralized intra-cluster communication. In [25], the authors focused on reducing transmission failures and designed a distributed network coding-based medium access control protocol (NC-MAC) for reliable V2V beacon broadcasting. By combining preamble-based feedback, retransmissions, and network coding, the NC-MAC protocol enhances broadcasting reliability.

These studied have primarily focused on short-term optimization of resource allocation, overlooking the potential long-term performance gains. It is essential to consider a broader perspective and incorporate long-term strategies into resource allocation algorithms to achieve sustained performance improvement over time. By considering factors such as network dynamics, future traffic patterns, and system scalability, resource allocation approaches can be designed to optimize not only immediate resource allocation decisions but also their impact on overall network performance in the long run.

To address this challenge, researchers have turned to machine learning techniques, especially deep reinforcement learning (DRL) and more recently federated learning. These approaches offer effective solutions for tackling sequential decision-making problems. By combining deep learning and reinforcement learning, DRL algorithms enable the learning of optimal strategies in complex environments with long-term consequences for actions. Federated learning, on the other hand, allows distributed devices to collaboratively learn from their local data without sharing it centrally. These powerful techniques hold promise for addressing various challenges and optimizing decision-making processes in complex environments such as connected vehicles.

Deep Reinforcement Learning have gained significant traction in the field of wireless communications as they provide effective solutions to the challenges encountered by traditional optimization methods specially for resource allocation [26].

In [27], the authors introduced the C-Decision architecture for resource allocation in V2X networks. It combines centralized decision making and distributed resource sharing to maximize the sum rate. Vehicles compress their information using deep neural networks and send it to the centralized decision unit. The decision unit employs a deep Q-network for resource allocation and

balances V2V and V2I links. To overcome the problem of high collision probability in conventional SPS with a fixed reservation process during high traffic density, [28] proposes a Q-learning based SPS (Q-SPS) algorithm. Q-SPS intelligently adjusts the reservation probability using reward feedback, adapting to the dynamic C-V2X network environment. However, this approach deviates from the fundamental assumption of RL, which requires a stationary environment. In this case, a single vehicle is unable to update the evolving policies of other vehicles, thereby compromising the effectiveness of the approach [29][30].

In [31], the authors employed the DRL algorithm as a means to allocate resource blocks (RBs) and minimize signal collisions during transmissions. However, this approach did not consider the heterogeneous nature of quality-of-service (QoS) requirements across different types of messages where various message types may have distinct QoS demands, such as latency, reliability, and priority. Ignoring these varying requirements could lead to suboptimal resource allocation decisions and potential degradation of overall network performance.

Several studies have focused on addressing these challenges through the implementation of multi-agent DRL techniques. In [32], authors address resource allocation challenges in V2X communications and proposes two algorithms. The first algorithm uses deep reinforcement learning (DRL) with deep Q-network (DQN) and deep deterministic policy-gradient (DDPG) to improve performance for V2I and V2V links. The second algorithm, based on meta-learning, enhances adaptability to dynamic environments. [33] focuses on spectrum allocation in V2X networks using a graph representation. A graph neural network (GNN) is employed to extract low-dimensional features from the graph. Multi-agent RL is then used to allocate spectrum based on the learned features and deep Q-network is utilized for optimizing the sum capacity of the V2X network. Several other studies have employed different types of DRL algorithms and proposing scheme utilizing multiagent deep deterministic policy gradient, proximal policy optimization (PPO)-based multi-agent reinforcement, deep deterministic policy-gradient (DDPG), long short-term memory (LSTM) etc. to optimize the allocation of resources in V2X communications [34]-[39].

While DRL approaches have shown promise for resource allocation in vehicular networks, there are challenges to consider regarding training efficiency, particularly in highly dynamic and large-scale environments. The complexity and rapid changes in network conditions pose difficulties in achieving fast and accurate convergence during the training process. Addressing these issues is crucial to ensure the practicality and scalability of DRL-based resource allocation algorithms in real-world vehicular communication scenarios.

Moreover, using a fully distributed DRL method can lead to convergence at local optima, while fully centralized DRL methods are not suitable for vehicular networks due to the significant delay caused by information exchange with central nodes, particularly for delay-sensitive applications. Furthermore, the computational complexity

of centralized DRL increases substantially with a larger number of vehicles. Hence, a more viable approach is to combine the strengths of centralized and distributed DRL algorithms to effectively support direct communications in vehicular networks.

To address the limitations of traditional DRL approaches, researchers have recently turned to the emerging technique of federated learning [40] for resource allocation problems. Federated learning enables collaborative learning across multiple decentralized devices or nodes without the need to share raw data. The application of federated learning in resource allocation holds great promise in improving the performance and adaptability of wireless networks, as it leverages the collective intelligence of distributed devices to optimize resource allocation decisions.

The process of federated learning consists of three steps: First, the FL server determines the training task and distributes the initial global model to selected distributed devices. These devices then utilize their local data to train their individual models, aiming to minimize the loss function based on the initial global model. After several rounds of local training, the devices upload their local models to the FL server. Finally, the FL server aggregates these local models and sends back the updated model to the data owner. This process is repeated until the global loss function converges or a desired training accuracy is achieved. By conducting local model training on decentralized devices using their own raw data and infrequent model aggregation at the centralized server, federated learning significantly enhances the performance of model training [41]-[43].

Federated learning is a promising approach that has gained attention in various domains. However, its application in C-V2X resource allocation is still relatively new, and there are limited studies exploring its potential in this context. Authors in [44] introduce federated learning into a MEC-assisted vehicular network framework with focuses on participant selection, computing resource allocation optimization, and a distributed computing resource allocation method.

Authors in [45] propose a deep reinforcement learning (DRL)-based federated learning (FL) approach for decentralized resource allocation in an underlay mode D2D-enabled wireless network. The aim is to maximize sum capacity, minimize power consumption, and ensure quality of service (QoS) for both cellular and D2D users. Furthermore, a joint optimization problem involving transmission mode selection and resource allocation is investigated in [2], and formulated as a Markov decision process. The authors also proposed a DRL-based decentralized algorithm to maximize the sum capacity of V2I users while meeting latency and reliability requirements for V2V pairs. Also, to overcome training limitations, a two-timescale federated DRL algorithm is introduced, utilizing a graph theory-based vehicle clustering algorithm on a large timescale and federated learning on a small timescale.

None of the mentioned studied have not taken into account the scenario of shared spectrum in V2V communication, where the cellular channels are already assigned, and the traffic is highly congested. Authors in

[46] proposed approach is a federated multi-agent deep reinforcement learning (FedMARL) method that optimizes channel selection and power control for V2V communication. By leveraging both deep reinforcement learning (DRL) and federated learning (FL), the approach ensures reliability, delay requirements, and maximizes cellular link transmit rates. Individual V2V agents are constructed using the dueling double deep Q-network (D3QN) and trained collaboratively with a designed reward function. Federated learning is incorporated to address training instability in the multi-agent environment. An important limitation of this study is that it relied on static and pre-defined resources for decentralized channel access. The use of fixed resources may not effectively adapt to dynamic channel conditions or varying traffic demands in real-time.

3. System Model

A. Network model

The system model we consider is a network consisting of various clusters of vehicles, including Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I), Roadside Units (RSUs), and a Base Station (BS) as shown in Fig. 2. Each cluster is denoted as $C = c_1, c_2, \dots, c_K$, with K representing the total number of clusters in the network. Within each cluster C_k , there is a set of vehicles denoted as $V_k = v_1, v_2, \dots, v_{n_k}$ where n_k is the total number of vehicles in cluster C_k . The set of RSUs is denoted as $\mathcal{I} = j_1, j_2, \dots, j_m$ with m representing the total number of infrastructure units.

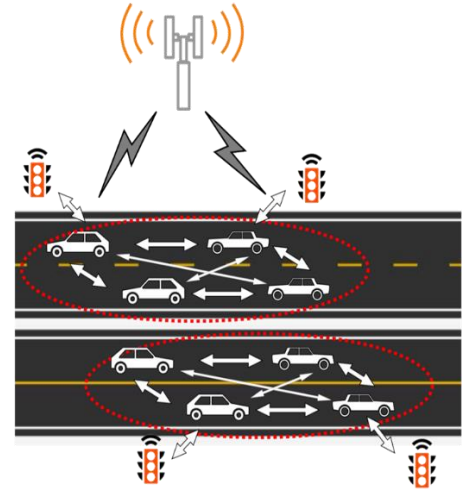


Figure 1. High-level System Model

In this network model, the time is divided into slots, indexed by $t = 1, 2, \dots$, where each slot has a duration of τ . This slotted communication system provides a structured framework for organizing and managing the transmission and reception of data in the system. In our model, each base station serves T_c number of cellular users, which include vehicles and RSUs. The selected users within each cluster communicate with the base station using allocated cellular channels. However, within the clusters, V2V and V2I communication take place by reusing the allocated channels. This enables efficient utilization of resources and facilitates direct communication between vehicles and infrastructure units within the same cluster.

In our network model, we consider the channel power gains between different links and the interference could cause when they use same channel. Specifically, the channel power gain from the transmitter of the $\alpha - th$ V2I link to the BS over the $f - th$ RB outside the cluster is denoted as $g_{\alpha,B}[f]$.

We can define the SINR of the $\alpha - th$ V2I link at the BS, and $\beta - th$ V2V link at the $f - th$ RB and in time slot i as [19]:

$$SINR_{i,\alpha,f} = \frac{P_{i,\alpha,f} \times g_{\alpha,B}[f]}{\sigma^2 + \sum_j (\rho_{j,\beta,f} \times P_{j,\beta,f} \times g_{\beta,B}[f])} \quad (1)$$

where the $\rho_{j,\beta}$ is the spectrum allocation indicator where $\rho_{\beta,f} = 1$ means the $\alpha - th$ V2I links is transmitting over the $f - th$ RB and $\rho_{j,\beta} = 0$ means it does not transmit

Similarly, for $\beta - th$ V2V link at the V2V receiver over the $f - th$ RB we can define:

$$SINR_{j,\beta,f} = \frac{P_{j,\beta,f} \times g_{\beta,B}[f]}{\sigma^2 + \sum_j (\rho_{j,\beta,f} \times P_{j,\beta,f} \times g_{j,\beta}[f]) + \sum_{\gamma} (\rho_{\gamma,\beta} \times P_{\gamma,f} \times g_{\gamma,B}[f])} \quad (2)$$

In these equations, $P_{i,\alpha,f}$ and $P_{j,\beta,f}$ represent the transmit powers of the $\alpha - th$ V2I and the $\beta - th$ V2V transmitter over the $f - th$ RB, respectively. The indicator $\rho_{c,m,f} \in \{0,1\}$ is the spectrum allocation indicator, where $\rho_{i,\alpha,f} = 1$ implies that the $\alpha - th$ V2I link is transmitting over the $f - th$ RB, and $\rho_{i,\alpha,f} = 0$ otherwise. Similarly, the spectrum allocation indicator for the $\beta - th$ V2V link, $\rho_{j,\beta,f}$ is defined in a similar manner. In addition, the $P_{\gamma,f}$ represent the interfering channel from the $\gamma - th$ V2V transmitter to the $\beta - th$ V2V receiver over the $f - th$ RB.

B. QoS Requirements

- Delay for V2V Pairs

In our system, each transmitter of a V2V link is equipped with a finite-length buffer, and safety-related packets are generated at a constant rate λ (bits/s). However, due to varying transmit rates $R_{tk} = W \log_2(1 + \gamma_{tk})$ at different time slots, there can be a mismatch between packet generation and instantaneous throughput. Consequently, queues can build up at the transmitters of V2V pairs, resulting in increased queuing delays.

At the beginning of time slot t , the queue length of the $\beta - th$ V2V pair, denoted by $Q_{t\beta}$, is determined by the following equation [46]:

$$Q_{t\beta} = \max(0, Q_{t-1,\beta} + \tau\lambda - \tau R_{t-1,\beta}) \quad (3)$$

where $Q_{t-1,\beta}$ is the queue length at the transmitter in the previous time slot ($t - 1$), $\tau\lambda$ represents the number of bits arrived at the queue per slot, and $\tau R_{t-1,\beta}$ denotes the number of bits sent to the corresponding receiver in the previous time slot ($t - 1$).

We focus on the queuing delay as it dominates the delay over a V2V link. Based on Little's Law, the average queuing delay is proportional to the queue length. Let D_{max} present the tolerable transmission delay for V2V packets, and the number of packets experiencing delays longer than D_{max} given by $Q_{max} = \lambda D_{max}$. Therefore, the delay constraint for the $\beta - th$ V2V pair can be rewritten to ensure a steady-state queue length with a tolerable probability threshold [46]:

$$\Pr(D_{t\beta} \geq D_{max}) \leq \Pr(Q_{t\beta} \geq Q_{max}) \leq p_o \quad (4)$$

p_o is the probability threshold.

By applying Markov's inequality, which states that $\Pr(X \geq a) \leq \frac{E[X]}{a}$ for a non-negative random variable (X and $a > 0$), we can further strengthen the constraint on the queue length (4) in the following manner [46]:

$$\bar{Q}_k = \lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T E[Q_{t\beta}] \leq p_o \cdot Q_{max} \quad (5)$$

here \bar{Q}_k represents the time-averaged queue length of the transmitter. The upper bound on the time-averaged queue length ensures that V2V packets can be delivered within the specified time constraints.

- Reliability Requirements

In order to ensure reliable transmission of V2V pairs, the SINR outage probability is considered as a key metric. The outage probability represents the likelihood of the instantaneous SINR falling below a specified threshold γ_o , indicating a loss of signal quality due to wireless channel fading. Evaluating the reliability of V2V transmissions, the outage probability is compared against a predetermined threshold value. A V2V link γ is considered reliable if its outage probability is below a specified threshold p_o . Mathematically, this condition can be expressed as:

$$\Pr\{\gamma_{\beta,f} \leq \gamma_o\} \leq p_o \quad (6)$$

To achieve reliable transmission, vehicles have the option to adjust their power levels or switch to less congested channels if their outage probability constraint cannot be met. By incorporating the SINR of V2V/V2I pairs, the constraint for outage probability can be expressed as:

$$\Pr \left\{ P_{j,\beta,f} \times g_{\beta,B}[f] \leq \sum_{\gamma} (\rho_{\gamma,\beta} \times P_{\gamma,f} \times g_{\gamma,B}[f]) + \gamma_o ((\rho_{j,\alpha,f} \times P_{i,\alpha,f} \times g_{\alpha,B}[f])) \right\} \leq p_o \quad (7)$$

If x_1, x_2, \dots, x_n are independent exponentially distributed random variables with expected values $E[x_i] = \frac{1}{\lambda_i}$, where $i = 1, 2, \dots, n$, then the probability that x_1 is less than or equal to the sum of x_2, x_3, \dots, x_n plus a positive constant c can be expressed as [47].

$$\Pr\{x_1 \leq x_2 + x_3 + x_4 + \dots + x_n + c\} = 1 - e^{-\lambda_1 c} \prod_{i=2}^n \frac{1}{1 + \frac{\lambda_1}{\lambda_i}} \quad (8)$$

Considering that the fast fading varies independently between slots, it follows an exponential distribution. Thus, $P_{j,\beta,f} \times g_\beta[f]$, and $\gamma_0 \times (\rho_{\gamma,\beta} \times P_{\gamma,f} \times g_{\gamma,\beta}[f])$, also follow exponential distribution and can be expressed as:

$$E[P_{j,\beta,f} \times g_\beta[f]] = \frac{1}{\lambda_1} \quad (9)$$

and

$$E[\gamma_0 \times (\rho_{\gamma,\beta} \times P_{\gamma,f} \times g_{\gamma,\beta}[f])] = \frac{1}{\lambda_\gamma} \quad (10)$$

$\gamma \neq 1$

Based on equation (8), we can derive the expression for the outage probability as follows:

$$1 - \exp\left(-\frac{\gamma_0 \times \sigma^2}{P_{j,\beta,f} \times \zeta}\right) \left(1 + \frac{P_{j,\beta,f} \times \gamma_0}{P_{\gamma,\beta} \times \zeta}\right) \prod_{k=k}^K \left(1 + \frac{P_{j,\beta,f} \times \gamma_0}{P_{j,\beta,f} \times \zeta}\right) \leq p_0 \quad (11)$$

ere ζ is the frequency-independent large-scale fading effect, which encompasses path loss and shadowing, can be characterized by the path loss model defined as $128.1 + 37.6 \log_{10} d$, as specified in 3GPP TR 36.885 [48]. Here, d represents the distance between the transmitter and receiver.

Considering the inequality [48]:

$$e^k \times \prod_{i=1}^n x_i \leq e^{k+(x_1+x_2+\dots+x_n)} \quad (12)$$

upper bound of (11) can be defined and reliability constraint can be expressed as follows [46]:

$$\frac{P_{j,\beta,f} \times g_\beta[f]}{\sigma^2 + \sum_j (\rho_{j,\beta,f} \times P_{j,\beta,f} \times g_{j,\beta}[f]) + \sum_\gamma (\rho_{\gamma,\beta} \times P_{\gamma,f} \times g_{\gamma,\beta}[f])} \geq \frac{\gamma_0}{\ln\left(\frac{1}{1-p_0}\right)} \quad (13),$$

I. Problem Formulation

In order to satisfy the varying QoS demands of different vehicular links, such as high capacity for V2I connections and reliable performance for V2V connections, we aim to

maximize the total capacity of the V2I links while ensuring a reliability level of reliability for each V2V link. This leads us to formulate the spectrum and power allocation problems as follows:

The objective is to maximize the expression:

$$\max_{\rho_{i,\alpha,f}, \rho_{j,\beta,f}} \frac{1}{T} \sum_{m,f} \rho_{i,\alpha,f} \log_2(1 + \gamma_{i,\alpha,f}) \quad (14)$$

subject to the following constraints:

$$\rho_{j,\beta,f} \Pr(\gamma_{j,\beta,f} \leq \gamma_0) \leq p_0, \forall k, f \quad (14-a)$$

$$\sum_{\beta} \rho_{j,\beta,f} = 1, \forall f \quad (14-b)$$

$$\sum_f \rho_{j,\beta,f} = 1, \forall \beta \quad (14-c)$$

$$Q_t^\beta \leq Q_{max}, \forall \beta \quad (14-d)$$

$$\sum_f \rho_{j,\beta,f} = 1, \forall \beta \quad (14-e)$$

$$\sum_f \rho_{i,\alpha,f} P_{i,\alpha,f} \leq P_{i,max}, \forall \alpha \quad (14-f)$$

$$\sum_f \rho_{j,\beta,f} P_{j,\beta,f} \leq P_{f,max}, \forall \beta \quad (14-g)$$

$$P_{i,\alpha,f} \geq 0, P_{j,\beta,f} \geq 0, \forall \alpha, \beta, f \quad (14-h)$$

$$\rho_{i,\alpha,f}, \rho_{j,\beta,f} \in [0,1], \forall \alpha, \beta, f \quad (14-i)$$

This optimization problem aims to maximize the sum capacity of the V2I links while satisfying the reliability constraint for V2V links, along with various spectrum and power allocation constraints. Problem (14) represents a complex optimization problem that is challenging both mathematically and computationally. It involves making joint decisions on channel selection and power allocation over time, which requires considering various combinations and scenarios. Traditional centralized approaches struggle to handle this problem effectively, primarily due to the difficulty in acquiring accurate and up-to-date channel state information (CSI) for all links in real-time.

To address these challenges, we propose a federated-based decentralized solution leveraging the power of DRL. By applying DRL techniques, we transform the original problem into a multi-agent framework, where each V2V pair acts as an independent agent responsible for its own resource allocation strategy.

In this decentralized approach, the communication pairs autonomously make decisions on channel selection and power allocation based on local observations and rewards obtained through interactions with the environment. Through continuous learning and policy updates, each agent improves its resource allocation strategy over time, optimizing the overall system performance.

By distributing the decision-making process among the individual communication links, the decentralized DRL approach offers several advantages. It reduces the computational burden by distributing the optimization task across multiple agents. It also mitigates the need for centralized coordination and real-time CSI exchange, which can be challenging in practical scenarios.

Furthermore, the decentralized nature of the DRL approach enables scalability and adaptability to dynamic network conditions. Each agent can quickly adapt to changes in the environment and adjust its resource allocation strategy, accordingly, ensuring efficient and reliable communication.

A. DRL Formula

In order to understand the application of DRL in our context, let's introduce the fundamental concepts of DRL and its extension to a multi-agent setting. DRL involves training an intelligent agent to make optimal sequential decisions by interacting with its environment through trial and error.

At each time step t , the agent observes its surrounding environment and receives an observation $s^t \in S$. Based on this observation, the agent selects an action $a^t \in A$ according to a policy $\pi: S \rightarrow A$, which determines the probability of taking a specific action given a certain state. The environment is influenced by the executed action, leading to a transition to the next state $s^{t+1} \in S$. In response to the action, a reward $r^t = R(s^t, a^t)$ is provided to the agent, evaluating the impact of the chosen action and enabling the agent to adjust its policy accordingly. Each interaction between the agent and the environment creates an experience, represented by a tuple (s^t, a^t, r^t, s^{t+1}) .

By accumulating these experiences and employing appropriate DRL algorithms, the agent can learn to make informed decisions over time. Through a series of trials and adjustments, the agent improves its policy, maximizing the cumulative rewards obtained from the environment. In a multi-agent setting, each agent follows this learning process independently, interacting with its own observations, actions, and rewards.

Q-learning is a widely used DRL algorithm that aims to maximize the expected cumulative reward, also known as the Q-value, based on a given policy π . The Q-value denoted as $Q_\pi(s, a)$, represents the expected total reward an agent can achieve by taking action a in state s and following policy π thereafter. Mathematically, the Q-value can be defined as the expected sum of discounted future rewards, as shown in Equation (15).

$$Q_\pi(s, a) = E_\pi \left[\sum_{k=0}^{\infty} \beta^k r_{t+k+1} \mid s_t = s, a_t = a \right] \quad (15)$$

where E_π denotes the expectation under policy π , r_t is the reward obtained at time step t , and β is a discount factor that determines the importance of future rewards.

Q-learning is a popular DRL algorithm that maximizes the expected cumulative reward (Q-value) based on a given policy π . The Q-value $Q^\pi(s, a)$ represents the expected cumulative reward when taking action a in state s following policy π . The goal of Q-learning is to find the

optimal policy that maximizes the Q-value for each state-action pair.

In Q-learning, an agent maintains a Q-table to store the Q-values for all possible state-action pairs. The Q-values are updated iteratively based on the observed rewards and the agent's learning rate. At each step, the agent selects an action a based on the current state s and updates the Q-value using the following equation:

$$Q(s, a) \leftarrow (1 - \alpha) \cdot Q(s, a) + \alpha \cdot \left(r + \gamma \cdot \max_{a'} Q(s', a') \right) \quad (16)$$

where α is the learning rate, r is the immediate reward obtained by taking action a in state s , γ is the discount factor, s' is the next state, and a' is the next action. This update equation combines the current Q-value with the discounted future Q-value of the next state-action pair, scaled by the learning rate.

To handle large-scale problems with high-dimensional state and action spaces, Deep Q-Network (DQN) was introduced. Instead of using a Q-table, DQN employs a deep neural network to approximate the Q-value function. The neural network takes the state s as input and outputs the Q-values for all possible actions. The parameters of the neural network are updated through gradient descent using a loss function that minimizes the difference between the predicted Q-values and the target Q-values. In the multi-agent setting, each agent interacts with the environment and learns independently. However, their actions collectively influence the environment's dynamics. To address this, the concept of multi-agent reinforcement learning (MARL) is introduced. MARL allows agents to learn and adapt their policies by considering the joint actions and observations of other agents.

One approach in MARL is Independent Q-Learning (IQL), where each agent maintains its own Q-values and learns independently. The Q-values are updated based on the observed rewards and the Q-values of other agents' actions. The update equation for agent i can be written as:

$$Q_i(s, a_i) \leftarrow (1 - \alpha_i) \cdot Q_i(s, a_i) + \alpha_i \cdot \left(r_i + \gamma \cdot \max_{a'_i} Q_i(s', a'_i) \right) \quad (17)$$

where a_i is the action taken by agent i , r_i is the immediate reward obtained by agent i , and s' is the next state. The Q-value update is similar to the single-agent case, but it takes into account only the individual agent's actions and rewards. It is important to note that MARL introduces additional challenges such as coordination among agents and balancing exploration and exploitation. Various algorithms and techniques have been proposed to address these challenges, including centralized training with decentralized execution, communication among agents, and opponent modeling.

In our paper, we propose a decentralized DRL approach that operates within a collaborative reward setting. The aim of our research is to enable multiple

agents to learn and make optimal decisions in a distributed manner, while striving to maximize the global cumulative reward. By leveraging DRL techniques, we address the challenges associated with large-scale problems and the complex interactions between agents.

The key idea behind our approach is to train individual agents to independently learn their own policies based on local observations of the environment. Each agent selects actions based on its own policy, contributing to a joint action that impacts the overall state transition. The agents receive a common reward signal, encouraging them to coordinate their actions towards achieving a collective objective meeting the QoS requirements. The learning model consists of the following key elements:

- 1) **State:** The global state, denoted as S_t , captures the channel conditions of all V2V / V2I pairs, as well as the resource allocation actions and reusing the channel of the V2V pairs. Each V2V agent (or agent k) has access to a local state, which includes the channel coefficient of the V2V pair ($h_{t,k}[m]$), the channel coefficients of the V2I ($h_{t,m,B}[m]$), the channel selection of neighboring V2V pairs in the previous slot ($N_{m,t-1,k}$), and the current queue length at the transmitter ($Q_{t,k}$). The state space size per V2V pair is $3M + 1$, where M represents the number of available channels.
- 2) **Action:** Each V2V agent takes actions that determine the channel selection $\zeta_{k,m}$ and transmit power P_k . The transmit power is discretized into $N_p + 1$ levels, and the action space dimension is $M \times (N_p + 1)$.
- 3) **Reward:** The reward function, denoted as R_t , is designed to maximize the total capacity of the V2I links while ensuring a reliability level of reliability for each V2V link. It is defined as:

$$R_t = \Gamma_1 \sum_{k \in K} U \left(\frac{P_{j,\beta,f} \times g_{\beta}[f]}{\sigma^2 + \sum_j (\rho_{j,\beta,f} \times P_{j,\beta,f} \times g_{j,\beta}[f]) + \sum_Y (\rho_{Y,\beta} \times P_{Y,f} \times g_{Y,B}[f])} - \frac{\gamma_o}{\ln\left(\frac{1}{1-p_o}\right)} \right) + \Gamma_2 \sum_{k \in K} U(Q_{t,k} - Q_{\max}) + \Gamma_3 \sum_{m \in M} U(R_{t,m} - R_{\min,m}) \quad (18)$$

Here, $(\Gamma_1), (\Gamma_2), and (\Gamma_3)$ are parameter coefficients that balance the importance of different components in the reward function. $R_{t,m}$ represents the achieved rate of V2I communication on channel (m), while $(R_{\min,m})$ denotes the minimum required rate for V2I communication on channel (m). $(Q_{t,k})$ represents the queue length at the transmitter of V2V pair (k), and (Q_{\max}) is the maximum tolerable queue length.

The last term of the reward function represents the reward for the V2V pairs. It incorporates the SINR

$$\left(\frac{P_{j,\beta,f} \times g_{\beta}[f]}{\sigma^2 + \sum_j (\rho_{j,\beta,f} \times P_{j,\beta,f} \times g_{j,\beta}[f]) + \sum_Y (\rho_{Y,\beta} \times P_{Y,f} \times g_{Y,B}[f])} \right)$$

and compares it with the threshold $\left(\frac{\gamma_o}{\ln\left(\frac{1}{1-p_o}\right)} \right)$. The

function $U(x)$ provides a penalty if the reward condition is not satisfied, where x can be either positive or negative.

The reward function aims to maximize the achieved rates of V2I communication, minimize the queue lengths of V2V pairs, and ensure satisfactory SINR levels for V2V links. The coefficients $(\Gamma_1), (\Gamma_2), and (\Gamma_3)$ allow for balancing the trade-offs between these objectives.

In this section, we introduce the "Multi-Scale Federated DRL Framework," designed to address the challenges posed by stringent latency requirements and limited training data for accurate DRL models. Moreover, it tackles the issues of suboptimal decisions by newly activated V2V pairs and potential obsolescence of well-trained DRL models due to vehicle mobility.

The proposed framework leverages the similarities in channel quality and environmental observations among nearby V2V pairs through a multi-scale approach. It combines centralized clustering on a large timescale with federated DRL on a small scale, aiming to train robust DRL models and enhance the performance of newly activated V2V/V2I pairs. The ultimate goal is to optimize V2X communication in vehicular networks.

We propose a novel multi-scale decentralized federated DRL which synergizes federated learning and DRL techniques to address the mode selection and resource allocation challenges in vehicular networks. Fig. 3 illustrates the architecture of the multi-scale decentralized federated DRL framework, comprising two distinct procedures operating at different scales.

For a centralized training and for less periodic timescale, the base station periodically constructs undirected graphs based on large-scale channel gains and clusters nearby with similar channel conditions. Additionally, each cluster's candidate resource block group is determined to minimize network dimension and mitigate resource conflicts.

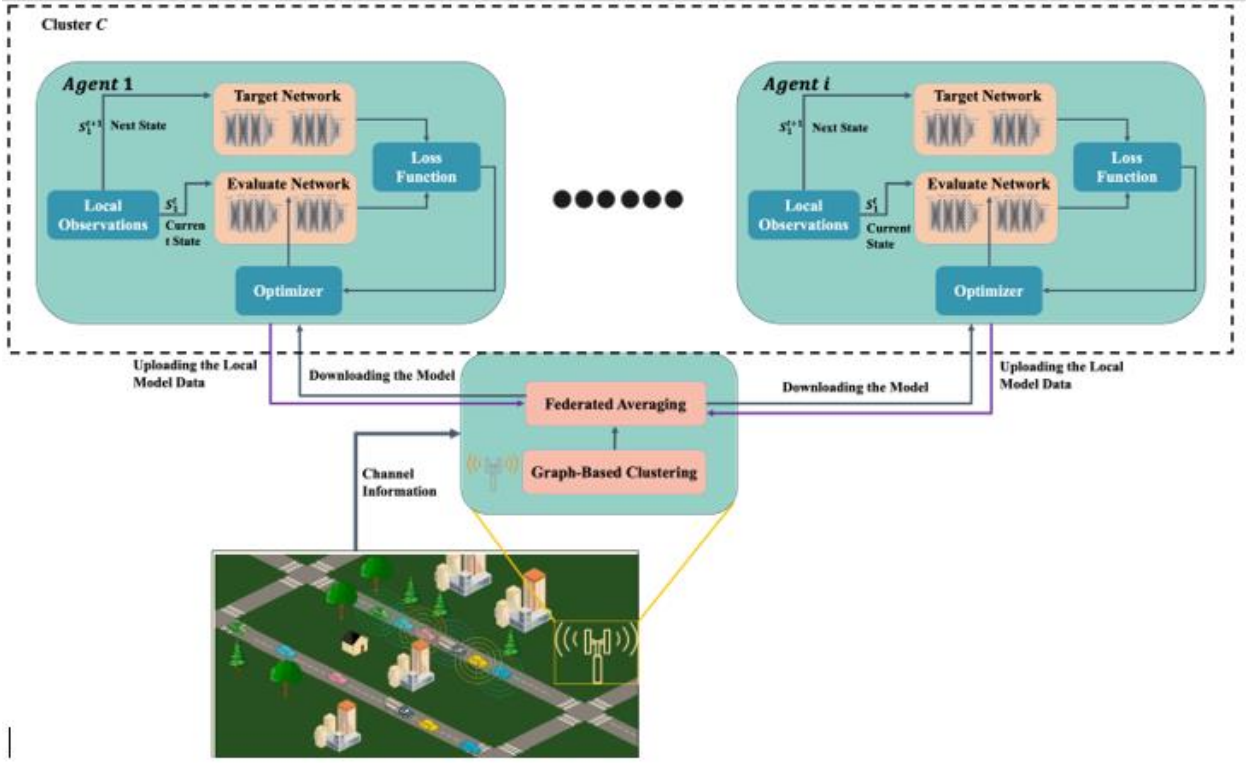


Figure 2. Proposed Federated DRL Architecture.

For local agents, federated learning is employed to collaboratively average local models of V2V and V2I pairs within each cluster. V2V and V2I pairs in the same cluster independently select actions and train their local models in each subframe. Periodically, every few hundreds of subframes, the local models of member pairs within a cluster are uploaded, averaged, and then shared as a global network with all members. Notably, the global network can be efficiently downloaded by newly activated pairs to expedite their deployment without time-consuming training processes.

The procedure for clustering is as follows. Initially, we create an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where each V2V/V2I pair represents a vertex, and edges connect pairs of vertices. $\mathcal{V}(\mathcal{G})$ and $\mathcal{E}(\mathcal{G})$ denote the sets of vertices and edges, respectively. Given the unreliable link connections between nearby connections vehicular networks due to blockage, we use large-scale channel gains as edge weights instead of Euclidean distances. The weight of the edge between vertex α and β is defined as [2]:

$$w_{\alpha,\beta} = \max(g_{\alpha,\beta}, g_{\beta,\alpha}) \quad \text{for all } \beta \neq \alpha. \quad (19)$$

To clusters with similar channel gains, we formulate the clustering problem as a graph partitioning problem aiming to maximize the sum of weights of edges inside clusters. The objective function can be expressed as:

$$\max_{C_1, \dots, C_C} \sum_{c=1}^C \sum_{i,j \in C_c} w_{i,j}$$

subject to the constraints:

$$\begin{aligned} C_1 \cup C_2 \cup \dots \cup C_C &= \mathcal{V}(\mathcal{G}) \\ C_i \cap C_j &= \emptyset \quad \text{for all } i \neq j \end{aligned}$$

where $\mathcal{V}(\mathcal{G})$ and $\mathcal{E}(\mathcal{G})$ represent the sets of vertices and edges of the undirected graph \mathcal{G} , respectively.

The above graph partitioning problem is known to be NP-hard, and traditional Euclidean distance-based clustering methods like K-means and K-medoids are not applicable due to the weights in the constructed undirected graph being based on channel gains instead of Euclidean distances. To address these challenges, we adopt the spectral clustering method, which utilizes similarity-based weights and finds an optimal solution through multiple searches [2][50]. To mitigate interference among V2V and V2I pairs in the same cluster, it is essential to allocate orthogonal resources to them. Based on the clustering results, we define the candidate RB group for cluster C_c as:

$$F_c = \mathcal{F} \setminus \{m \mid m \in \mathcal{M}, m \in C_c\}$$

This approach aims to train robust DRL models in local agents and improve the performance of newly activated V2V or V2I pairs. With the cluster sets and candidate RB groups obtained from the centralized clustering, we introduce federated learning to facilitate the training of robust DRL models. The federated DRL process consists of numerous coordination rounds. During each coordination round $r = 1, 2, \dots$, the base station BS distributes the pretrained or averaged model to V2V/V2I pairs in the same clusters. Each pair then performs DRL-Based

Decentralized Learning Algorithm to train their own local models based on local training data. After training, the BS selects pairs from the same cluster to upload their local models. Federated averaging is then performed to calculate the weights of the global Q network, which is then later redistributed to all pairs in the cluster.

In the federated DRL process, each pair independently selects its action based on local observations, without any knowledge of actions taken by other pairs. This may limit the ability to characterize the entire environment, potentially leading to resource collisions and suboptimal decisions. To address this, we introduce an asynchronous scheme where subframes are divided into multiple subframe blocks. Each pair in the same cluster is allocated to a specific subframe and asynchronously performs action selection at the designated subframe.

For newly activated pairs, they request the BS to decide the cluster set to which they belong. The global DRL model and detailed network parameters of their specific clusters are then downloaded to these newly activated pairs. By doing so, the time-consuming training process of local DRL models is avoided, and they can quickly integrate into the existing federated DRL framework.

The core process of federated DRL is achieved through minibatch-based stochastic gradient descent for federated averaging. The global model's weights are updated based on the local models' weights from pairs within the same cluster. The update process occurs with a soft update factor τ , which stabilizes the learning process and ensures that the parameters of the target network (φ_t) are slowly updated compared to the evaluate network (φ_e). We can summarize the reward function of local agents as follow:

$$R(s, a) = \lambda_1 \sum_{m \in M} U(R_{s,m} - R_{\min,m}) + \lambda_2 \sum_{\alpha \in A} U(Q_{s,\alpha} - Q_{\max}) + \lambda_3 \sum_{\alpha \in A} U \left(\frac{P_{s,\alpha} \times g_{\alpha}[f]}{\sigma^2 + \sum_{\beta} (\rho_{s,\beta} \times P_{s,\beta} \times g_{\beta}[f]) + \sum_{\gamma} (\rho_{\gamma,\alpha} \times P_{\gamma,f} \times g_{\gamma,B}[f])} - \frac{\gamma_0}{\ln(1-p_0)} \right)$$

Where:

- s represents the current state.
- a denotes the action taken in state s .
- M is the set of resource blocks RBs.
- A is the set of available actions.
- $R_{s,m}$ is the received signal-to-noise ratio SNR of $RB(m)$ in state s .
- $R_{\min,m}$ is the minimum required SNR of $RB(m)$.
- $Q_{s,\alpha}$ is the channel quality of action α in state s .
- Q_{\max} is the maximum allowable channel quality.
- $P_{s,\alpha}$ is the transmit power for action α in state s .
- $g_{\alpha}[f]$ represents the channel gain of action α at frequency f .
- σ^2 denotes the total interference and noise power.
- $\rho_{s,\beta}$ and $\rho_{\gamma,\alpha}$ are binary variables that indicate if a V2V or V2I belongs to cluster β or γ , respectively.
- $P_{\gamma,f}$ is the transmit power of V2I γ at frequency f .
- $g_{\gamma,B}[f]$ is the channel gain between V2I pairs γ and the BS at frequency f .
- γ_0 is a parameter representing a threshold value.

- p_0 is the target outage probability.

- λ_1 , λ_2 , and λ_3 are weighting factors for the different components of the reward function.

With this new formula, the reward function captures the trade-offs between the signal quality, power consumption, and resource allocation, helping the federated DRL-based algorithm make more informed decisions during the learning process. By combining the centralized clustering on a large scale with federated DRL on a small-scale and shorter timeframe, the multi-scale decentralized framework, could lead to improved resource allocation and overall network performance.

4. Simulations and Results

In this section, we evaluate the performances of the proposed multi-scale decentralized federated DRL algorithms for cellular vehicle-to-everything (V2X) communications through simulations.

For our simulation study, we chose the SUMO, a widely used open-source microscopic traffic simulator. SUMO allows us to model and simulate vehicular movements and traffic scenarios with high fidelity, making it an ideal choice for evaluating the performance of our proposed multi-scale decentralized federated DRL framework in vehicular networks. To implement our multi-scale decentralized federated DRL framework and interact with SUMO dynamically, we utilized FLOW, a framework that provides deep reinforcement learning-related APIs to work seamlessly with SUMO. FLOW simplifies the integration of reinforcement learning techniques with traffic simulations, enabling us to design and evaluate our DRL-based algorithms efficiently.

To facilitate the development and optimization of our DRL models, we relied on various libraries and tools. Scipy and NumPy provided us with essential functionalities for scientific computing and numerical operations, respectively. Asynchronous RL algorithms allowed us to efficiently train our models by leveraging parallelism and concurrency, speeding up the learning process.

We consider a crossroads scenario in our simulation, where vehicles are distributed based on the spatial Poisson process, and a base station is located at the center of various clusters. Among the vehicles, ten infrastructures simulated transmitting and receiving signals and K active V2V transmitters are randomly selected, and each V2V transmitter establishes a V2V link with the farthest vehicle in its broadcast range. We adopt the large-scale channel gains for the link for V2V pairs, considering their unreliability due to blockage. The communication parameters are based on the urban street scenario in 3GPP TR 37.885. We defined the safety-critical messages of 1060 bytes for latency and reliability requirements of 10ms and 99% with an outage threshold of 5dB. The capacity requirement of V2I defined as 5 bps/Hz. The number of predefined clusters is set as 20. The specific parameters used in the simulations are listed in Table I.

Table 1. Simulation Parameters

Parameter	Value
Carrier Freq.	5.9 GHz
# of Channels	15
Channel Bandwidth	1 MHz
# of Resource Blocks	20
# of Clusters	10
# of V2I Pairs	20
# V2V Pairs in each cluster	5 - 45
Path Loss Model	Line of Sight: $44.23 + 16.7\log(\text{distance})$ None-LOS: $42.52 + 30\log(\text{distance})$
Transmit Power	23 dBm
Noise Power	-114 dBm
Network Update Frequency	2
Federated Averaging Freq.	200
Weights in reward function	0.2, 0.8, 1, 1.2
Discount Factor	0.7
Learning Rate	0.001
Initial and Final Exploration	1, 0.01
Total # of steps	2000

For the simulations, we employ a fully connected neural network as the DRL model [2]. It consists of an input layer, a hidden layer with 256 neurons, and an output layer. ReLU ($f(x) = \max(0, x)$) is used as the activation function, and adaptive moment estimation is the optimizer. The parameters related to the DRL model are provided in Table I. We considered various parameters to evaluate the performance and three algorithms are considered for comparison in this work including a random C-V2X resource selection algorithm [52], a greedy approach where agents always select the channel with the lowest interference, and DRL based algorithm [23]. We evaluate the performance of the proposed algorithms for data rate, and the reliability and latency requirements.

First, we focus on assessing the performance of the proposed algorithm in terms of data rate. In Fig. 4, shows the data rate versus number of created V2V/V2I links and it obvious that the average data rate experiences a decline as the number of communications pairs increases. The increased number of pairs sharing the same channel intensifies interference, thereby leading to reduced transmit rates for the links. However, the proposed method outperforms other approaches in mitigating interference. By employing clustering, coordination among vehicles on different channels and selecting appropriate power levels based on local observations, the proposed method effectively alleviates interference.

On the other hand, the random selection algorithm performs poorly, and the greedy method shows almost equally unsatisfactory results. The random selection fails to consider channel quality and resource allocation, while the greedy method compels V2V pairs to utilize maximum power, further degrading data rates. The strength of the

proposed method lies in its ability to leverage clustering and federated techniques. This enables the method to optimize resource allocation and efficiently manage interference.

Figure 4 illustrates the relationship between data rate and the number of established V2V/V2I links. As the number of communication pairs increases, the average data rate exhibits a gradual decline. This phenomenon is attributed to heightened interference as more pairs share the same channel, resulting in reduced transmission rates for individual links.

However, the proposed method demonstrates superior performance in mitigating interference compared to other approaches. By employing clustering, coordinating vehicles on different channels, and strategically adjusting power levels based on local observations, the proposed method effectively alleviates interference and maintains higher data rates.

In contrast, the random selection algorithm and the greedy method exhibit suboptimal results. The random selection algorithm fails to consider channel quality and resource allocation, leading to inefficient resource utilization. The greedy method, by forcing V2V pairs to use maximum power, exacerbates interference and further degrades data rates.

The proposed method's strength lies in its ability to leverage clustering and federated techniques. These approaches enable the method to optimize resource allocation and effectively manage interference, resulting in improved overall system performance.

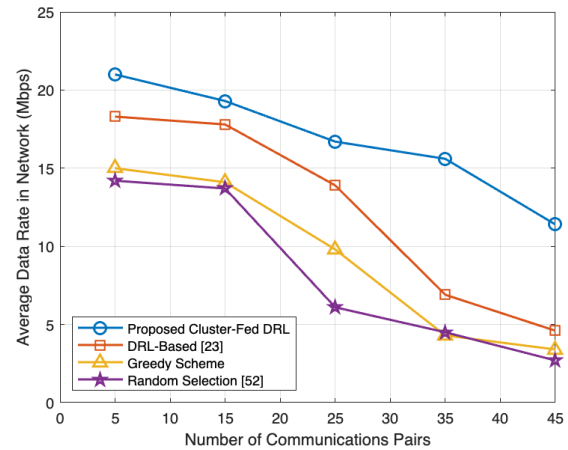


Figure 3. Average Data Rate versus Number of Communications (V2V/V2I) Pairs

We further assess the system's performance by evaluating whether the links meet the latency and quality requirements (success rate). As shown in Figure 5, the proposed algorithm exhibits high performance. The effectiveness of the proposed algorithm lies in its ability to accurately identify unstable links and make optimal transmission mode selections based on local observations and also clustering approach. As a result, the algorithm demonstrates improved performance even as the number of V2V pairs increases.

Moreover, when V2V pairs select the V2I mode, the algorithm efficiently manages transmit power to meet reliability requirements. This approach effectively reduces interference levels, especially in scenarios with a large number of communication pairs. Consequently, as the number of links grows, the performance gap between the proposed algorithm and other alternatives becomes more apparent. Overall, the simulation results indicate that the proposed algorithm is robust and capable of delivering satisfactory data rates, latency, and quality of service, making it a promising solution for enhancing vehicular communication systems' performance in real-world scenarios.

Figure 5 illustrates the system's performance in terms of meeting latency and quality requirements (success rate). The proposed algorithm consistently demonstrates superior performance in this regard.

The algorithm's effectiveness stems from its ability to accurately identify unstable links and select optimal transmission modes based on local observations and clustering. This approach enables the algorithm to adapt to changing network conditions and mitigate interference effectively, even as the number of V2V pairs grows.

When V2V pairs choose the V2I mode, the algorithm efficiently manages transmit power to ensure reliable communication. This power control strategy helps to reduce interference levels, particularly in scenarios with a high density of communication pairs.

As the number of links increases, the performance gap between the proposed algorithm and other alternatives becomes more pronounced. The simulation results clearly demonstrate the robustness and efficacy of the proposed algorithm in delivering satisfactory data rates, latency, and quality of service. This makes it a promising solution for enhancing the performance of vehicular communication systems in real-world scenarios.

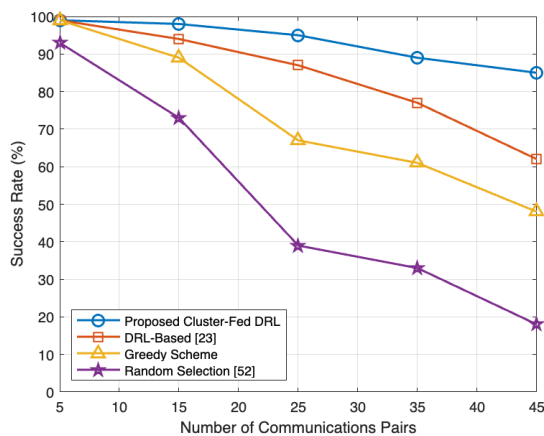


Figure 4. Success Rate versus Number of Communications Pairs

We also conducted evaluations on both data rate and success rate in relation to the SINR threshold. Figures 6 and Fig. 7 illustrates these metrics for varying thresholds, with the number of communications pairs fixed at 25. The proposed algorithm exhibits adaptability to larger thresholds by selecting optimal transmission modes, best reusable channels and adopting appropriate transmission

powers, thereby effectively mitigating interference. This adaptability leads to reduced interference and higher reliability, contributing to its overall superior performance. Notably, as the thresholds increase, the average data rate declines. This decrease is attributed to the fact that larger thresholds necessitate communicating pairs to select higher transmission power levels to meet reliability requirements. Consequently, this leads to stronger interference for pairs sharing the same channel. In contrast, the greedy method and random selection schemes always opt for maximum transmission power without considering the transmit rate. As a result, the average rate remains unchanged across different thresholds. This lack of adaptability limits the performance of these two methods.

Figure 6 and Figure 7 depict the relationship between data rate and success rate with varying SINR thresholds, while maintaining a fixed number of communication pairs (25).

The proposed algorithm demonstrates remarkable adaptability to larger SINR thresholds. By strategically selecting optimal transmission modes, identifying the best reusable channels, and adjusting transmission powers, the algorithm effectively mitigates interference. This adaptability results in reduced interference and higher reliability, contributing to its overall superior performance.

However, as the SINR threshold increases, the average data rate gradually declines. This is because higher thresholds necessitate communicating pairs to employ higher transmission power levels to meet reliability requirements. Consequently, this leads to increased interference among pairs sharing the same channel.

In contrast, the greedy method and random selection schemes consistently operate at maximum transmission power, regardless of the SINR threshold. This lack of adaptability limits their performance, as they fail to optimize power usage and mitigate interference effectively. As a result, the average data rate remains relatively unchanged across different thresholds.

Overall, the proposed algorithm's ability to adapt to varying SINR thresholds and optimize transmission parameters is a key factor in its superior performance. This adaptability enables it to achieve higher data rates and reliability, even in challenging communication environments.

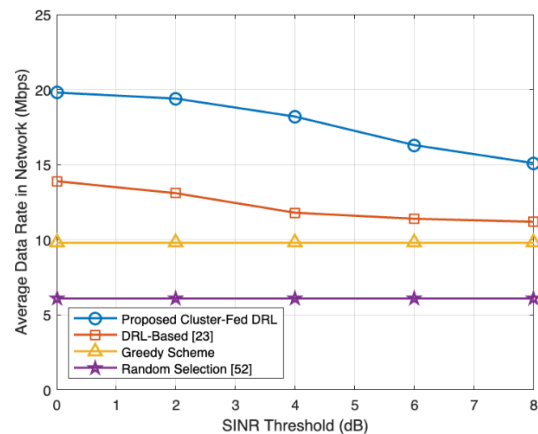


Figure 5. Average Data Rate versus SINR Threshold

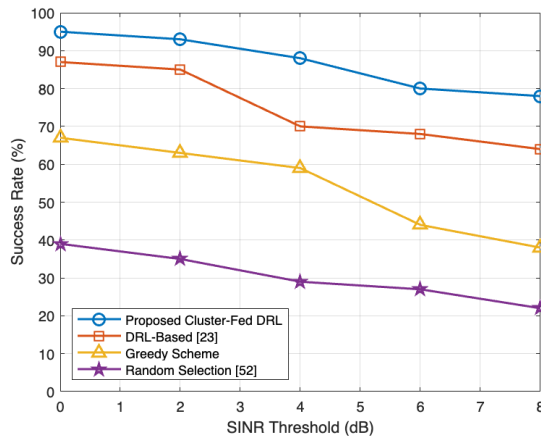


Figure 6. Success Rate versus SINR Threshold

5. Conclusion

In this research, we aimed to empower vehicles with the autonomy to make intelligent channel selection decisions for their transmissions. To achieve this goal, we proposed a novel two-layer multi-agent approach, wherein individual vehicles acted as agents, making autonomous decisions based on their local observations, while clusters of nearby vehicles collaborated to improve overall system performance.

Our decision-making model was built on the foundation of deep reinforcement learning (DRL) techniques, which considered multiple factors such as instantaneous channel state information, queue backlog at the transmitter, interference from different links, and historical selections of nearby links. This allowed the agents to make well-informed choices concerning channel selection and power allocation, optimizing communication efficiency.

To further enhance the performance of our decentralized vehicular agents, we integrated federated learning (FL) into our approach. This facilitated knowledge sharing and synchronization among the individual agents, harnessing the collective intelligence of the network. By consolidating and synchronizing the local models through FL, each agent gained insights into the broader network dynamics beyond its limited observations, leading to more accurate and coordinated decision-making.

The integration of DRL and FL offered several advantages. Firstly, it enabled real-time adaptability for each agent in response to the dynamic nature of the vehicular environment. Secondly, the collective intelligence of the network, harnessed through FL, improved decision-making efficiency, leading to better channel utilization and reduced interference.

The results from our simulations demonstrated the effectiveness of our proposed approach, showcasing its ability to tackle the challenges of vehicular communication systems.

6. Compliance with Ethical Standards

- Authors declare that they have no conflict of interest.

- This article does not contain any studies with human participants or animals performed by any of the authors.

7. Reference:

- [1] E. Obiodu, A. Raman, A. K. Abubakar, S. Mangiante, N. Sastry and A. H. Aghvami. (2022, Feb.). DSM-MoC as Baseline: Reliability Assurance via Redundant Cellular Connectivity in Connected Cars. *IEEE Transactions on Network and Service Management*. [Online]. 19(3), pp. 2178-2194. Available: 10.1109/TNSM.2022.3153452
- [2] X. Zhang, M. Peng, S. Yan and Y. Sun. (2019, Dec.). Deep-Reinforcement-Learning-Based Mode Selection and Resource Allocation for Cellular V2X Communications. *IEEE Internet of Things Journal*. [Online]. 7(7), pp. 6380-6391. Available: 10.1109/JIOT.2019.2962715
- [3] M. Yang, Y. Ju, L. Liu, Q. Pei, K. Yu and J. J. P. C. Rodrigues, "Secure mmWave C-V2X Communications Using Cooperative Jamming," in *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, Rio de Janeiro, Brazil, 2022, pp. 2686-2691
- [4] T. -X. Zheng *et al.* and K. K. W. (2022, Jun.). Physical-Layer Security of Uplink mmWave Transmissions in Cellular V2X Networks. *IEEE Transactions on Wireless Communications*. [Online]. 21(11), pp. 9818-9833. Available: <https://doi.org/10.1109/TWC.2022.3179706>
- [5] M. S. Bahbahani, E. Alsusa and A. Hammadi. (2022, Nov.). A Directional TDMA Protocol for High Throughput URLLC in mmWave Vehicular Networks. *IEEE Transactions on Vehicular Technology*. [Online]. 72(3), pp. 3584-3599. Available: 10.1109/TVT.2022.3219771
- [6] P. Xiang, H. Shan, Z. Su, Z. Zhang, C. Chen and E. -P. Li. (2022, Oct.). Multi-Agent Reinforcement Learning-Based Decentralized Spectrum Access in Vehicular Networks with Emergent Communication. *IEEE Communications Letters*. [Online]. 27(1), pp. 195-199. Available: <https://doi.org/10.1109/LCOMM.2022.3214792>
- [7] K. Sehla, T. M. T. Nguyen, G. Pujolle and P. B. Velloso. (2022, Mar.). Resource Allocation Modes in C-V2X: From LTE-V2X to 5G-V2X. *IEEE Internet of Things Journal*. [Online]. 9(11), pp. 8291-8314. Available: <https://doi.org/10.1109/JIOT.2022.3159591>
- [8] H. Bagheri *et al.* and K. Moessner. (2021, Mar.). 5G NR-V2X: Toward Connected and Cooperative Autonomous Driving. *IEEE Communications Standards Magazine*. [Online]. 5(1), pp. 48-54. Available: <https://doi.org/10.1109/MCOMSTD.001.2000069>
- [9] G. Twardokus and H. Rahbari. (2023, Mar.). Towards Protecting 5G Sidelink Scheduling in C-V2X Against Intelligent DoS Attacks. *IEEE Transactions on Wireless Communications*. [Online]. 22(11), pp. 7273-7286. Available: <https://doi.org/10.1109/TWC.2023.3249665>
- [10] K. Sehla, T. M. T. Nguyen, G. Pujolle and P. B. Velloso. (2022, Mar.). Resource Allocation Modes in C-V2X: From LTE-V2X to 5G-V2X. *IEEE Internet of Things Journal*. [Online]. 9(11), pp. 8291-8314.

- Available:
<https://doi.org/10.1109/JIOT.2022.3159591>
- [11] S. -H. Wu, R. -H. Hwang, C. -Y. Wang and C. -H. Chou. "Deep Reinforcement Learning Based Resource Allocation for 5G V2V Groupcast Communications," in 2023 International Conference on Computing, Networking and Communications (ICNC), Honolulu, HI, USA, 2023, pp. 1-6.
 - [12] X. Lin, J. G. Andrews, A. Ghosh and R. Ratasuk. (2014, Apr.). An overview of 3GPP device-to-device proximity services. *IEEE Communications Magazine*. [Online]. 52(4), pp. 40-48. Available: <https://doi.org/10.1109/MCOM.2014.6807945>
 - [13] S. Chen et al. and R. Zhao. (2017, Jul.). Vehicle-to-Everything (v2x) Services Supported by LTE-Based Systems and 5G. *IEEE Communications Standards Magazine*. [Online]. 1(2), pp. 70-76. Available: <https://doi.org/10.1109/MCOMSTD.2017.1700015>
 - [14] T. Ran, "Study on Evaluation Methodology of New V2X Use Cases for LTE and NR," Dubrovnik, Croatia, 2017.
 - [15] M. H. C. Garcia et al. and T. Şahin. (2021, Feb.). A Tutorial on 5G NR V2X Communications. *IEEE Communications Surveys & Tutorials*. [Online]. 23(3), pp. 1972-2026. Available: <https://doi.org/10.1109/COMST.2021.3057017>
 - [16] T. Shahgholi, A. Sheikahmadi, K. Khamforoosh, and S. Azizi. (2021, Feb.). LPWAN-based hybrid backhaul communication for intelligent transportation systems. Architecture and performance evaluation. *EURASIP Journal on Wireless Communications and Networking*. [Online]. (35). Available: <https://doi.org/10.1186/s13638-021-01918-2>
 - [17] H. Yang, L. Zhao, L. Lei and K. Zheng, "A two-stage allocation scheme for delay-sensitive services in dense vehicular networks," *IEEE International Conference on Communications Workshops (ICC Workshops)*, Paris, France, 2017, pp. 1358-1363. Available: <https://doi.org/10.1109/ICCW.2017.7962848>
 - [18] L. Liang, S. Xie, G. Y. Li, Z. Ding and X. Yu. (2018, Feb.). Graph-Based Resource Sharing in Vehicular Communication. *IEEE Transactions on Wireless Communications*. [Online]. 17(7), pp. 4579-4592. Available: <https://doi.org/10.1109/TWC.2018.2827958>
 - [19] C. Chen, B. Wang and R. Zhang. (2018, Oct.). Interference Hypergraph-Based Resource Allocation (IHG-RA) for NOMA-Integrated V2X Networks. *IEEE Internet of Things Journal*. [Online]. 6(1), pp. 161-170. Available: <https://doi.org/10.1109/JIOT.2018.2875670>
 - [20] B. Bai, W. Chen, K. B. Letaief and Z. Cao. (2010, Dec.). Low Complexity Outage Optimal Distributed Channel Allocation for Vehicle-to-Vehicle Communications. *IEEE Journal on Selected Areas in Communications*. [Online]. 29(1), pp. 161-172. Available: <https://doi.org/10.1109/JSAC.2011.110116>
 - [21] L. Liang, J. Kim, S. C. Jha, K. Sivanesan and G. Y. Li. (2017, May.). Spectrum and Power Allocation for Vehicular Communications With Delayed CSI Feedback. *IEEE Wireless Communications Letters*. [Online]. 6(4), pp. 458-461. Available: <https://doi.org/10.1109/LWC.2017.2702747>
 - [22] H. Ye, G. Y. Li and B. -H. F. Juang. (2019, Feb.). Deep Reinforcement Learning Based Resource Allocation for V2V Communications. *IEEE Transactions on Vehicular Technology*. [Online]. 68(4), pp. 3163-3173. Available: <https://doi.org/10.1109/TVT.2019.2897134>
 - [23] Y. Wang, X. Zheng and X. Hou, "A Novel Semi-Distributed Transmission Paradigm for NR V2X," in *IEEE Globecom Workshops (GC Wkshps)*, Waikoloa, HI, USA, 2019, pp. 1-6.
 - [24] H. Mosavat-Jahromi, Y. Li, L. Cai and L. Lu, "NC-MAC: Network Coding-based Distributed MAC Protocol for Reliable Beacon Broadcasting in V2X," *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, Taipei, Taiwan, 2020, pp. 1-6.
 - [25] A. Zappone, M. Di Renzo and M. Debbah. (2019, Jun.). Wireless Networks Design in the Era of Deep Learning: Model-Based, AI-Based, or Both?. *IEEE Transactions on Communications*. [Online]. 67(10), pp. 7331-7376. Available: <https://doi.org/10.1109/TCOMM.2019.2924010>
 - [26] L. Wang, H. Ye, L. Liang and G. Y. Li. (2020, Mar.). Learn to Compress CSI and Allocate Resources in Vehicular Networks. *IEEE Transactions on Communications*. [Online]. 68(6), pp. 3640-3653. Available: <https://doi.org/10.1109/TCOMM.2020.2979124>
 - [27] Lu, Yan, Ping Wang, Shuai Wang, and Wangding Yao, "A Q-learning based SPS resource scheduling algorithm for reliable C-V2X communication," in *5th International Conference on Digital Signal Processing*, 2021, pp. 201-206.
 - [28] L. Busoniu, R. Babuska and B. De Schutter. (2008, Feb.). A Comprehensive Survey of Multiagent Reinforcement Learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*. [Online]. 38(2), pp. 156-172. Available: <https://doi.org/10.1109/TSMCC.2007.913919>
 - [29] B. Gu, W. Chen, M. Alazab, X. Tan and M. Guizani. (2022, Jul.). Multiagent Reinforcement Learning-Based Semi-Persistent Scheduling Scheme in C-V2X Mode 4. *IEEE Transactions on Vehicular Technology*. [Online]. 71(11), pp. 12044-12056. Available: <https://doi.org/10.1109/TVT.2022.3189019>
 - [30] L. Cao and H. Yin, "Resource Allocation for Vehicle Platooning in 5G NR-V2X via Deep Reinforcement Learning," in 2021 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom), Bucharest, Romania, 2021, pp. 1-7.
 - [31] Y. Yuan, G. Zheng, K. -K. Wong and K. B. Letaief. (2021, Jul.). Meta-Reinforcement Learning Based Resource Allocation for Dynamic V2X Communications. *IEEE Transactions on Vehicular Technology*. [Online]. 70(9), pp. 8964-8977. Available: <https://doi.org/10.1109/TVT.2021.3098854>
 - [32] Z. He, L. Wang, H. Ye, G. Y. Li and B. -H. F. Juang, "Resource Allocation based on Graph Neural Networks in Vehicular Communications," *GLOBECOM 2020 - 2020 IEEE Global*

- Communications Conference, Taipei, Taiwan, 2020, pp. 1-5.
- [33] R. Wang, X. Jiang, Y. Zhou, Z. Li, D. Wu, T. Tang, A. Fedotov and V. Badenko. (2022, Jun.). Multi-agent reinforcement learning for edge information sharing in vehicular networks. *Digital Communications and Networks*. [Online]. 8(3), pp. 267-277. Available: <https://doi.org/10.1016/j.dcan.2021.08.006>
- [34] C. Wu, Z. Liu, F. Liu, T. Yoshinaga, Y. Ji and J. Li. (2020, Jun.). Collaborative Learning of Communication Routes in Edge-Enabled Multi-Access Vehicular Environment. *IEEE Transactions on Cognitive Communications and Networking*. [Online]. 6(4), pp. 1155-1165. Available: <https://doi.org/10.1109/TCCN.2020.3002253>
- [35] B. Gu, X. Yang, Z. Lin, W. Hu, M. Alazab and R. Kharel. (2020, Sep.). Multiagent Actor-Critic Network-Based Incentive Mechanism for Mobile Crowdsensing in Industrial Systems. *IEEE Transactions on Industrial Informatics*. [Online]. 17(9), pp. 6182-6191. Available: <https://doi.org/10.1109/TII.2020.3024611>
- [36] L. Liang, H. Ye and G. Y. Li. (2019, Aug.). Spectrum Sharing in Vehicular Networks Based on Multi-Agent Reinforcement Learning. *IEEE Journal on Selected Areas in Communications*. [Online]. 37(10), pp. 2282-2292. Available: <https://doi.org/10.1109/JSAC.2019.2933962>
- [37] J. Tian, Q. Liu, H. Zhang and D. Wu. (2021, Jun.). Multiagent Deep-Reinforcement-Learning-Based Resource Allocation for Heterogeneous QoS Guarantees for Vehicular Networks. *IEEE Internet of Things Journal*. [Online]. 9(3), pp. 1683-1695. Available: <https://doi.org/10.1109/JIOT.2021.3089823>
- [38] X. Chen et al. and Y. Zhang. (2020, Jan.). Age of Information Aware Radio Resource Management in Vehicular Networks: A Proactive Deep Reinforcement Learning Perspective. *IEEE Transactions on Wireless Communications*. [Online]. 19(4), pp. 2268-2281. Available: <https://doi.org/10.1109/TWC.2019.2963667>
- [39] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics, 2017*, pp. 1273-1282.
- [40] S. Niknam, H. S. Dhillon and J. H. Reed. (2020, Jun.). Federated Learning for Wireless Communications: Motivation, Opportunities, and Challenges. *IEEE Communications Magazine*. [Online]. 58(6), pp. 46-51. Available: <https://doi.org/10.1109/MCOM.001.1900461>
- [41] J. Qi, Q. Zhou, L. Lei and Kan Zheng, "Federated reinforcement learning: Techniques, applications, and open challenges," 2021.
- [42] R. Valente, C. Senna, P. Rito and S. Sargento. (2023, Feb.). Embedded Federated Learning for VANET Environments. *Applied Sciences*. [Online]. 13(4), p. 2329. Available: <https://doi.org/10.3390/app13042329>
- [43] G. Wang, X. Fangmin, H. Zhang and C. Zhao. (2022, Apr.). Joint resource management for mobility supported federated learning in Internet of Vehicles. *Future Generation Computer Systems*. [Online]. 129, pp. 199-211. Available: <https://doi.org/10.1016/j.future.2021.11.020>
- [44] Q. Guo, F. Tang and N. Kato. (2022, Sep.). Federated Reinforcement Learning-Based Resource Allocation in D2D-Enabled 6G. *IEEE Network*. [Online]. 37(5), pp. 89-95. Available: <https://doi.org/10.1109/MNET.122.2200102>
- [45] S. Kandukuri and S. Boyd. (2002, Jan.). Optimal power control in interference-limited fading wireless channels with outage-probability specifications. *IEEE Transactions on Wireless Communications*. [Online]. 1(1), pp. 46-55. Available: <https://doi.org/10.1109/7693.975444>
- [46] X. Li, L. Lu, W. Ni, A. Jamalipour, D. Zhang and H. Du. (2022, May.). Federated Multi-Agent Deep Reinforcement Learning for Resource Allocation of Vehicle-to-Vehicle Communications. *IEEE Transactions on Vehicular Technology*. [Online]. 71(8), pp. 8810-8824. Available: <https://doi.org/10.1109/TVT.2022.3173057>
- [47] J. M. Meredith, "Technical Specification Group Radio Access Network; Study on LTEBased V2X Services; (Release 14)," 3rd Generation Partnership Project, 2016.
- [48] C. Guo, L. Liang and G. Y. Li. (2019, Feb.). Resource Allocation for Low-Latency Vehicular Communications: An Effective Capacity Perspective. *IEEE Journal on Selected Areas in Communications*. [Online]. 37(4), pp. 905-917. Available: <https://doi.org/10.1109/JSAC.2019.2898743>
- [49] J. Zeng et al and Y. Wu, "Multi-D3QN: A multi-strategy deep reinforcement learning for service composition in cloud manufacturing," *International Conference on Collaborative Computing: Networking, Applications and Worksharing*. Cham: Springer International Publishing, 2021, pp. 225-240.
- [50] H. Yang, X. Xie and M. Kadoch. (2019, Jan.). Intelligent Resource Management Based on Reinforcement Learning for Ultra-Reliable and Low-Latency IoV Communication Networks. *IEEE Transactions on Vehicular Technology*. [Online]. 68(5), pp. 4157-4169. Available: <https://doi.org/10.1109/TVT.2018.2890686>
- [51] S. Chen, J. Hu, Y. Shi and L. Zhao. (2016, Sep.). LTE-V: A TD-LTE-Based V2X Solution for Future Vehicular Network. *IEEE Internet of Things Journal*. [Online]. 3(6), pp. 997-1005. Available: <https://doi.org/10.1109/JIOT.2016.2611605>
- [52] L. Liang, S. Xie, G. Y. Li, Z. Ding and X. Yu. (2018, Apr.). Graph-Based Resource Sharing in Vehicular Communication. *IEEE Transactions on Wireless Communications*. [Online]. 17(7), pp. 4579-4592. Available: <https://doi.org/10.1109/TWC.2018.2827958>



An Enhanced Sine Cosine Algorithm for Feature Selection in Network Intrusion Detection*

Research Article

Zahra Asghari Varzaneh¹, Soodeh Hosseini² 

DOI: [10.22067/cke.2024.83460.1097](https://doi.org/10.22067/cke.2024.83460.1097)

Abstract For computer networks to remain secure, intrusion detection is essential. Analyzing network traffic data is part of this activity to spot possible cyber threats. However, the curse of dimensionality presents a challenge because there are so many dimensions in the data. To overcome this challenge, feature selection is essential to creating a successful intrusion detection system. It involves removing irrelevant and redundant features, which enhances the classification model's accuracy and lowers the dimensionality of the feature space. Metaheuristic algorithms are optimization techniques inspired by nature and are well-suited to choose features for network intrusion detection. They are effective in exploring large search spaces and have been widely used for this purpose. In this study, we improve the Sine Cosine Algorithm named ISCA for feature selection by introducing a controlling parameter to balance exploration and exploitation. Based on the NSL-KDD dataset, the results show that compared to other competing algorithms, the ISCA performs better than other metaheuristic algorithms in terms of both the number of features selected and the accuracy of classification.

Keywords: Sine Cosine Algorithm, Metaheuristic algorithms, Feature selection, Network intrusion detection.

1. Introduction

With the development of computer technology and the increase in the use of the Internet, the issue of information security has become more important [1]. Any illegal action that is done to disrupt security goals such as integrity, confidentiality, and accessibility is called intrusion [2]. To create complete security in any computer system, in addition to intrusion prevention tools such as firewalls, authentication, and encryption, other systems called Intrusion Detection Systems (IDS) are needed [3]. If the intruder passes the aforementioned security mechanisms, an IDS detects it and finds a solution to deal with it. Intrusion detection is an essential component of network

security that seeks to recognize and address attempts at unwanted entry and malicious activities within a network [4, 5]. IDS monitor the flow of network activity and distinguishes normal activities from suspicious activities and attacks. The two fundamental IDSs are host-based and network-based which all IDSs fall [6, 7]. The system's location distinguishes between these two types. The host-based is placed on the client's computer, while the network-based is disseminated throughout the network. In fact, an IDS works like a pattern recognition system, and by receiving a series of features, it detects the intrusion. Network traffic datasets may contain millions of samples containing a lot of features [8].

With the increasing complexity and sophistication of cyber threats, IDSs are becoming more critical for protecting network infrastructures [8, 9]. However, one of the main problems facing IDS is the high dimensionality of the data generated from network traffic, which could result in a decline in performance, increased false alarms, and high computational costs. Feature selection is one important phase in the process of building an effective IDS, as it aims to minimize the dimensionality of the data by choosing the most relevant features while retaining the essential information [10, 11]. The literature has presented a number of feature selection models, such as filter, wrapper, and embedding methods. However, due to the high dimensionality of IDS data, traditional feature selection methods may not be adequate, and alternative approaches are required [12].

Metaheuristic algorithms are a class of optimization techniques that have been utilized extensively in various fields, including feature selection [13-16]. The algorithms are modeled around natural phenomena, such as evolutionary processes and swarm intelligence, and is proficient at finding the best answers in intricate, high-dimensional search areas [17-19]. As such, metaheuristic algorithms have become increasingly popular in feature selection for IDS, as they can efficiently find the most

* Manuscript received: 2023 July 16, Revised, 2024 July 9, Accepted, 2024 September 24.

¹ Ph.D Student, Department of Computer Science, Faculty of Mathematics and Computer, Shahid Bahonar University of Kerman, Kerman, Iran.

² Corresponding Author. Associate Professor, Department of Computer Science, Faculty of Mathematics and Computer, Shahid Bahonar University of Kerman, Kerman, Iran. **Email:** so_hosseini@uk.ac.ir

pertinent features with the least amount of computational cost. In 2016, Mirjalili proposed a new search strategy that makes use of the sine and cosine trigonometric functions [20]. This search strategy called the SCA, is a stochastic algorithm based on population that follows basic optimization principles of exploration and exploitation.

The SCA algorithm is simple yet effective, and it has been effectively used to solve a range of optimization issues. However, Similar to other metaheuristic algorithms, SCA has limitations such as falling into local optima and premature convergence [21]. To deal with these problems, we propose an enhanced version of the SCA algorithm called ISCA for feature selection in intrusion detection. The ISCA algorithm introduces a controlling parameter that balances exploration and exploitation. By tuning the controlling parameter, we aim to enhance the algorithm's efficiency by avoiding premature convergence and improving the standard of the chosen features.

In this paper, we assess how well the suggested algorithm performs on the NSL-KDD dataset and compare it with the standard SCA algorithm. We prove that the ISCA algorithm outperforms the benchmark competing algorithms regarding the number of features selected and the classification accuracy. The outcomes of the experiments show how well our suggested method works as a feature selection tool for network intrusion detection.

The rest of this article is organized as follows: A summary of relevant works is provided in Section 2. Section 3 outlines the typical SCA algorithm and its basic principles. Details of our ISCA algorithm are provided in Section 4 and introduces a controlling parameter to balance exploration and exploitation. In Section 5, we describe the simulation setup and present the results of applying the proposed algorithm to intrusion detection datasets for feature selection. Finally, in Section 6, we provide an overview of our findings and consider potential avenues for further improving the proposed algorithm.

2. Related work

This section discusses the challenge of detecting network intrusions due to high-dimensional data and the significance of feature selection in building effective IDSs. Metaheuristic algorithms are well-suited for feature selection in network intrusion detection as they can effectively explore large search spaces. This section will review the literature on the use of metaheuristic algorithms for feature selection and their efficiency in raising intrusion detection systems' accuracy.

Alazzam *et al.* [22] They suggested an IDS wrapper feature selection approach that makes use of an optimizer inspired by pigeons. A new method for binarizing the continuous optimizer was introduced and compared to other methods used for binarizing intelligent swarm algorithms that are continuous. The proposed algorithm outperformed several state-of-the-art feature selection algorithms in terms of TPR, FPR, accuracy, and F-score on three popular datasets. The algorithm's proposed cosine similarity method for binarization converged more quickly than the sigmoid method. Beulah *et al.* [23] introduced a hybrid method that was used for feature reduction in any

domain and integrates the best features from several feature selection techniques. Their method was applied to IDSs using the NSL-KDD data, and six predominant features were chosen. Performance investigation showed that their algorithm improves the detection rate and the accuracy of the IDS.

In [24], the ideal feature subset was found by the authors using the CFS-DE technique, which was followed by the weighted stacking approach. that improves classification performance by increasing the base classifiers' weights that produce strong training results and dropping those with bad results. Experiments were carried out on NSL-KDD and CSE-CIC-IDS2018 datasets, and the model had high accuracy, recall, precision, and F1-score on both datasets. When compared to other articles' models and conventional machine learning models, the suggested CFS-DE-weighted-Stacking IDS had the top-performing classification. Vijayanand and Devaraj [25] presented a wrapper-based model for intrusion detection using the modified WOA. They integrated WOA with genetic algorithm operators and picked useful features from the network data to accurately identify incursions. They used a support vector machine (SVM) to recognize intrusions based on the selected features.

In [26] authors proposed an opposition self-adaptive grasshopper optimization algorithm (GOA) based on perceptive ideas and mutation, to detect new malicious or anomalous attacks. Furthermore, an SVM employing gain actor-critic with SVM uses reinforcement learning to improve the capacity for detecting fresh cyberattacks. The suggested algorithm beats other evolutionary techniques and the fundamental GOA regarding accuracy, detection rate and false-positive rate for resolving intrusion detection system issues, according to comparative simulation results. Salo *et al.* [27] introduced a hybrid model for dimensionality reduction in IDS that integrates an ensemble classifier based on SVM, instance-based learning methods (IBK), and multilayer perceptron (MLP) with information gain (IG) and principal component analysis (PCA). Comparative analysis showed that the Compared to most current state-of-the-art methodologies, their method performs better regarding accuracy, false alarm rate and detection rate.

Ahmed *et al.* [28] introduced a modified model for enhancing the efficiency of IDS by proposing an alternative feature selection optimizer algorithm, called GTO. The method involved using a feature extraction model, such as a convolutional neural network (CNN), as a first step, to decrease the dataset's dimensionality. The FS model was then given the retrieved features to use in detection. The results of the devised approach were in contrast to prominent IDS methodologies, and their proposed algorithm based on performance criteria, showed superiority over all other techniques. Safaldin *et al.* [29] proposed an enhanced IDS called GWOSVM-IDS, which used the enhanced binary GWO with SVM. The algorithm was tested using 3, 5, and 7 wolves to determine the ideal number of wolves for the problem. The suggested approach sought to raise the detection rate and accuracy of intrusion detection while reducing processing time in a wireless sensor network environment by decreasing false alarm rates and the number of features resulting from the

IDSs. The findings indicate that the suggested GWOSVM-IDS with seven wolves outperforms the other algorithms.

To improve network intrusion detection's efficacy, Jayalatchumy et al. [30] deployed a special intrusion detection system (IDS). Data-denoising techniques were initially used to deal with the imbalance in the data. The most important characteristics were then found using the improved Crow search method to improve intrusion attack classification. Using the chosen characteristics, an ensemble classifier classified the normal and invader labels in the last stage. The experimental findings show that the developed method provides a remarkable 99.2% accuracy for the NSL-KDD. In [31], a PCA was used by the suggested system to minimize the dimensionality of the dataset. Strong global search capacity was provided by the efficient feature selection process using an Improved HHO (IHHO). SVM is used in the first stage of a two-stage classifier, while KNN is used in the second. Combining the benefits of SVM and KNN is the main objective to minimize false alarm rate and enhance accuracy.

The key challenge in this study is selecting the most effective features in network intrusion detection using metaheuristic algorithms. In the studies conducted, most metaheuristic algorithms, despite their high computational complexity, still struggle with imbalances in the exploration and exploitation processes. The proposed ISCA addresses this by providing a simple yet computationally efficient method that achieves high accuracy, selects a smaller subset of features, and has faster execution times compared to other feature selection techniques.

3. Sine-cosine algorithm (SCA)

The sine-cosine algorithm, introduced by Mirjalili [20] is a swarm intelligence algorithm that utilizes the trigonometric functions of sine and cosine. This enables individuals to move freely to another region within the range of $[0, 2\pi]$, thereby facilitating an exploration of a larger search space or exploitation of neighborhood space. Through multiple iterations of the population, the algorithm gradually obtains the global optimal position. During each iteration, the SCA updates the position using Eq. 1, which is the update formula of the SCA algorithm as shown below:

$$X_i^{t+1} = \begin{cases} X_i^t + r_1 \times \sin(r_2) \times |r_3 p_i^t - X_i^t|, & r_4 \leq 0.5 \\ X_i^t + r_1 \times \cos(r_2) \times |r_3 p_i^t - X_i^t|, & r_4 > 0.5 \end{cases} \quad (1)$$

where the number of iterations at this time is t , X_i^t is the particle i 's location at iteration t , and the position of the global optimal particle at iteration t is determined by p_i^t and three random parameters that are uniformly distributed: $r_2 \in (0, 2\pi)$, $r_3 \in [0, 2]$, and $r_4 \in (0, 1)$.

In addition, the algorithm uses a linearly decreasing convergence parameter, r_1 , which is expressed as shown in Eq. 2. This parameter regulates how the algorithm is explored and exploited.

$$r_1 = a - t \frac{a}{T} \quad (2)$$

Where, $a = 2$, the highest number of iterations is T , and the number of iterations that are currently in use is t .

4. Proposed Algorithm

In this paper, we propose a modification to the SCA algorithm called ISCA that enhances the balance between exploitation and exploration. The modification introduces a time-varying parameter, *Alpha*, that regulates the exploration rate of the algorithm in the beginning and the exploitation rate towards the end. The updated equation can be modified as follows:

$$X_i^{t+1} = \begin{cases} X_i^t + r_1 \times \sin(r_2) \times |r_3 p_i^t - X_i^t| \times Alpha, & r_4 \leq 0.5 \\ X_i^t + r_1 \times \cos(r_2) \times |r_3 p_i^t - X_i^t| \times Alpha, & r_4 > 0.5 \end{cases} \quad (3)$$

where *Alpha* is a parameter that changes throughout time has a high initial value and progressively falls over time, favoring exploration in the beginning and exploitation towards the end. Here is how to calculate *Alpha*'s value:

$$Alpha = alpha_{max} - (alpha_{max} - alpha_{min}) \times \left(\frac{t}{T}\right)^k \quad (4)$$

where *alpha_max* and *alpha_min* are the highest and lowest numbers of *Alpha*, t is the current iteration number, T is the highest number of iterations, and k is a parameter that controls the rate of change of *Alpha*. As the algorithm progresses through the iterations, the value of t increases from 1 to the maximum number of iterations, T . This means that the value of *Alpha* will gradually decrease from *alpha_max* to *alpha_min* over the course of the optimization process. The role of t in this equation is to control the rate of change of *Alpha*. At the beginning of the algorithm (when t is small), the value of (t/T) will be small, and *Alpha* will be closer to *alpha_max*, favoring exploration. As the algorithm progresses (t increases), the value of (t/T) will get closer to 1, and *Alpha* will decrease towards *alpha_min*, favoring exploitation.

The value of k determines the rate at which *Alpha* transitions from the exploration phase to the exploitation phase. A larger value of k will result in a faster decrease in *Alpha*, leading to a quicker shift from exploration to exploitation. Conversely, a smaller value of k will lead to a more gradual decrease in *Alpha*, allowing the algorithm to maintain a balance between exploration and exploitation for a longer period. The appropriate selection of *alpha_max* and *alpha_min*, and k depends on the specific problem and the desired balance between exploration and exploitation. The ISCA algorithm allows for this balance to be dynamically adjusted, which can be particularly beneficial for complex optimization problems such as feature selection in network intrusion detection.

By incorporating the time-varying parameter *Alpha*, the ISCA algorithm aims to enhance the performance of the original SCA algorithm by ensuring a more effective exploration of the search space in the early stages and a more focused exploitation of the promising regions towards the end of the optimization process.

At the beginning of the algorithm, the value of *Alpha* is close to *alpha_max*, which favors exploration and allows the particles to investigate a more expansive search area. As the algorithm progresses, the value of *Alpha* gradually decreases towards *alpha_min*, which favors exploitation and allows the particles to focus on exploiting the local optima. Figure 1 shows the flowchart of the ISCA algorithm.

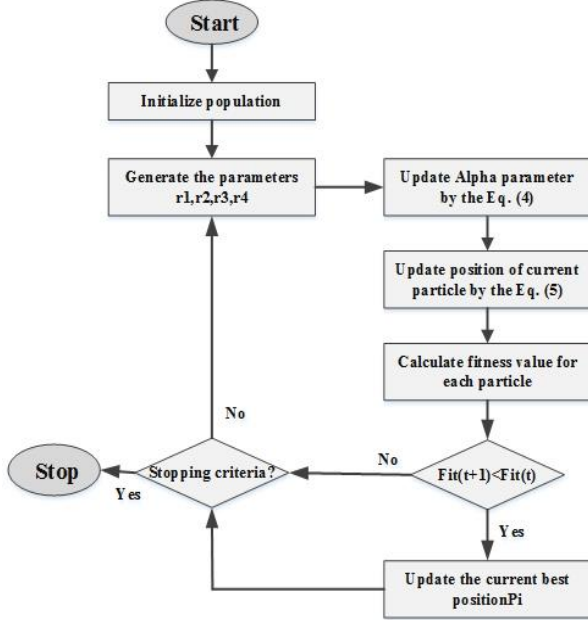


Figure 1. Flowchart of the ISCA Algorithm

4.1. Proposed ISCA Algorithm for feature selection

In this part, we introduce a model based on the ISCA for solving feature selection in IDSs. To solve this problem, we use the ISCA algorithm with a modification that creates a binary space by mapping the continuous search space, using the V-shaped transfer function. The transfer function with a V-shaped is a hyperbolic function that maps a real value in the range $[0, 1]$ to a probability within the range $[0, 1]$. The probability values are calculated by Eq. 5 [32]:

$$V(x_i^d(t+1)) = |\tanh(x_i^d(t))| \quad (5)$$

where $x_i^d(t)$ reveals the i^{th} particle's position with dimension d and iteration t . Following the computation of the probability values, the position of each particle is calculated by Eq. 6.

$$b_i^d(t+1) = \begin{cases} x_i^d(t)^{-1} & \text{if } r < V(x_i^d(t+1)) \\ x_i^d(t) & \text{if } r \geq V(x_i^d(t+1)) \end{cases} \quad (6)$$

where, $b_i^d(t+1)$ is the i^{th} particle's binary position in dimension d at iteration $t+1$. Also, r is a random number in $[0, 1]$.

The ISCA algorithm tackles the feature selection problem by using a population where each member represents a potential solution, i.e., a subset of available

features. Each solution $x_i(t)$ in the population is a binary vector in which each element corresponds to a feature. A value of 1 represents the selection of a feature, while a value of 0 indicates non-selection. We adapt the ISCA algorithm for the feature selection problem by specifying the objective function that reflects the balance between feature count and accuracy chosen.

Let X represent the $N * D$ feature matrix, where N denotes the number of samples and D the number of features. Let Y be the corresponding label vector with dimensions $N * 1$. Our objective is to find a subset of features F that reduces the number of selected features while increasing classification accuracy. Moreover, to assess the quality of solutions, we employ a K-Nearest Neighbors (KNN) classifier [33]. Every time the algorithm runs, a subset of the features is selected by the solution, and the KNN is trained using the samples of data that correspond to the selected feature subset. The KNN classifier's accuracy is then calculated using the remaining data samples that were not used for training. This procedure helps to evaluate the performance of the solution regarding its capacity to classify new data samples accurately based on the selected features. The objective function is defined as follows:

$$Fit_i = \alpha E + \beta \frac{|F|}{D}$$

where E is the KNN classifier's error rate trained using the selected features, D is the dataset's total features, and $|F|$ denotes the number of features that have been chosen. The weighting factors β and α check the proportionate significance of feature subset size and accuracy, respectively. In this paper, we set α to 0.99 and β to 0.01 based on previous studies [34].

5. Experimental results

In this section, the efficiency of the ISCA algorithm proposed in this paper is evaluated in selecting the optimal features for network intrusion detection. For this purpose, experiments were conducted using the dataset NSL-KDD by the ISCA algorithm and other competing metaheuristic algorithms including HHO [35], SSA [36], GWO [37], and DE [38] algorithms and two state-of-the-art models. HHO is a population-based algorithm inspired by the cooperative hunting behavior of Harris hawks. It utilizes exploration and exploitation phases to efficiently search the solution space. SSA is a swarm-based optimization algorithm that mimics the foraging behavior of salps. It uses a leader-follower structure to guide the swarm towards the global optimum. GWO is a nature-inspired algorithm that simulates the social hierarchy and hunting mechanism of grey wolves. It employs an adaptive mechanism to balance exploration and exploitation. DE is an efficient optimization algorithm that utilizes mutation, crossover, and selection operations to evolve a population of candidate solutions towards the global optimum.

The KDD Cup99 dataset has been changed to create a preprocessed NSL-KDD dataset to remove duplicate records and reduce its size. The data includes simulated network traffic data, including various types of attacks, including probe attacks, remote-to-local (R2L), user-to-

root (U2R), and denial of service (DoS) attacks [39]. By using the NSL-KDD data, we can assess how well our suggested algorithm works in detecting and classifying several attack kinds commonly found in network traffic data. The preprocessing steps utilized for the dataset ensure that the data is representative of real-world scenarios and can provide reliable evaluation results.

To ensure a fair comparison, all experiments are conducted in an identical experimental setup. We implement the ISCA and run all comparative algorithms in MATLAB R2019b programming language. We conduct 20 independent runs of all algorithms on a desktop PC equipped with a 2.40 GHz Intel® Core™ i5 processor and 6.0 GB of RAM. The experiments are conducted using the MATLAB 2019b platform and run on Windows 7.

5.1. Data preprocessing

To perform the procedure for data mining and apply classification algorithms on the intrusion detection datasets, it is necessary to perform the preprocessing stage [40]. Some of the most significant pre-processing procedures in this study including data transfer, data normalization, and feature selection are performed. At first, all symbolic data are transferred into numerical values. Also, the data related to the data class are identified by the numbers 0 and 1, where 0 identifies the normal class and 1 indicates the attack class. Also, duplicate records are removed and missing values are managed. In the next step, data normalization is done. In the scaling process, each feature's data values are placed in a proportionate range. In this study, data normalization is done in the range [0, 1], based on the Eq. 8 [41].

$$X_{normalized} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (8)$$

Where, X_{max} indicates the highest possible value of the X feature, and X_{min} determined the lowest possible number of the X feature. The $X_{normalized}$ value is the normalized equivalent of the X feature. Finally, the objective of the feature selection is to minimize the number of features in the dataset to boost the effectiveness of classification and reduce the complexity of calculations. In this study, a wrapper-based technique is proposed to minimize the number of data set features using an improved algorithm.

5.2. Parameter settings of algorithms and Evaluation metrics

For all experiments, we employ the KNN as a classifier with $k = 5$ to classify features obtained from the algorithms. To train the KNN, we perform K-fold cross-validation on each dataset, where K is set to 10. The dataset is split up into K identical pieces at random, with the remaining $K-1$ folds being employed for training and one-fold serving as the testing set. To ensure a fair comparison, we conduct 20 independent runs of all algorithms, using a uniform random distribution used to create the starting population. The highest number of iterations and population size are set at 100 and 20, respectively, across all algorithms. The algorithms' parameter configurations are consistent with their initial papers and are summarized in Table 1.

The proposed ISCA and comparative algorithms are evaluated according to a range of performance measures, such as the mean fitness value, size of selected features, and accuracy.

With D being the total count of features in the original data and $Avg.size^m$ being the mean number of features picked from the dataset, Eq. (9) calculates the mean of the proportion of chosen features to complete features across 20 runs.

$$Average\ selection = \frac{1}{20} \sum_{m=1}^{20} \frac{Avg.size^m}{D}$$

The fitness values' average is determined by the mean fitness value by running each of the algorithms 20 times independently as follows:

$$Mean\ fitness = \frac{1}{20} \sum_{m=1}^{20} f \quad (10)$$

Eq. (11) formulates the average accuracy value, which is the mean of the classification accuracy values acquired by executing the method 20 times. $Accuracy^m$ is the accuracy obtained from the m runs.

$$Average\ Accuracy = \frac{1}{20} \sum_{m=1}^{20} Accuracy^m$$

Table 1. Parameters setting of algorithms

Algorithms	Parameters
GWO	A= [2,0] (Linearly decreasing)
DE	CR= 0.7
SCA	r1=[2,0] (Linearly decreasing), r2=[0, 2] r3=[0, 2] , r4=[0, 1]
SSA	C1= [2,0] (Linearly decreasing)
HHO	E= [2, -2] →0 (Linearly decreasing)

Table 2. Comparison of ISCA with competing algorithms of 20 runs

Algorithms	Accuracy	Fitness value	No. of features
BHHO	0.927	0.073	18
BSSA	0.925	0.075	21
BGWO	0.923	0.077	24
BDE	0.923	0.077	24
BHOA[42]	0.941	0.059	16.7
Alazzam et al.[22]	0.869	0.131	18
ISCA	0.998	0.002	14

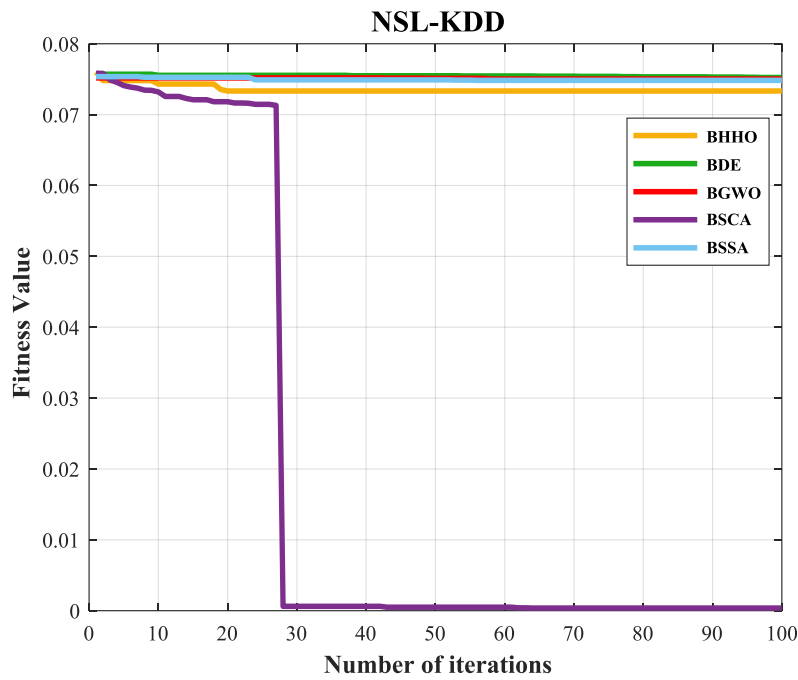


Figure 2. Convergence comparison of the ISCA and comparative algorithms

5.3. Numerical results and discussion

In this subsection, the outcomes of the ISCA algorithm are verified in contrast to a few cutting-edge feature selection techniques introduced in previous subsections. All similar methods have the same parameter values, and each algorithm yields results after 20 runs. Table 2 compares the ISCA algorithm with other competing algorithms regarding fitness value, accuracy, and the number of selected features on the NSL-KDD dataset. The obtained numerical outcomes in the table prove that the ISCA algorithm with a classification accuracy of 0.998 has a higher efficiency than other algorithms, and its fitness value is also the lowest among competitors with a value of 0.002. Regarding the quantity of features selected by the ISCA algorithm is less than other competing algorithms.

In addition to the basic meta-heuristic algorithms, two other improved meta-heuristic algorithms were also considered for comparison. Both the BHOA[42] algorithm and the method proposed by Alazzam et al.[22]. In terms of accuracy, fitness, and the number of selected features, they are weaker than the proposed ISCA algorithm. But

BHOA is considered the best algorithm after ISCA and is superior to other competitors.

Also, the convergence curve of the compared metaheuristic algorithms is presented in Figure 2. According to this figure, ISCA outperformed all competitive algorithms in convergence over the NSL-KDD dataset. Thus, the comparison of ISCA convergence to other algorithms proved its superior performance regarding fitness value. The convergence curve of the ISCA algorithm shows that the presence of the controller parameter introduced in the algorithm causes it to depart from the local optimum and converge to the general optimal solution.

The dynamic exploration-exploitation balance and the sine-cosine-based updates enable ISCA to identify the optimal or near-optimal feature subset, leading to improved classification accuracy compared to the original SCA. Moreover, the adaptive exploration-exploitation mechanism in ISCA accelerates the convergence of the algorithm, allowing it to reach the optimal solution in fewer iterations than the original SCA.

Friedman test (significance level of 0.05)		
Statistic	p-value	Result
562.42077	0.00000	H0 is rejected

Ranking	
Rank	Algorithm
1.00000	ISCA
2.00000	BHOA
3.00000	BHHO
4.15000	BSSA
5.40000	BGWO
5.45000	BDE
7.00000	Alazzam et al.

Figure 3 . Friedman test results

The implementation complexity of the proposed ISCA is $O(N \times M)$, where N represents the number of algorithm iterations and M is the population size. This is because the algorithm performs a fixed number of computations for each feature, including sine and cosine calculations, as well as the update step, and this is repeated for each iteration and over the entire population. However, in the proposed method, the computational complexity has not increased compared to the original SCA, and it remains at $O(N \times M)$. This complexity is lower than that of other comparative algorithms used for comparison, which means that the computation time of the improved SCA is also lower than its competitors.

Friedman's statistical test can be used to compare the performance of the ISCA algorithm with other algorithms for feature selection in intrusion detection. The results of Friedman's test are displayed in Fig. 3, comparing the efficiency of the ISCA algorithm to that of its competitors based on fitness values. The figure illustrates that this algorithm outperformed other competing algorithms in terms of rank. Therefore, the ISCA algorithm has shown its effectiveness in addressing optimization problems, especially binary problems like feature selection.

The reasons for the high efficiency of the ISCA algorithm are: The ISCA algorithm employs a dynamic adjustment of the exploration-exploitation balance through the α parameter. This parameter is controlled by the iteration count (t) and the maximum number of iterations (T). The dynamic balance allows the algorithm to effectively explore the search space in the early stages and gradually shift towards exploitation as the optimization progresses. This leads to more accurate identification of the optimal feature subset. The feature selection mechanism of the ISCA algorithm combines the sine and cosine functions in a novel way, allowing for a more comprehensive exploration of the feature space. This enhanced feature selection approach enables the algorithm

to more effectively identify the most relevant features, resulting in a compact set of selected features while maintaining high classification accuracy.

6. Conclusion

Detecting network intrusions is a crucial assignment in preserving computer networks' integrity and security. However, network traffic data's high dimensionality presents a problem for intrusion detection systems, as it can result in a lot of features, many of which can be irrelevant or redundant. To overcome this challenge, feature selection is an essential stage in creating efficient intrusion detection systems, as it helps to minimize the dimensionality of the feature space and increase the classification model's accuracy. In this study, we proposed an improved model of the SCA to select effective features in network intrusion detection. Our proposed algorithm introduced a controlling parameter to balance exploration and exploitation, which enhanced the accuracy and addressed the dimensionality curse of the classification model. We evaluated the efficiency of our introduced ISCA algorithm on the NSL-KDD data and showed that it outperformed other metaheuristic algorithms in the number of features chosen and the accuracy of classifier. The proposed ISCA algorithm can be used as an effective tool for building intrusion detection systems that are both accurate and efficient. By lowering the feature space's dimensionality, our proposed algorithm helps to focus on the most essential characteristics and raise the classification model's accuracy.

Although our proposed algorithm showed promising results, there are still several avenues for future research. One direction is to investigate the use of hybrid algorithms that combine metaheuristic algorithms with other optimization techniques. Another direction is to evaluate the proposed algorithm's performance on other datasets and in real-world intrusion detection systems. Finally,

more research is needed to investigate the interpretability of the selected features and their relevance to different types of cyber threats.

Declarations:

Authors' contributions: Conceptualization, Z.Asghari; methodology, Z.Asghari.; software, Z. Asghari; validation, Z. Asghari and S.Hosseini; formal analysis, Z. Asghari and S.Hosseini; investigation, S.Hosseini; resources, Z. Asghari; data curation, S.Hosseini.; writing—original draft preparation, Z. Asghari; writing—review and editing, S.Hosseini.; visualization, Z. Asghari.; supervision, S.Hosseini.; project administration, S.Hosseini.;

Funding: This research received no external funding.

Data Availability declaration: The datasets analyzed during the current study are available at the, <https://iee-dataport.org/documents/nsf-kdd-0>. For the academic/public use of this dataset, the authors have to cite original papers.

Conflict of Interest: The authors declare that they have no conflict of interest.

Human Participants and/or Animals: This article does not contain any studies with human participants and/or animals performed by any of the authors.

Ethics approval and consent to participate: Not applicable.

Acknowledgments: Not applicable.

7. References

- [1] G. A. Marin. (2005, Dec.). Network security basics. *IEEE security & privacy*. [Online]. 3(6), pp. 68-72. Available: <https://doi.org/10.1109/MSP.2005.153>
- [2] *Computer and information security handbook*, 2nd ed., J. R. Vacca Co., 2012.
- [3] M. Ahmed, A. N. Mahmood, and J. Hu. (2016, Jan.). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*. [Online]. 60, pp. 19-31. Available: <https://doi.org/10.1016/j.jnca.2015.11.016>
- [4] V. Z. Asghari and M. Kuchaki Rafsanjani. (2021, Jun.). Intrusion detection system using a new fuzzy rule-based classification system based on genetic algorithm. *Intelligent Decision Technologies*. [Online]. 15(2), pp. 231-237. Available: <https://doi.org/10.3233/IDT-200036>
- [5] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung. (2013, Jan.). Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*. [Online]. 36(1), pp. 16-24. Available: <https://doi.org/10.1016/j.jnca.2012.09.004>
- [6] *Handbook of research on modern cryptographic solutions for computer and cyber security*, B. Gupta, D. P. Agrawal, and S. Yamaguchi Co., IGI global, 2016.
- [7] J. M. Vidal, M. A. S. Monge, and S. M. M. Monterrubio. (2020). Anomaly-based intrusion detection: adapting to present and forthcoming communication environments. in *Handbook of Research on Machine and Deep Learning Applications for Cyber Security*. [Online]. pp. 195-218. Available: <https://doi.org/10.4018/978-1-5225-9611-0.ch010>
- [8] M. K. Rafsanjani and Z. A. Varzaneha. (2013). Intrusion detection by data mining algorithms: a review. *Journal of New Results in Science*. [Online]. 2(2), pp. 76-91. Available: <https://dergipark.org.tr/en/pub/jnrs/issue/27513/123346>
- [9] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," Department of Computer Engineering Chalmers University of Technology Göteborg, 2000.
- [10] M. A. Ambusaidi, X. He, P. Nanda, and Z. Tan. (2016, Jan.). Building an intrusion detection system using a filter-based feature selection algorithm. *IEEE transactions on computers*. [Online]. 65(10), pp. 2986-2998. Available: <https://doi.org/10.1109/TC.2016.2519914>
- [11] A. Alazab, M. Hobbs, J. Abawajy, and M. Alazab, "Using feature selection for intrusion detection system," in *2012 international symposium on communications and information technologies (ISCIT)*, Gold Coast, QLD, Australia, 2012, pp. 296-301.
- [12] S. Aljawarneh, M. Aldwairi, and M. B. Yassein. (2018, Mar.). Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model. *Journal of Computational Science*. [Online]. 25, pp. 152-160. Available: <https://doi.org/10.1016/j.jocs.2017.03.006>
- [13] Z. A. Varzaneh, S. Hossein, S. E. Mood, and M. M. Javidi. (2022, Sep.). A new hybrid feature selection based on Improved Equilibrium Optimization. *Chemometrics and Intelligent Laboratory Systems*. [Online]. 228, p. 104618. Available: <https://doi.org/10.1016/j.chemolab.2022.104618>
- [14] J. Huang, Y. Chen, A. A. Heidari, L. Liu, H. Chen, and G. Liang. (2024, Jun.). Improved Runge Kutta Optimization Using Compound Mutation Strategy in Reinforcement Learning Decision Making for Feature Selection. *Journal of Bionic Engineering*. [Online]. 21, pp. 1-37. Available: <https://doi.org/10.1007/s42235-024-00555-x>
- [15] A. I. Hammouri, M. A. Awadallah, M. S. Braik, M. A. Al-Betar, and M. Beseiso. (2024, May.). Improved Dwarf Mongoose Optimization Algorithm for Feature Selection: Application in Software Fault Prediction Datasets. *Journal of Bionic Engineering*. [Online]. 21, pp. 1-34. Available: <https://doi.org/10.1007/s42235-024-00524-4>
- [16] M. H. Nadimi-Shahraki, H. Zamani, Z. A. Varzaneh, A. S. Sadiq, and S. Mirjalili. (2024, Jul.). A Systematic Review of Applying Grey Wolf Optimizer, its Variants, and its Developments in Different Internet of Things Applications. *Internet of Things*. [Online]. 26, p. 101135. Available: <https://doi.org/10.1016/j.iot.2024.101135>
- [17] M. Abdel-Basset, L. Abdel-Fatah, and A. K. Sangaiah. (2018, Jan.). Metaheuristic algorithms: A comprehensive review. *Computational intelligence*

- for multimedia big data on the cloud with engineering applications. [Online]. pp. 185-231. Available: <https://doi.org/10.1016/B978-0-12-813314-9.00010-4>
- [18] X.-S. Yang, "Nature-inspired metaheuristic algorithms," in Luniver press, 2nd ed., United Kindom, 2010.
- [19] G. Hu, Y. Guo, and G. Sheng. (2024, May.). Salp Swarm Incorporated Adaptive Dwarf Mongoose Optimizer with Lévy Flight and Gbest-Guided Strategy, *Journal of Bionic Engineering*. [Online]. 21, pp. 1-35. Available: <https://doi.org/10.1007/s42235-024-00545-z>
- [20] S. Mirjalili. (2016, Mar.). SCA: a sine cosine algorithm for solving optimization problems. *Knowledge-based systems*. [Online]. 96, pp. 120-133. Available: <https://doi.org/10.1016/j.knosys.2015.12.022>
- [21] M. H. Nadimi-Shahraki, H. Zamani, Z. Asghari Varzaneh, and S. Mirjalili. (2023, May.). A systematic review of the whale optimization algorithm: theoretical foundation, improvements, and hybridizations. *Archives of Computational Methods in Engineering*. [Online]. 30(7), pp. 4113-4159. Available: <https://doi.org/10.1007/s11831-023-09928-7>
- [22] H. Alazzam, A. Sharieh, and K. E. Sabri. (2020, Jun.). A feature selection algorithm for intrusion detection system based on pigeon inspired optimizer. *Expert systems with applications*. [Online]. 148, p. 113249. Available: <https://doi.org/10.1016/j.eswa.2020.113249>
- [23] J. R. Beulah and D. S. Punithavathani. (2018, Sep.). A hybrid feature selection method for improved detection of wired/wireless network intrusions. *Wireless Personal Communications*. [Online]. 98(2), pp. 1853-1869. Available: <https://doi.org/10.1007/s11277-017-4949-x>
- [24] R. Zhao, Y. Mu, L. Zou, and X. Wen. (2022, Jun.). A hybrid intrusion detection system based on feature selection and weighted stacking classifier. *IEEE Access*. [Online]. 10, pp. 71414-71426. Available: <https://doi.org/10.1109/ACCESS.2022.3186975>
- [25] R. Vijayanand and D. Devaraj. (2020, Mar.). A novel feature selection method using whale optimization algorithm and genetic operators for intrusion detection system in wireless mesh network. *IEEE Access*. [Online]. 8, pp. 56847-56854. Available: <https://doi.org/10.1109/ACCESS.2020.2978035>
- [26] A. K. Shukla. (2020, Nov.). Detection of anomaly intrusion utilizing self-adaptive grasshopper optimization algorithm. *Neural Computing and Applications*. [Online]. 33(13), pp. 7541-7561. Available: <https://doi.org/10.1007/s00521-020-05500-7>
- [27] F. Salo, A. B. Nassif, and A. Essex. (2019, Jan.). Dimensionality reduction with IG-PCA and ensemble classifier for network intrusion detection. *Computer network*. [Online]. 148, pp. 164-175. Available: <https://doi.org/10.1016/j.comnet.2018.11.010>
- [28] I. Ahmed, A. Dahou, S. A. Chelloug, M. A. Al-qaness, and M. A. Elaziz. (2022, Mar.). Feature selection model based on gorilla troops optimizer for intrusion detection systems. *Journal of Sensors*. [Online]. (1), pp. 1-12. Available: <https://doi.org/10.1155/2022/6131463>
- [29] M. Safaldin, M. Otair, and L. Abualigah. (2020, Jun.). Improved binary gray wolf optimizer and SVM for intrusion detection system in wireless sensor networks. *Journal of ambient intelligence and humanized computing*. [Online]. 12, pp. 1559-1576. Available: <https://doi.org/10.1007/s12652-020-02228-z>
- [30] D. Jayalatchumy, R. Ramalingam, A. Balakrishnan, M. Safran, and S. Alfarhood. (2024, Mar.). Improved Crow Search-Based Feature Selection and Ensemble Learning for IoT Intrusion Detection. *IEEE Access*. [Online]. 12, pp. 33218 - 33235. Available: <https://doi.org/10.1109/ACCESS.2024.3372859>
- [31] U. Nandhini and S. K. SVN. (2024, Jun.). An improved Harris Hawks optimizer based feature selection technique with effective two-staged classifier for network intrusion detection system. *Peer-to-Peer Networking and Applications*. [Online]. 17, pp. 1-35. Available: <https://doi.org/10.1007/s12083-024-01727-6>
- [32] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi. (2010, Dec.). BGSA: binary gravitational search algorithm, *Natural computing*. [Online]. 9, pp. 727-745. Available: <https://doi.org/10.1007/s11047-009-9175-3>
- [33] P. Cunningham and S. J. Delany. (2021, Jul.). k-Nearest neighbour classifiers-A Tutorial. *ACM computing surveys (CSUR)*. [Online]. 54(6), pp. 1-25. Available: <https://doi.org/10.1145/3459665>
- [34] N. Al-Madi, H. Faris, and S. Mirjalili. (2019, Dec.). Binary multi-verse optimization algorithm for global optimization and discrete problems. *International Journal of Machine Learning and Cybernetics*. [Online]. 10(12), pp. 3445-3465. Available: <https://doi.org/10.1007/s13042-019-00931-8>
- [35] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen. (2019, Aug.). Harris hawks optimization: Algorithm and applications. *Future generation computer systems*. [Online]. 97, pp. 849-872. <https://doi.org/10.1016/j.future.2019.02.028>
- [36] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili. (2017, Dec.). Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Advances in engineering software*. [Online]. 114, pp. 163-191. Available: <https://doi.org/10.1016/j.advengsoft.2017.07.002>
- [37] S. Mirjalili, S. M. Mirjalili, and A. Lewis. (2014, Mar.). Grey wolf optimizer. *Advances in engineering software*. [Online]. 69, pp. 46-61. Available: <https://doi.org/10.1016/j.advengsoft.2013.12.007>
- [38] R. Storn and K. Price. (1997, Dec.). Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*. [Online]. 11, pp. 341-359.

Available:


<https://doi.org/10.1023/A:1008202821328>

- [39] S. Revathi and A. Malathi. (2013, Dec.). A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection. *International Journal of Engineering Research & Technology (IJERT)*. [Online]. 2(12), pp. 1848-1853.
 - [40] S. García, J. Luengo, and F. Herrera, *Data preprocessing in data mining*. Cham, Switzerland: Springer International Publishing, 2015, pp. 59-139.
 - [41] S. K. Sahu, S. Sarangi, and S. K. Jena, "A detail analysis on intrusion detection datasets," in *2014 IEEE international advance computing conference (IACC)*, Gurgaon, India, 2014, pp. 1348-1353.
 - [42] Z. Asghari Varzaneh, S. Hosseini, and M. M. Javidi. (2023, Mar.). A novel binary horse herd optimization algorithm for feature selection problem. *Multimedia Tools and Applications*. [Online]. 82(26), pp. 1-35. Available: <https://doi.org/10.1007/s11042-023-15023-7>
-



Evaluating Developers' Expertise in Serverless Functions by Mining Activities from Multiple Platforms *

Research Article

Aref Talebzadeh Bardsiri¹, Abbas Rasoolzadegan² 

DOI: [10.22067/cke.2024.84447.1103](https://doi.org/10.22067/cke.2024.84447.1103)

Abstract In the domain of software development, the evaluation of developer expertise has gained prominence, particularly with the rise of serverless functions. These functions, which simplify the development process by delegating infrastructure management to cloud providers, are becoming more common. As developers may utilize functions created by their peers, understanding the expertise of the original developer is crucial since it can serve as an indicator of the functions' quality. While there are existing methods for expertise evaluation, certain gaps remain, especially concerning serverless functions. To address this, our research aims to enhance the assessment of developer expertise in this area by extracting activity-based features from both GitHub and Stack Overflow. After processing the extracted data, we applied various machine learning algorithms. Our findings suggest a potential improvement in evaluating developer expertise when incorporating features from Stack Overflow compared to using only GitHub data. The extent of this improvement was observed to differ among programming languages, with variations in accuracy improvement percentages ranging from 2% to 19%. This study contributes to the ongoing discourse on developer expertise evaluation, highlighting the potential benefits of drawing from multiple data sources.

Keywords: *Developer Expertise Evaluation, Data Analysis, Machine Learning Algorithms, Serverless Functions, Software Development.*

1. Introduction

In the domain of software development, the ability to accurately evaluate developer expertise has become paramount [1-5]. This emphasis on expertise evaluation is not just a theoretical concern but has practical implications, especially in the evolving landscape of

serverless functions. Serverless functions, often referred to as Function as a Service (FaaS) [6], simplify the development process by offloading infrastructure management to cloud providers. Recent data suggests that over 40% of companies have integrated serverless functions into their workflows, drawn by their scalability, cost-effectiveness, and the convenience of reduced infrastructure management [7][8].

With the increasing adoption of serverless functions, there's a growing need to understand the expertise behind the functions being developed. Developers frequently integrate functions developed by others into their projects. In such contexts, assessing the expertise of the original developer is crucial to ensure the reliability and efficiency of the integrated functions. Evaluating developer expertise in serverless functions is particularly important as it can significantly impact the quality and performance of the applications that use these functions.

Furthermore, as serverless functions are often developed using "Target Languages" like Java, Python, NodeJs³, Ruby, Go, and C#—selected for their compatibility with serverless architectures and their widespread use [9] it becomes imperative to evaluate expertise specifically within these languages to ensure the quality and efficiency of serverless applications.

The accurate assessment of developer expertise in the broader software development field has been the focus of numerous investigations [1-4], [10-14]. While many works have been conducted in this area, several challenges persist. For instance, some studies have found that simple metrics, such as counting commits, might not be a reliable indicator of expertise within specific libraries or frameworks. Others have introduced tools that leverage Natural Language Processing (NLP) to pinpoint expertise, but these often rely solely on data from a single platform like GitHub. There's also a recognized need to merge data

* Manuscript received: 2023 September 15, Revised, 2024 February 14, Accepted, 2024 March 18.

¹ Graduate Student, Department of Computer Engineering, Ferdowsi University of Mashhad, Iran.

² Corresponding author. Associate Professor, Department of Computer Engineering, Ferdowsi University of Mashhad, Iran. **Email:** rasoolzadegan@um.ac.ir

³ While NodeJs technically serves as a runtime environment facilitating the execution of JavaScript code on the server side, it is often colloquially referred to as a programming language due to its prevalent standalone usage in discourse. In this paper, we refer to it as one of our target languages

from multiple platforms, such as GitHub and Stack Overflow, but many existing approaches primarily focus on user profiles and specific APIs.

Building on these valuable insights from prior research, our study endeavors to bridge the identified gaps. Specifically, we aim to offer a fresh perspective on developer expertise by tapping into both GitHub and Stack Overflow, thereby opening a broader window of feature collecting, especially in the context of serverless functions. We've observed that while some research has adeptly employed machine learning classifiers, a predominant reliance on a single data platform suggests an opportunity for enhancement. Our work underscores the significance of adopting a multi-platform approach, which we believe can pave the way for a more holistic understanding of developer expertise.

Evaluating developer expertise, especially within serverless functions, requires a systematic approach. Following this notion, our research utilized the official GitHub REST API⁴ for data extraction. We initially identified 408 GitHub repositories related to serverless functions. From these, we selected the top 150 to ensure representation across different target languages. On GitHub, we extracted 13 activity-related features, providing insights into a contributor's activities and expertise. Our interpretation of these features was informed by Montandon's work [11].

Turning to Stack Overflow, we sought to link contributors to their Stack Overflow profiles using the official StackAPI⁵. This allowed us to extract 9 additional features related to their activity on this platform. After data collection, we invited contributors to self-assess their expertise on a 0 to 5 scale. Of the 2539 emails we sent, 237 were answered, leading to a response rate of about 9.3%. This feedback aided in initially labeling our dataset.

After data extraction, we engaged in preprocessing to manage data-related challenges. For analysis, we employed machine learning algorithms like SVM [15], [16], Random Forest [17], Gradient Boosting [18], and Logistic Regression [19]. Initial results showed a preference for SVM and Random Forest in several datasets. When compared to other research, our findings suggested potential benefits from using data from both Stack Overflow and GitHub. The efficacy varied by target language: NodeJs exhibited an accuracy increase of approximately 19%, while C# showed an increase of about 2%, resulting in an average improvement of around 10.7%. Having outlined our proposed method, we sought to address these two specific research questions:

RQ1. Which machine learning algorithms are most effective in evaluating developer expertise in serverless functions based on the extracted features? The answers

to this question, based on our comparative analyses of different algorithms, are discussed in the 'Evaluation' section.

RQ2. What features or metrics are most indicative of a developer's expertise in serverless functions? The insights related to this question, derived from our feature importance analysis, can be found in the 'Evaluation' section, specifically in the third part of that section.

In our research, we've made several contributions to evaluating developer expertise in serverless functions. Notably, we've adopted a dual-platform data extraction approach, gathering activity-based features of contributors⁶ from both GitHub and Stack Overflow. This approach aims to provide a more detailed perspective on a developer's engagement by leveraging data from two major platforms. From this extraction, we've compiled six language-specific datasets, representing developer activities in serverless functions for the respective target languages. Importantly, by making these datasets publicly available, we aim to foster collaborative research and encourage further exploration in this domain. This detailed approach allows for a nuanced evaluation based on the specific programming language. Additionally, we've conducted a feature importance analysis using SHAP values to understand the relative importance of each extracted feature. This step helps in discerning which activities might be more indicative of a developer's expertise in serverless functions.

Our proposed method caters not only to serverless functions but also have versatility for broader software development contexts. This adaptability accentuates the potential of our techniques in navigating the multifaceted challenges of contemporary software development. Furthermore, our research underscores the imperative of a holistic, multi-platform approach in developer expertise evaluation, with implications for refining recruitment strategies in the industry and fostering a platform proficiency within academic frameworks.

The remainder of this paper is structured as follows: The next section delves into the Background, providing a foundational understanding of the domain and contextualizing our work within existing literature. Following this, we present our Proposed Method, detailing the approach and techniques we employed. The Evaluation section then discusses our findings, shedding light on the efficacy and implications of our method. We subsequently address potential Threats to Validity, ensuring a transparent and critical discussion of our study's limitations. The paper concludes with a Conclusion section, summarizing our key contributions, and then looks ahead to Future Directions, suggesting potential avenues for further research and exploration in this

⁴ <https://docs.github.com/en/rest?apiVersion=2022-11-28>

⁵ <https://stackapi.readthedocs.io/en/latest>

⁶ In this study, term 'contributors' refers to developers active in serverless functions.

domain.

2. Background

The rapid evolution of software development has led to the emergence of various platforms where developers collaborate, share knowledge, and showcase their expertise. Platforms like GitHub and Stack Overflow have become central to this ecosystem, providing a wealth of data that can be mined to understand developer expertise and behavior. Several studies have delved into this realm, each offering unique insights and methodologies [5].

Vasilescu et al. embarked on a comprehensive exploration of the intricate relationship between Stack Overflow and GitHub activities. Their exploration spanned three distinct levels: macro, intermediate, and micro. Their macro-level analysis aimed to discern the differences between GitHub contributors based on their Stack Overflow involvement, probing whether activity on one platform could serve as a proxy for the other. The intermediate level delved into the distribution of developers' time between GitHub commits and their Q&A activity on Stack Overflow. At the micro level, the temporal coordination between GitHub commits and Stack Overflow Q&A activities was scrutinized. This multifaceted analysis underscores the intertwined nature of developer activities across these platforms, emphasizing the potential influence of participation on one platform over the other [12].

In a similar vein, Song et al. sought to profile developer expertise by harnessing data from both Stack Overflow and GitHub. The research underscored the challenges of profiling expertise based solely on a single platform, given the sparsity of expertise matrices for both Stack Overflow and GitHub. By integrating data from both platforms, the study illuminated the multifaceted nature of developer expertise, underscoring the potential benefits of a cross-community approach. Such a collaboration-aware method can potentially mitigate challenges like unanswered questions on Stack Overflow and delayed responses to pull requests on GitHub [2].

The realm of developer expertise assessment witnessed a novel approach with the introduction of "CVExplorer," a tool designed to identify potential developer candidates by meticulously analyzing their contributions to open-source projects on GitHub. By mining skills from GitHub contributions, the tool offers recruiters a more accurate representation of a developer's skills, emphasizing the importance of real-world contributions in assessing developer expertise. This approach, which transcends the traditional reliance on self-authored CVs, underscores the evolving paradigms in developer assessment and recruitment [14].

Building on this Constantinou and Kapitsaki delved deep into the nuances of developer expertise and their roles in software technologies. Their research introduced the concept of "core expertise," signifying the primary domain or technology group where a developer is most prolific. By employing various metrics and data from platforms like Stack Overflow and GitHub, the study provided a granular analysis of how developers transition between roles and how their expertise can be categorized. Such insights are pivotal in understanding the evolutionary trajectory of

developers and the multifarious roles they assume over time [3].

Tian et al. introduced a novel approach aimed at constructing a cross-platform expert recommendation system by synergizing datasets from GitHub and Stack Overflow. This system, designed to spotlight top expert developers, or "geek talents," underscores the value of expert recommendation systems in the open-source community and for companies at large. By leveraging various attributes of user profiles, platform-specific APIs, and multiple account matching strategies, the system can adeptly identify top experts in specific technology fields. Such a method offers a fresh perspective on recommending top expert developers, emphasizing the increasing importance of these platforms in the software development community [4]. Santos et al. ventured into the domain of mining software repositories with the primary objective of identifying library experts. By analyzing the source code of collaborative projects on GitHub, the study introduced a method that ranks developers based on five dimensions of skills. Preliminary results from this research underscored the method's capability of identifying relevant users of specific libraries, emphasizing the importance of GitHub as a platform to showcase developers' knowledge and skills. Such a structured approach offers a comprehensive evaluation of a developer's proficiency, highlighting the potential benefits for recruitment and human resource allocation [13].

Oliveira et al. embarked on a comprehensive empirical study to identify library experts by analyzing source code. By evaluating the strategy with popular Java libraries and conducting an online survey with developers, the study provided insights into the challenges and methodologies of identifying library experts based on code analysis. The findings underscored that traditional metrics like "Lines of Code" or "Number of Commits" might not be sufficient indicators of a developer's expertise with specific libraries. Such insights are pivotal in understanding the nuances of developer expertise in specific libraries and the potential limitations of certain metrics in the identification process [10].

Lastly, Montandon's research focused on identifying experts in popular JavaScript libraries by mining GitHub data. By integrating repository mining with developer surveys, Montandon provided a robust method for pinpointing expertise in specific software libraries and frameworks. While this approach emphasized the importance of combining multiple data sources for accurate assessments, our research Recognizing the potential limitations in Montandon's approach, we advocate for a broader spectrum of features, underscoring the significance of a more detailed and comprehensive feature set. This expanded perspective is particularly crucial in the realm of serverless functions, where the landscape is rapidly evolving and the nuances of developer expertise are multifaceted [11]. In light of the existing studies, our research aims to enhance the understanding of developer expertise by amalgamating data from both GitHub and Stack Overflow. While we draw inspiration from the works mentioned, our unique contribution lies in

Table 1 Extracted Features from GitHub and Stack Overflow

Platform	Feature	Description
GitHub	commits	Number of total commits of the developer in client projects
GitHub	commits_client_files	Number of commits changing at least one client file
GitHub	commits_import_library	Number of commits in client files adding library import statements
GitHub	code_churn	Code churn considering all commits in client projects
GitHub	code_churn_client_files	Code churn considering only changes in client files
GitHub	imports	Number of added library import statements
GitHub	days_since_first_import	Number of days since the first commit where an import statement was added
GitHub	days_since_last_import	Number of days since the last commit where an import statement was added
GitHub	days_between_imports	Number of days between the first/last commits where an import statement was added
GitHub	avg_days_commits_client_files	Average interval (in days) of the commits changing client files
GitHub	avg_days_commits_import_library	Average interval (in days) of the commits adding import statements
GitHub	projects	Number of client projects the developer contributed at least once
GitHub	projects_import	Number of client projects where the developer added a library import statement
Stack Overflow	answers	Total number of answers provided by the user
Stack Overflow	accepted_answers	Total number of the user's answers that were accepted
Stack Overflow	upvotes	Total upvotes received by the user
Stack Overflow	downvotes	Total downvotes received by the user
Stack Overflow	average_score_per_answer	Average score (upvotes minus downvotes) per answer provided by the user
Stack Overflow	questions	Total number of questions posted by the user
Stack Overflow	first_answers	Number of times the user was the first to answer a question
Stack Overflow	tag_score	Cumulative score of the user in specific tags related to serverless functions or related topics
Stack Overflow	time_of_activity	Duration of the user's activity on Stack Overflow (e.g., from the first post to the most recent)

Step 2. Enriching Profiles with Stack Overflow Features

To augment our dataset, we aimed to identify the Stack Overflow profile corresponding to each GitHub contributor. We began by initiating a search using the contributor's GitHub username. If any list of Stack Overflow profiles was returned, we took additional steps to ensure the authenticity and relevance of the identified profile. Specifically, we applied a method based on probabilistic record linkage [32], a well-established technique in data integration. This method involves calculating a similarity score based on the probabilities of agreement and disagreement for each attribute (website URL, location, bio, and company affiliation). These probabilities were computed based on the distribution of values in each field in our dataset. The profile with the highest similarity score was selected for further analysis. In situations where no profiles met our stringent criteria, we opted for the first profile returned by the search results, given StackAPI's tendency to list the most relevant user first.

To complement our GitHub data, we delved deeply into Stack Overflow, aiming to extract features that would provide a detailed view of each contributor's expertise and

engagement. Starting with the Stack Overflow profile of each contributor, identified in the previous step, we centered our efforts on the target languages. For each language, we:

- (1) Retrieved answers and questions provided by the user, which not only showcased their expertise but also their level of engagement with the community.
- (2) Computed various metrics that reflected the user's overall activity and reputation for that target language (Tag Score) on Stack Overflow. This encompassed the number of answers they've given, upvotes received, downvotes given, and other related metrics. find their detailed representation in Table 1
- (3) Consolidated these features and stored them in a CSV file, ensuring they were organized and ready for subsequent analysis.

Our research method heavily relied on the REST APIs of both GitHub and Stack Overflow. By integrating the GitHub REST API with the PyGitHub Python library, we ensured timely and accurate data extraction from GitHub. Similarly, our use of StackAPI streamlined our interactions with Stack Overflow, particularly in retrieving user profiles and associated data. We opted for these APIs over pre-existing datasets to ensure we were working with

the most up-to-date online data, reinforcing the repeatability of our method. Given the dynamic nature of expertise, our method was designed to be adaptable, allowing for potential replication in future studies. A crucial part of our data collection process was managing the rate limits imposed by both platforms. To ensure a smooth data collection process, we sequentially extracted data, first focusing on GitHub features for each contributor and then moving to Stack Overflow.

Step 3. Creating Language Specific Datasets for Evaluating Developers' Expertise in Serverless Functions We initiated this phase by reaching out to contributors for a self-assessment of their expertise. Each contributor received an email detailing our research objectives and was asked to rate their expertise on a scale from 0 (no expertise) to 5 (expert level). Supported by several studies as a reliable measure of expertise [11], [23], [24], this method yielded a response rate of approximately 9.33% from the 2539 emails dispatched.

After updating the contributors' profiles with their self-assessed scores, we curated separate datasets for each of the target languages: Java, Python, NodeJs, Ruby, Go, and C#. These datasets encapsulated the contributors' activities specific to the respective programming language, with their self-assessed scores serving as the labels. This approach ensured a granular understanding of developer expertise in serverless functions, tailored to the nuances of each language.

Phase 2. Data Analysis

Upon completion of the data collection phase, we obtained six distinct datasets. Each dataset encapsulates 22 features detailing developers' activities on both GitHub and Stack Overflow, specific to each of our target languages. In the next step, we transitioned to the data analysis phase, which is further divided into two main steps:

Step 1- Data Visualization

Our initial exploration began with visualizing the distribution of developer expertise across our target languages, as depicted in Figure 2. For clarity in the figure, we've categorized our binned labels as follows: ratings of 0 and 1 are denoted as "Novice (0)", ratings of 2 and 3 are labeled "Intermediate (1)", and ratings of 4 and 5 are classified as "Expert (2)". We will delve into the binning process in a subsequent section. The visualization reveals compelling insights. For instance, languages like Python and Ruby, which are often considered beginner-friendly, had a significant portion of their developer base in the novice category.

This suggests that these languages are more accessible for beginners. In contrast, languages such as Java, a prevalent choice in enterprise settings, showcased a more balanced expertise distribution, indicating a diverse user base with varying levels of expertise.

Moving forward, we examined the skewness across

different programming languages, visualized in Figure 3. Skewness, a measure indicating the asymmetry of data distribution around the mean, revealed specific patterns. A notable observation was the positive skewness values for commits in languages like Go, NodeJs, and Ruby. This suggests a scenario where a majority of users have fewer commits, but a select few exhibit exceptionally high commit counts. This skewness indicates that while most developers contribute at a moderate pace, there are a few highly active developers who contribute significantly more. Understanding this skewness is pivotal as it influences our data interpretation and subsequent modeling strategies [20]. Our analysis subsequently delved into correlation. It is crucial to emphasize that correlation does not imply causation [21]. Particularly in the realm of expertise, it is common for features to exhibit high correlation. However, even if two variables appear to move in tandem, this does not necessarily indicate a cause-and-effect relationship. For example, we observed a high correlation between the number of commits and the number of pull requests made by a developer. While these two variables are correlated, it does not necessarily mean that making more commits causes a developer to make more pull requests, or vice versa. It could simply be that more active developers tend to both commit and pull request more frequently.

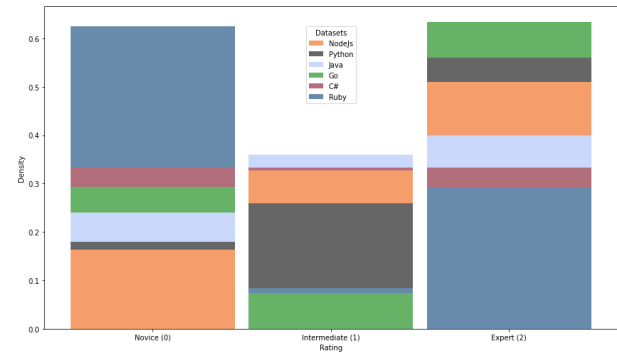


Figure 2. Incremental Distribution of Ratings Across Target Language

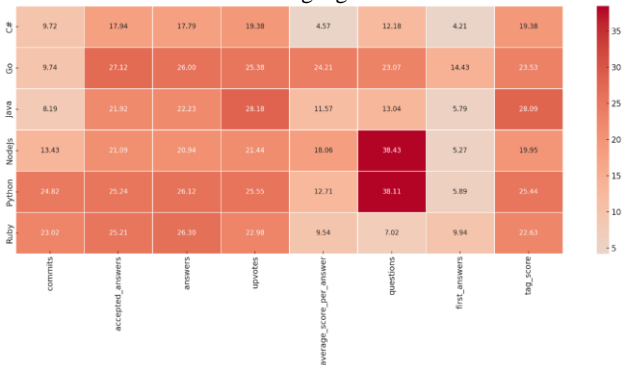


Figure 3. Skewness of Features Across Target Languages

Step 2- Data Preprocessing

After the data analysis phase, we transitioned to data preprocessing, a crucial step to ensure the data was primed for modeling. This phase addressed several challenges, including missing values, skewness, feature selection,

label binning, and label imputation.

Handling Missing Values: In addressing missing values, we observed that their presence in our dataset often signified a developer's inactivity for a specific feature. Given this observation, we logically imputed these missing values with zero, symbolizing no activity.

Addressing Skewness: Skewness presented another challenge. Some features in our dataset displayed right-skewed distributions. To make the data more suitable for modeling, we applied a logarithmic transformation [22] to these skewed features, stabilizing their variance and approximating a normal distribution. However, before this transformation, we had to manage zeros in the data, as the logarithm of zero is undefined. To counteract this, we added a constant of 1 to the respective columns.

Feature Selection: Feature selection emerged as a pivotal aspect of our preprocessing [23]. Ensuring that our model is trained on relevant features is paramount for enhancing its performance and interpretability. Our multifaceted approach to feature selection began with an examination of correlations. While correlation provides a measure of the linear relationship between two variables, it doesn't capture non-linear relationships or causation, making sole reliance on it potentially misleading [21]. We also incorporated domain knowledge for the feature selection, by understanding the context of each feature in the software development realm, we made informed decisions. For example, while the number of answers and accepted answers on Stack Overflow might be correlated, the emphasis on quality in the latter could offer more valuable insights. Additionally, we employed the Gradient Boosting technique, an ensemble learning method, to gain insights into feature importance. This technique ranked features based on their significance in influencing the model's predictions. Armed with these analyses, we made informed decisions on which features to discard, ensuring our model was built on a relevant feature set, as visualized in figure 4.

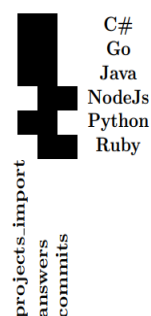


Figure 4: Dropped Features for Each Programming Language

Label Binning: Following feature selection, we addressed the challenge of label binning. Our dataset contained ratings ranging from 0 to 5, indicative of the expertise level. To streamline our modeling process and make the problem more approachable, we categorized these ratings into three broader categories. Ratings between 0 and 1

were represented as Novice (0), those between 2 and 3 as Intermediate (1), and ratings between 4 and 5 were categorized as Expert (2). This categorization ensured our models could predict expertise levels in broader, more generalizable categories, enhancing the interpretability and applicability of the results.

Label Imputation: Transitioning from the binning process, we confronted another intricate challenge: Label Imputation. In the vast landscape of data-driven research, addressing the absence of labels is a multifaceted endeavor. To adeptly navigate this, we adopted a dual-strategy approach. This method seamlessly integrated the domain-centric heuristic labeling [24] with the algorithmic precision of k-Nearest Neighbors (kNN) [25] imputation. The rationale behind employing heuristic labeling as our initial step stemmed from the nature of our data. Given that some of our labels represented minority classes, relying solely on kNN label imputation could introduce biases towards the majority classes. Heuristic labeling, while dependent on the accuracy of the criteria set for label determination, offers a way to balance the label distribution without such biases. However, recognizing the limitations of heuristic accuracy, we used this method to label only 10-12% of the dataset. This preliminary step aimed to create a more balanced label distribution, setting the stage for the subsequent kNN imputation. The combination of heuristic labeling and kNN imputation allowed us to leverage the strengths of both methods, providing a balance between domain-centric insights and algorithmic precision, and mitigating the potential weaknesses of each method when used alone.

1: Heuristic Labeling: Heuristic labeling leverages domain insights and data patterns to make informed decisions about labels. This method, while not exhaustive, offers a valuable starting point, especially when dealing with imbalanced datasets. Our heuristic algorithm works iteratively. By adjusting criteria based on the quantiles of key features, we assign scores and labels to each instance. The process either converges to a desired label distribution or stops after a predetermined number of iterations.

2: k-Nearest Neighbors (kNN) Imputation: In advancing our research, we transitioned from heuristic labeling to the more nuanced kNN imputation. The underlying principle of the kNN algorithm is rooted in similarity; for any unlabeled data point, it identifies its 'k' closest labeled neighbors and adopts the predominant label among them. The selection of 'k' is crucial, and through meticulous testing, we ascertained the ideal 'k' for distinct data subsets. In our case, we performed an iterative evaluation to determine the optimal 'k' value and found that k=5 yielded the best results. Following the primary imputation, certain labels emerged as fractional. To ensure consistency with our discrete categories—Novice, Intermediate, and Expert—we adjusted these by rounding to the closest whole number. It's worth noting that we used kNN in conjunction with heuristic labeling for the label binning task. While our two-phase approach aimed to provide the most accurate labels possible, it's

worth noting that any label imputation method can introduce errors. For instance, heuristic labeling, while effective in balancing the dataset, might not always capture the nuances of individual developer expertise. Similarly, kNN, though a more common method, can sometimes be influenced by noisy neighbors, leading to potential mislabeling.

After implementing our dual-strategy, the datasets displayed more balanced distributions across various programming languages. For example, widely-used languages like Java showed a more uniform distribution, whereas specialized languages like Go had a higher concentration of expert developers. Notably, after the labeling process, there was a noticeable increase in the number of novice developers. This trend mirrors the real-world scenario where many individuals begin coding, but only a select few achieve expert status.

Upon addressing missing labels with heuristic labeling and kNN imputation, we tackled class imbalance using the Synthetic Minority Over-sampling Technique (SMOTE) [26], [27]. For each dataset, we set target percentages for the 'expert' class, like 15% for C# and 10% for others. We then split the data, ensuring a representative class distribution. Before applying SMOTE, we dynamically set the number of nearest neighbors based on the 'expert' instances in the training data. Using SMOTE, we balanced our datasets, enhancing model accuracy. However, SMOTE can introduce noise, potentially leading to model over-generalization on real-world data.

Phase 3. Model Training:

With our datasets now complete and labeled, we transitioned to the third phase of our research method, we directed our attention towards the pivotal aspect of model training. This phase is instrumental in elucidating the predictive capabilities inherent within our rigorously assembled datasets.

Given our objective of predicting developer expertise levels, we identified it as a multi-class classification problem. We experimented with various machine learning algorithms, including Random Forest (RF), Gradient Boosting (GB), Support Vector Machines (SVM), and Logistic Regression (LR), to assess their performance on our datasets. The rationale behind selecting these classifiers is multi-fold:

Random Forest and Gradient Boosting are both ensemble learning methods known for their high accuracy and ability to handle large datasets with higher dimensionality. They can effectively manage missing values and provide a good indicator of feature importance. Support Vector Machines are renowned for their power in high-dimensional spaces, which is particularly beneficial given the number of features in our dataset. They offer robustness, especially when the number of dimensions exceeds the number of samples. Logistic Regression while is a simpler algorithm, is effective for binary and multi-class classification problems. Its ease of implementation and interpretability make it a valuable tool in our arsenal [15], [17], [18].

In our study, we examined our six distinct datasets, each corresponding to a target language, alongside three datasets from the base article [11]. To understand the

potential influence of incorporating Stack Overflow features, we employed a two-pronged analytical approach.

GitHub-Only Features: Initially, all Stack Overflow features were dropped from our datasets, leaving only the GitHub features. This step was inspired by our base article, which solely considered GitHub features. It is essential to note that the datasets derived from the base article [11] were accessible to the public. To maintain consistency and ensure an equitable comparison, we subjected these datasets to analogous preprocessing steps as those implemented for our own datasets. This approach not only aligned our method but also enhanced the performance of the models on the base article datasets. The results presented in Table 3 for the three base datasets surpass the performance metrics reported in their original paper[11], offering a more robust comparison.

Incorporating Stack Overflow Features: Subsequently, we reintroduced the Stack Overflow features to our six datasets and retrained our models. The performance results post this addition are showcased in Table 4. For each of the above approaches, we embarked on a systematic method. First, we loaded the respective datasets, each tailored to a specific approach, be it GitHub-only or incorporating Stack Overflow features. Then, the data was split into training and testing sets using stratified sampling [28], ensuring that each set accurately represented the overall class distribution. Subsequently, we standardized the features using the StandardScaler [29], ensuring they were on the same scale, which is crucial for effective model training. Next, for each classifier, we defined a set of hyperparameters. To find the optimal parameters, we employed GridSearchCV, an exhaustive search method over the specified parameter values for an estimator. This method pinpointed the parameters of the estimator that yielded the best results on the left-out validation set, ensuring our models were primed for optimal performance.

3.1 Assumptions and Limitations of the proposed method.

In every research endeavor, certain assumptions guide the method, and inherent limitations bound the scope. This section elucidates the foundational assumptions underpinning our proposed method and highlights potential constraints that might influence the interpretation of our findings. Recognizing these factors ensures a nuanced understanding of our research outcomes.

Assumptions:

Data Completeness: We assumed that the data sourced from GitHub and Stack Overflow accurately represents the activities and contributions of developers. However, there might be private repositories or contributions that are not publicly accessible.

Feature Relevance: The features selected for model training were deemed relevant based on prior literature and domain knowledge. It's assumed that these features are indicative of a developer's expertise.

Table 2 Performance metrics of models trained using GitHub for our Target Languages Datasets

Dataset	Model	Acc.*	Kp.*	AUC	P* (Nv.)*	R* (Nv.)	F1 (Nv.)	P (Int.)*	R (Int.)	F1 (Int.)	P (Exp.)*	R (Exp.)	F1 (Exp.)
C#	LR*	0.81	0.66	0.91	0.88	0.92	0.90	0.71	0.86	0.78	1.00	0.23	0.38
C#	RF*	0.86	0.76	0.93	0.86	0.92	0.89	0.85	0.78	0.82	0.85	0.85	0.85
C#	GB*	0.86	0.76	0.94	0.88	0.90	0.89	0.83	0.81	0.82	0.85	0.85	0.85
C#	SVM	0.84	0.73	0.91	0.90	0.94	0.92	0.83	0.78	0.81	0.62	0.62	0.62
Go	LR	0.81	0.65	0.86	0.83	0.95	0.89	0.79	0.82	0.80	0.67	0.06	0.11
Go	RF	0.80	0.65	0.91	0.85	0.90	0.88	0.76	0.83	0.79	0.64	0.22	0.33
Go	GB	0.82	0.69	0.89	0.85	0.94	0.89	0.81	0.81	0.81	0.61	0.34	0.44
Go	SVM	0.83	0.70	0.90	0.87	0.94	0.90	0.81	0.84	0.82	0.64	0.28	0.39
Java	LR	0.78	0.61	0.85	0.87	0.95	0.90	0.71	0.76	0.73	0.40	0.15	0.22
Java	RF	0.81	0.65	0.86	0.86	0.98	0.92	0.75	0.76	0.76	0.50	0.19	0.28
Java	GB	0.80	0.63	0.84	0.85	0.97	0.91	0.74	0.76	0.75	0.50	0.15	0.24
Java	SVM	0.80	0.64	0.87	0.86	0.97	0.91	0.74	0.76	0.75	0.44	0.15	0.23
NodeJs	LR	0.69	0.47	0.84	0.80	0.92	0.85	0.56	0.66	0.61	0.44	0.11	0.17
NodeJs	RF	0.73	0.55	0.85	0.81	0.95	0.88	0.64	0.64	0.64	0.56	0.30	0.39
NodeJs	GB	0.72	0.52	0.85	0.81	0.94	0.87	0.61	0.68	0.64	0.52	0.18	0.27
NodeJs	SVM	0.71	0.51	0.82	0.80	0.94	0.86	0.62	0.68	0.65	0.47	0.14	0.21
Python	LR	0.74	0.54	0.87	0.82	0.95	0.88	0.60	0.72	0.65	0.63	0.19	0.29
Python	RF	0.77	0.61	0.88	0.83	0.96	0.89	0.66	0.73	0.69	0.75	0.33	0.46
Python	GB	0.75	0.57	0.87	0.83	0.96	0.89	0.63	0.67	0.65	0.61	0.30	0.40
Python	SVM	0.74	0.55	0.86	0.84	0.98	0.90	0.60	0.81	0.69	1.00	0.00	0.00
Ruby	LR	0.75	0.44	0.84	0.81	0.92	0.86	0.54	0.53	0.54	0.75	0.17	0.27
Ruby	RF	0.79	0.54	0.84	0.85	0.94	0.90	0.61	0.53	0.57	0.58	0.39	0.47
Ruby	GB	0.76	0.48	0.83	0.83	0.92	0.88	0.53	0.44	0.48	0.67	0.44	0.53
Ruby	SVM	0.80	0.56	0.84	0.86	0.95	0.90	0.62	0.64	0.63	0.80	0.22	0.35

Key: Metrics - Acc. (Accuracy), Kp. (Kappa), AUC (Area Under the Curve), P (Precision), R (Recall), F1 (F1 Score); Expertise Levels - Nv. (Novice), Int. (Intermediate), Exp. (Expert); Models - LR. (Logistic Regression), RF. (Random Forest), GB. (Gradient Boosting).

Table 3 Performance metrics of models trained using GitHub for Datasets from the Base Article [11]

Dataset	Model	Acc.	Kp.	AUC	P (Nv.)	R (Nv.)	F1 (Nv.)	P (Int.)	R (Int.)	F1 (Int.)	P (Exp.)	R (Exp.)	F1 (Exp.)
Socket.io	RF	0.22	0.20	0.47	0.33	0.43	0.38	0.20	0.14	0.17	0.00	0.00	0.00
Socket.io	SVM	0.33	0.21	0.51	0.57	0.57	0.57	0.20	0.14	0.17	0.17	0.25	0.20
React	RF	0.55	0.24	0.66	0.64	0.72	0.68	0.11	0.05	0.06	0.57	0.67	0.61
React	SVM	0.52	0.16	0.73	0.55	0.55	0.55	0.00	0.00	0.00	0.54	0.73	0.62
MongoDB	RF	0.57	0.36	0.71	0.50	0.75	0.60	0.67	0.40	0.50	0.60	0.60	0.60
MongoDB	SVM	0.50	0.22	0.63	0.80	0.00	0.00	0.40	0.40	0.40	0.56	1.00	0.71

Table 4 Performance metrics of models trained using GitHub and Stack Overflow features

Dataset	Model	Acc.	Kp.	AUC	P (Nv.)	R (Nv.)	F1 (Nv.)	P (Int.)	R (Int.)	F1 (Int.)	P (Exp.)	R (Exp.)	F1 (Exp.)
C#	GB	0.85	0.74	0.96	0.86	0.92	0.89	0.82	0.76	0.79	0.85	0.85	0.85
C#	LR	0.86	0.77	0.92	0.93	0.90	0.91	0.83	0.78	0.81	0.71	0.92	0.80
C#	RF	0.88	0.80	0.95	0.90	0.92	0.91	0.86	0.84	0.85	0.85	0.85	0.85
C#	SVM	0.87	0.78	0.97	0.94	0.92	0.93	0.85	0.78	0.82	0.71	0.92	0.80
Go	GB	0.86	0.77	0.94	0.90	0.90	0.91	0.85	0.86	0.84	0.78	0.68	0.72
Go	LR	0.86	0.75	0.92	0.87	0.95	0.91	0.86	0.81	0.83	0.80	0.62	0.70
Go	RF	0.87	0.78	0.95	0.91	0.93	0.92	0.85	0.86	0.85	0.78	0.66	0.71
Go	SVM	0.90	0.82	0.95	0.94	0.92	0.93	0.87	0.92	0.89	0.79	0.69	0.73
Java	GB	0.90	0.83	0.98	0.93	0.95	0.94	0.88	0.85	0.87	0.85	0.85	0.85
Java	LR	0.90	0.84	0.98	0.97	0.94	0.95	0.85	0.89	0.87	0.78	0.81	0.79
Java	RF	0.92	0.87	0.98	0.94	0.96	0.95	0.93	0.86	0.89	0.86	0.92	0.89
Java	SVM	0.91	0.84	0.98	0.95	0.95	0.95	0.89	0.86	0.87	0.82	0.88	0.85
NodeJs	GB	0.87	0.82	0.96	0.91	0.90	0.90	0.83	0.86	0.85	0.95	0.89	0.92
NodeJs	LR	0.89	0.82	0.97	0.90	0.91	0.91	0.85	0.85	0.85	0.94	0.91	0.92
NodeJs	RF	0.89	0.82	0.97	0.90	0.91	0.90	0.84	0.85	0.84	0.97	0.89	0.93
NodeJs	SVM	0.92	0.87	0.97	0.92	0.96	0.94	0.90	0.88	0.89	0.95	0.89	0.92
Python	GB	0.91	0.85	0.97	0.92	0.95	0.93	0.87	0.85	0.86	0.95	0.89	0.92
Python	LR	0.92	0.87	0.98	0.93	0.97	0.95	0.90	0.84	0.87	0.94	0.92	0.93
Python	RF	0.92	0.87	0.97	0.92	0.96	0.94	0.87	0.86	0.87	0.98	0.89	0.93
Python	SVM	0.93	0.88	0.97	0.94	0.96	0.95	0.89	0.88	0.88	0.95	0.92	0.94
Ruby	GB	0.87	0.73	0.95	0.92	0.94	0.93	0.74	0.69	0.71	0.82	0.78	0.80
Ruby	LR	0.88	0.74	0.95	0.92	0.94	0.93	0.76	0.69	0.72	0.83	0.83	0.83
Ruby	RF	0.88	0.76	0.95	0.92	0.96	0.94	0.80	0.67	0.73	0.79	0.83	0.81
Ruby	SVM	0.89	0.79	0.96	0.94	0.94	0.94	0.76	0.81	0.78	0.88	0.78	0.82

Model Applicability: We assumed that the machine learning models chosen for this study are suitable for the type of data and the problem at hand. The performance of these models might vary with different datasets or contexts.

Developer Overlap between NodeJs and MongoDB: In the base article [11], MongoDB is represented as NODE-MONGODB, the official NodeJs driver for the MongoDB database server. While there's likely a significant overlap between MongoDB and NodeJs developers, it's crucial to acknowledge that not all MongoDB developers may be proficient in NodeJs, and vice versa. For the purpose of a more direct comparison between our datasets and those of the base article, we operate under the assumption that developers associated with MongoDB also have expertise in NodeJs. **Self-Assessment Reliability:** We operate under the assumption that the self-assessments provided by contributors are accurate and offer a reliable representation of their expertise levels.

Limitations:

Data Bias: Relying solely on GitHub and Stack Overflow might introduce a bias, as developers might be active on other platforms or might not be active online at all. This could lead to an incomplete representation of a developer's true expertise. GitHub and Stack Overflow are widely used platforms in the developer community, but they may not fully represent the broader developer community. For example, developers who primarily use other platforms or who do not participate in online communities may not be well-represented. Furthermore, the behaviors and activities on these platforms may not fully reflect a developer's offline activities or their activities on other platforms. One potential way to mitigate this bias, which we discuss in our conclusion, is to integrate data from a wider range of developer platforms in future research.

External Validity: While our models showed promising results in the context of serverless functions, their applicability to other technological domains needs further validation.

Feature Limitations: While we integrated features from GitHub and Stack Overflow, other platforms like LinkedIn or TopCoder might offer additional insights that could enhance the model's predictive power. We attempted to use the public LinkedIn REST API⁷ to complement our data collection with user activity data from LinkedIn. However, to the best of our knowledge, the LinkedIn public API is no longer accessible. Furthermore, while we considered implementing a web crawler to gather data from LinkedIn, we decided against it. Although it's not illegal to collect publicly accessible data from the web, such an approach would be against LinkedIn's terms and conditions.

Model Constraints: Every machine learning model has its inherent limitations. For instance, linear models might not capture non-linear relationships well, and tree-based models might overfit on sparse data [30].

4. Evaluation

In response to RQ1, which inquires about the most effective machine learning algorithms for evaluating developer expertise in serverless functions based on the extracted features, we conducted a rigorous evaluation of our trained models. Once the models were trained with the best parameters; they underwent rigorous evaluation on the test set. We computed a range of metrics, including accuracy, kappa score, and AUC, to evaluate their performance. Additionally, we calculated precision, recall, and F1-score for each class - Novice, Intermediate, and Expert.

This detailed evaluation ensured we captured the nuances of each model's predictive capabilities, providing a detailed understanding of their strengths and limitations. By adhering to this method for both approaches, we ensured a consistent evaluation, allowing for a fair comparison between the utility of GitHub-only features and the combined GitHub and Stack Overflow features. The results for both approaches can be found in Table 2, Table 3, and Table 4. Following the presentation of results, we delve into a deeper analysis of the datasets. Initially, we focus on datasets using only GitHub features, subsequently, we discuss the datasets enhanced with Stack Overflow features and the resultant impact on model outcomes. The evaluation concludes with a feature importance analysis using the SHAP method, offering insights into the pivotal role of each feature. In this structured discussion, we aim to present a thorough understanding of our findings and their broader implications.

Part 1. Discussion Datasets with only GitHub Features

In the evaluation of target languages' datasets presented in Table 2, several trends and patterns emerge. RF model consistently exhibits robust performance across multiple datasets, often securing the lead in accuracy, Kappa, and F1 scores. This superior performance can be attributed to RF's ensemble nature, which amalgamates the results of numerous decision trees, offering a more generalized and resilient model. On the other hand, the SVM model also demonstrates commendable performance, particularly in precision. However, certain anomalies, such as in the Python dataset for the Exp. category, reveal that while SVM can predict with high confidence, it might occasionally overlook specific classes, resulting in a diminished recall.

⁷ <https://developer.linkedin.com/product-catalog>

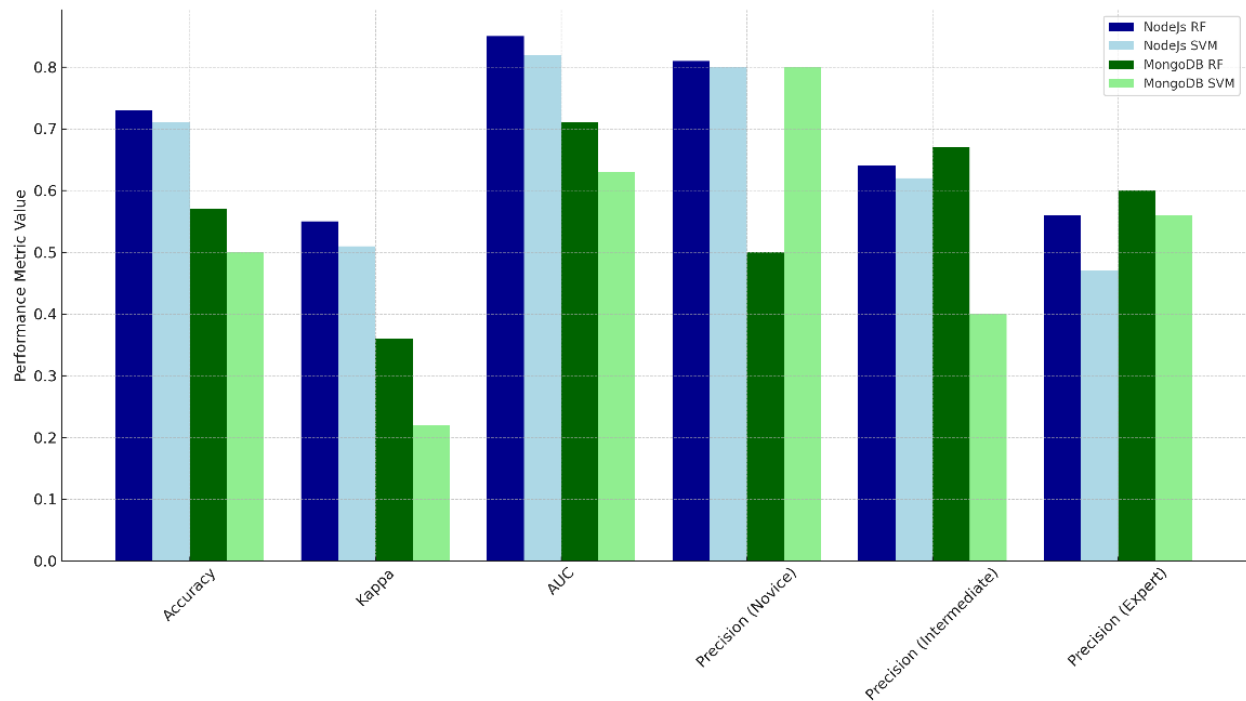


Figure 5. Comparison of performance metrics for NodeJs and MongoDB datasets

The GB model, another contender, frequently matches the performance of RF across datasets. GB's strength lies in its boosting algorithm, which zeroes in on challenging instances, potentially enhancing its performance on intricate datasets. Conversely, the LR model, inherently linear, occasionally falls short compared to tree-based or SVM models, especially when faced with datasets characterized by non-linear decision boundaries. Table 3, focusing on base work [11] results, highlights the challenges encountered in model predictions. The Socket.io dataset, for instance, presents notably low metrics for both RF and SVM models.

In Figure 5, we compared the performance of the Random Forest and SVM models for NodeJs and MongoDB datasets. The motivation behind selecting the NodeJs and MongoDB datasets for comparison is rooted in the anticipated overlap between their developer communities. Based on the premise that MongoDB, denoted as NODE-MONGODB — the official NodeJs driver for the MongoDB database server — would have a substantial confluence of developers skilled in both domains, these datasets were chosen for a deeper analysis. Upon analyzing the performance metrics, a distinct pattern becomes evident: models trained on the NodeJs dataset tend to surpass those trained on the MongoDB dataset across the majority of metrics. Moreover, within the NodeJs dataset, the RF model consistently outshines the SVM in parameters such as accuracy, kappa, and AUC. Both models demonstrate laudable precision in recognizing novice developers, underscoring a robust ability to accurately discern beginner-level expertise. Nonetheless, a discernible drop in precision is observed as we transition to higher levels of expertise, particularly in the SVM model for NodeJs. The performance metrics for MongoDB were influenced by multiple factors. A primary consideration is the label retrieval method, which predominantly hinged on self-assessment. While our datasets also utilized self-assessment, it is noteworthy that around 9% of our data was labeled based on genuine

expertise levels. Our hybrid labeling approach, encompassing heuristic techniques and kNN label imputation, seemed to yield more consistent outcomes. This approach counteracts the inherent discrepancies often associated with sole reliance on self-assessments. The potential for developers to inaccurately evaluate their own skills introduces the risk of misclassification. Such disparities might be accentuated for MongoDB, suggesting potential variances between self-declared and actual proficiency. In addition, the constraints posed by a limited number of training instances cannot be overlooked. Despite our efforts in employing techniques like SMOTE to address dataset imbalance, the foundational issue of a restricted data sample might induce overfitting, consequently diminishing the model's generalization capabilities. For consistency, the same preprocessing steps were applied across all nine datasets, and it's worth mentioning that the performance results reported for the base work [11] witnessed an enhancement due to our preprocessing techniques.

Moreover, while accuracy is a widely used metric, its limitations become pronounced, especially in datasets with class imbalances. Other metrics like Kappa, AUC, precision, recall, and F1 score provide a more comprehensive view, capturing the model's performance nuances across various classes. In conclusion, while RF often stands out in performance metrics, it's crucial to consider each dataset's unique characteristics when choosing a model. The occasional unexpected or below-average metric emphasizes the importance of a holistic evaluation using a diverse set of metrics, ensuring a well-rounded understanding of model performance.

Part 2. Discussion on the Enhanced Datasets with Stack Overflow Features

Upon integrating Stack Overflow features into our datasets, as shown in Table 4, we observed a marked improvement in the performance metrics of the models.

This enhancement underscores the significance of feature engineering and the potential of external data sources in boosting model performance. For the C# dataset, the RF model stands out with the highest accuracy of 0.88 and a Kappa score of 0.80. SVM, with an AUC of 0.97, indicates its capability to distinguish between classes effectively. Interestingly, LR showcases a perfect recall of 0.92 for the Exp. category, emphasizing its strength in identifying true positives for this class. In the Go dataset, SVM takes the lead with an accuracy of 0.90 and a Kappa score of 0.82. Its performance in the Int. category, with a recall of 0.92 and an F1 score of 0.89, is noteworthy, suggesting its proficiency in classifying intermediate instances. For the Java dataset, RF demonstrates robust performance with an accuracy of 0.92 and a Kappa score of 0.87. Its precision of 0.93 for the Int. category is commendable. However, GB's AUC of 0.98 is the highest, indicating its superior ability to differentiate between the classes.

In the NodeJs dataset, SVM emerges as the top performer with an accuracy of 0.92 and a Kappa score of 0.87. Its F1 scores across all categories are consistently high, reflecting its balanced precision and recall.

For the Python dataset, SVM shines with the highest accuracy of 0.93 and a Kappa score of 0.88. Its performance in the Exp. category, with an F1 score of 0.94, is particularly impressive. LR's recall of 0.97 for the Nv. category is the highest, indicating its strength in identifying true positives for novice instances.

Lastly, in the Ruby dataset, SVM leads with an accuracy of 0.89 and a Kappa score of 0.79. Its performance in the Nv. category, with an F1 score of 0.94, is outstanding, suggesting its proficiency in classifying novice instances. The addition of Stack Overflow features has evidently bolstered the models' performance across the datasets. The enriched datasets provide a more detailed view of each instance, allowing the models to capture intricate patterns and relationships. It's worth noting that while we couldn't expand the base datasets to include Stack Overflow features due to privacy concerns related to email hashing, the similarity in nature between NODE-MONGODB (NodeJs driver for MongoDB) and NodeJs offers a reasonable point of comparison. Given the overlaps between MongoDB and NodeJs developers, this similarity can serve as a benchmark to assess the impact of the added features. We hypothesize that if we were able to expand the MongoDB dataset similarly, we would likely observe comparable improvements as seen with the NodeJs dataset.

In conclusion, the integration of external features, such as those from Stack Overflow, can substantially enhance model performance. The importance of feature engineering is evident, and the potential of utilizing external data sources in machine learning tasks is undeniable. Referring to the results depicted in Figure 6, it's clear that leveraging additional features leads to noticeable accuracy boosts across various programming languages. For example, the SVM model accuracy for the 'Go' language saw an increase from 83% using only GitHub features to 92% when integrated with Stack Overflow features, and the continuous pursuit of integrating relevant external data to achieve optimal model performance. We observed similar significant

improvements for languages like 'Java', 'NodeJs', and 'Python'. However, for languages like 'C#' and 'Ruby', the enhancements were more modest. These findings underscore the value of broadening the feature space.

Part 3. Feature Importance Analysis

In response to RQ2, which seeks to identify the most indicative features or metrics of a developer's expertise in serverless functions, we employed the SHAP (SHapley Additive exPlanations) method. SHAP values offer a unified measure of feature importance, assigning each feature an importance value for a specific prediction. This method excels in providing both global interpretability—indicating the importance of each feature across the entire dataset—and local interpretability,

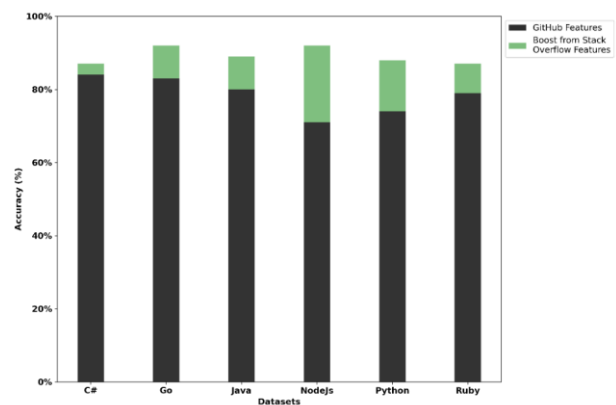


Figure 6: Comparison of SVM model accuracies using only GitHub features versus the improvement achieved by adding Stack Overflow features for target languages

which explains individual predictions. As illustrated in Figure 7 (a SHAP summary plot), the SHAP analysis offers a comprehensive perspective on the significance of each feature [31].

Our analysis reveals the pivotal role of various features in predicting a developer's proficiency. The 'avg_days_commits_import_library' feature, denoting the average number of days between commits that import libraries, stands out as paramount. This metric suggests that a developer integrating new libraries frequently might be inclined towards proactive experimentation and learning. Similarly, the 'commits_import_library' feature, which reflects the number of such commits, can hint at the intricacy of applications a developer crafts. The 'time_of_activity' metric, capturing the span of a developer's activity, indicates sustained technological interest, which is crucial for continuous learning and expertise development.

Yet, insights aren't solely derived from GitHub activity. Additionally, 'commits_client_files', which counts commits altering at least one client file, sheds light on a developer's active involvement and contributions in a project's primary language. This could be interpreted as a sign of a developer's commitment to a project and their expertise in the project's main language. Integration of Stack Overflow features enriches our understanding. For instance, 'upvotes', ranking as the third most influential

feature, accentuates the community's acknowledgment of a developer's input. A high 'tag_score' signifies domain-specific expertise. Metrics like 'average_score_per_answer' and 'first_answers' encapsulate both the caliber and the regularity of a user's contributions—the former indicating consistent answer quality and the latter reflecting active community participation. 'Accepted_answers' further vouch for the quality and pertinence of a developer's knowledge dissemination. While GitHub metrics provide a window into a developer's coding habits, Stack Overflow metrics delve into their community engagement and problem-solving acumen. Collectively, insights from both platforms paint a holistic picture of a developer's expertise in a specific technology.

4.1 DISCUSSION

Our results underscore the potential of integrating data from multiple platforms to enhance the precision of evaluating developers' expertise, especially those involved in serverless functions. The incorporation of Stack Overflow features alongside GitHub data, particularly in the context of serverless function development, has shown promising improvements in our model's performance.

In the industrial landscape, our model suggests a nuanced approach to recruitment, talent acquisition, and

team optimization, especially for roles centered around serverless functions. By amalgamating coding practices and community engagement metrics from platforms like GitHub and Stack Overflow, there's potential for a more in-depth insight into a developer's skills and contributions in the serverless domain. Such insights could refine the hiring process, potentially leading to more targeted training and role assignments. Although our study's focal point is the serverless function domain, the method might be adaptable to other technological areas. For new instances, our model provides a framework for assessing individual developer profiles, potentially offering predictions on their expertise levels based on the features we've identified.

In the academic realm, our model could serve as a potential reference for research, curriculum development, and student assessment in serverless function development and beyond. It might offer insights for studies exploring developer behavior and proficiency, especially when integrating data from multiple sources. By evaluating students' real-world coding activities, there's an opportunity for educators to offer feedback that encompasses both theoretical knowledge and practical application, promoting a balanced learning experience.

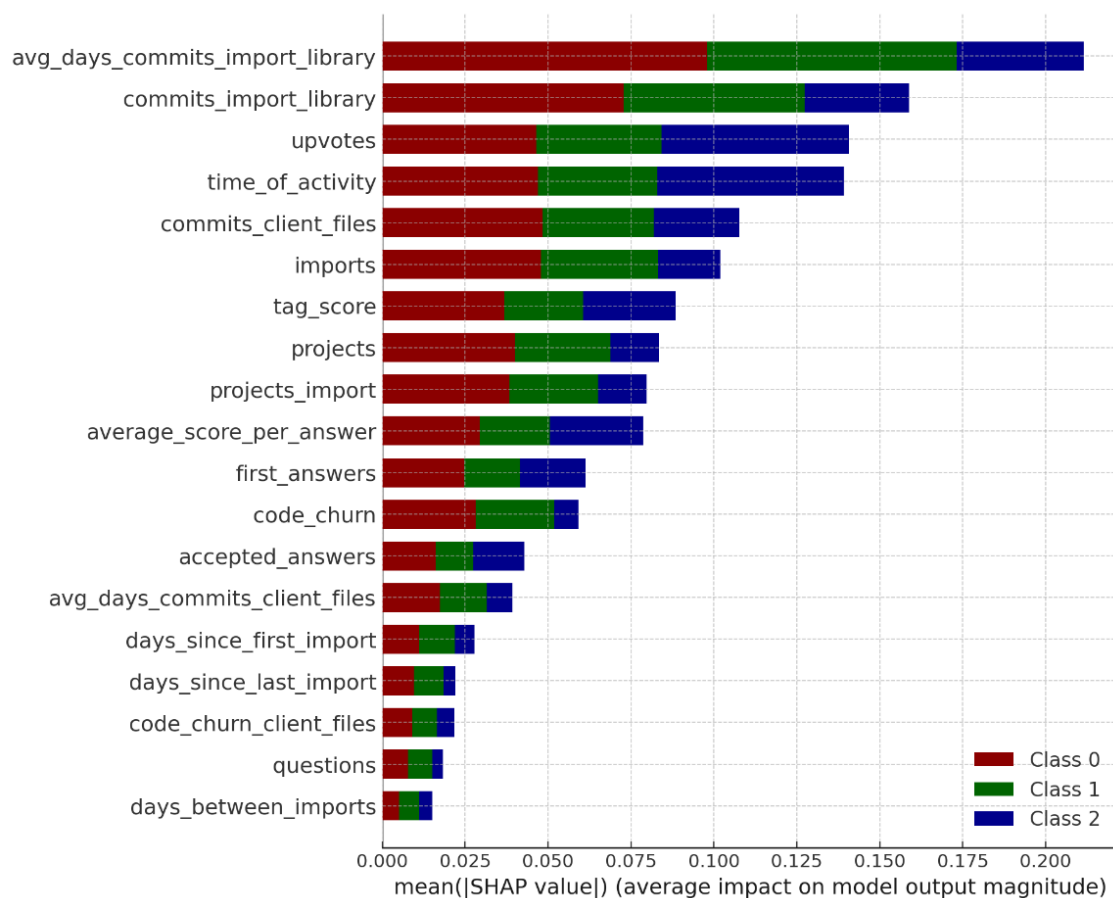


Figure 7: SHAP Summary Plot

5. Threats to Validity

Research, especially in the domain of empirical studies, is often subject to various threats that might affect the generalizability and validity of the results. In this section, we discuss potential threats to the validity of our study and the measures we've taken to mitigate them.

Model Bias: While we experimented with multiple machine learning algorithms, each model comes with its inherent biases. For instance, tree-based models like Random Forest might overfit on certain datasets, while linear models like Logistic Regression might not capture non-linear relationships effectively.

Hyperparameter Tuning: Although we employed GridSearchCV for exhaustive hyperparameter tuning, there's always a possibility that a different combination might yield slightly better results.

Feature Selection: The inclusion or exclusion of features can impact model performance. Our hybrid feature selection method combined heuristic labeling with kNN label imputation. However, heuristic labeling, despite aiding dataset balance, may not always reflect the intricacies of developer expertise. Likewise, while kNN is widely used, it can be susceptible to mislabeling due to noisy neighbors.

Dataset Specificity: Our study is based on specific datasets tailored to certain programming languages. The findings might not be directly generalizable to other languages or platforms.

Data Absence from Other Platforms: The lack of data from platforms like LinkedIn and TopCoder may limit our model's comprehensiveness. Missing insights from these platforms could challenge the external validity of our findings, potentially overlooking essential indicators of developer expertise.

Labeling and Classification: The classification of developers into categories like Novice, Intermediate, and Expert is based on certain metrics and might not capture the complete essence of a developer's expertise.

Feature Interpretation: While we employed the SHAP method for feature importance analysis, the interpretation of the importance of certain features might vary among experts.

Self-Assessment Accuracy: While we operated under the assumption that the self-assessments provided by contributors were accurate reflections of their expertise levels, there's an inherent risk associated with relying on subjective evaluations. Contributors might have overestimated or underestimated their skills due to factors

like overconfidence, modesty, or a lack of clear understanding of the assessment criteria. This potential discrepancy between perceived and actual expertise could influence the validity of our findings, especially if these self-assessments were used as ground truth or reference points in our analysis.

6. Conclusion and Future Directions

In this research, we ventured into the domain of predicting developer expertise specifically within the realm of serverless functions, using features extracted from GitHub and Stack Overflow. Our findings underscored the value of multi-platform data integration in providing an in-depth understanding of developer expertise.

In our research, we explored two key areas. The first area of exploration (RQ1) revolved around identifying the most effective machine learning algorithms for evaluating developer expertise. Our journey led us to the Random Forest (RF) model, which consistently demonstrated robust performance across multiple datasets. We also observed commendable performance from the Support Vector Machine (SVM) model, particularly in terms of precision.

The second area of exploration (RQ2) focused on uncovering the features or metrics that best indicate a developer's expertise in serverless functions. Our exploration revealed that the top 5 indicators were 'avg_days_commits_import_library', 'commits_import_library', 'upvotes', 'time_of_activity', and 'commits_client_files'. We used the SHAP method for feature importance analysis to arrive at these insights.

Our research adds to the field by predicting developer expertise in serverless functions, an area not widely studied before. We used 22 features from GitHub and Stack Overflow, which is more than what's typically used in this domain. This large set of features gives us a detailed look at developer activities and expertise. While there are other studies [4], [5] that also use data from multiple platforms, our study stands out because we use both GitHub and Stack Overflow data and a larger set of features. Our results agree with other studies that find it useful to combine insights from multiple platforms. But our research goes one step further by showing how this approach works well for serverless functions.

In addition, one of the tangible outputs of our research is the creation and public release of six language-specific datasets, representing our target languages. By making these datasets publicly available⁸, we not only aim to contribute to the academic community but also hope to foster further research in this area.

Our current investigation has also highlighted the potential benefits of integrating insights from platforms such as Stack Overflow. Such an integrative approach, which merges data from varied sources, can offer richer insights into the multifaceted nature of developer expertise, especially in the context of serverless functions.

⁸ <https://github.com/aref98/Evaluating-Developer-Expertise-in-Serverless-Functions-by-Mining-Activities-from-Multiple-Platforms>

As we move forward, there are multiple avenues we can explore to build upon our current findings. One potential direction is to widen our data collection to encompass a more extensive range of repositories, offering a deeper dive into developer activities. Another promising avenue is to investigate other metrics of developer expertise, such as peer reviews or code quality assessments, which might yield a more nuanced understanding. Finally, considering the vast ecosystem of developer platforms, integrating data from platforms like GitLab, Bitbucket, and LinkedIn, as well as competitive coding platforms like TopCoder, can provide a more rounded view of developer behavior and skills. This could be the next step in further refining and expanding our understanding of expertise in serverless function development.

7. References

- [1] S. Kourtzanidis, A. Chatzigeorgiou, and A. Ampatzoglou, "RepoSkillMiner: Identifying software expertise from GitHub repositories using Natural Language Processing," *Proc. - 35th IEEE/ACM Int. Conf. Autom. Softw. Eng. ASE, 2020*, pp. 1353–1357.
- [2] X. Song, J. Yan, Y. Huang, H. Sun, and H. Zhang, "A Collaboration-Aware Approach to Profiling Developer Expertise with Cross-Community Data," *IEEE Int. Conf. Softw. Qual. Reliab. Secur. QRS*, Guangzhou, China, 2022, pp. 344–355.
- [3] E. Constantinou and G. M. Kapitsaki, "Developers expertise and roles on software technologies," *Proc. - Asia-Pacific Softw. Eng. Conf. APSEC*, Hamilton, New Zealand, 2016, pp. 365–368.
- [4] Y. Tian, W. Ng, J. Cao, and S. McIntosh. (2019, Nov.). Geek talents: Who are the top experts on GitHub and stack overflow?. *Comput. Mater & Contin.* [Online]. 61(2), pp. 465–479. Available: 10.32604/cmc.2019.07818
- [5] S. L. Vadlamani and O. Baysal, "Studying Software Developer Expertise and Contributions in Stack Overflow and GitHub," *IEEE International Conference on Software Maintenance and Evolution (ICSME)*, Adelaide, SA, Australia, 2020, pp. 312–323.
- [6] P. Castro, V. Ishakian, V. Muthusamy, and A. Slominski. (2019, Nov.). The rise of serverless computing. *Communications of the ACM.* [Online]. 62(12), pp. 44–54. Available: 10.1145/3368454
- [7] M. Shahrad et al., "Serverless in the wild: Characterizing and optimizing the serverless workload at a large cloud provider," *Proc. 2020 USENIX Annu. Tech. Conf. ATC*, 2020, pp. 205–218.
- [8] A. Mujezinovic and V. Ljubovic, "Serverless architecture for workflow scheduling with unconstrained execution environment," *42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija, Croatia, 2019, pp. 242–246.
- [9] R. Cordingley et al., "Implications of Programming Language Selection for Serverless Data Processing Pipelines," *IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCCom/CyberSciTech)*, Calgary, AB, Canada, 2020, pp. 704–711.
- [10] J. Oliveira, M. Vigiato, and E. Figueiredo, "How well do you know this library? Mining experts from source code analysis," *ACM Int. Conf. Proceeding Ser.*, 2019, pp. 49–58.
- [11] J. E. Montandon, L. Lourdes Silva, and M. T. Valente, "Identifying experts in software libraries and frameworks among GitHub Users," *IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*, Montreal, QC, Canada, 2019, pp. 276–287.
- [12] B. Vasilescu, V. Filkov, and A. Serebrenik, "StackOverflow and GitHub: Associations between software development and crowdsourced knowledge," *International Conference on Social Computing*, Alexandria, VA, USA, 2013, pp. 188–195.
- [13] A. Santos, M. Souza, J. Oliveira, and E. Figueiredo, "Mining software repositories to identify library experts," *ACM Int. Conf. Proceeding Ser.*, 2018, pp. 83–91.
- [14] G. J. Greene and B. Fischer, "CVExplorer: Identifying candidate developers by mining and exploring their open source contributions," *Proceedings of the 31st IEEE/ACM international conference on automated software engineering*, 2016, pp. 804–809.
- [15] X. T. Trinh, "Online learning of multi-class Support Vector Machines," *Master dissertation*, no. 12 061, 2012.
- [16] Christopher J.C. Burges. (1998, Jun.). A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery.* [Online]. 2(2), pp. 121–167. Available: 10.1023/A:1009715923555
- [17] G. Biau and E. Scornet. (2016, Apr.). A random forest guided tour. *Test.* [Online]. 25(2), pp. 197–227. Available: 10.1007/s11749-016-0481-7
- [18] A. Natekin and A. Knoll. (2013, Dec.). Gradient boosting machines, a tutorial. *Front. Neurobot.* [Online]. 7, p. 21. Available: 10.3389/fnbot.2013.00021
- [19] A. Schneider, G. Hommel, and M. Blettner. (2010, Nov.). Lineare regressionsanalyse - Teil 14 der serie zur bewertung wissenschaftlicher publikationen. *Deutsches Ärzteblatt International.* [Online]. 107(44), pp. 776–782. Available: 10.3238/arztebl.2010.0776
- [20] Ö. Senger. (2013, Aug.). Impact of skewness on statistical power. *Modern Applied Science.* [Online]. 7(8), pp. 49–56. Available: 10.5539/mas.v7n8p49
- [21] N. J. Gogtay and U. M. Thatte. (2017, Mar.). Principles of correlation analysis. *Journal of the Association of Physicians of India.* [Online]. 65(3), pp. 78–81.
- [22] S. Chulani, B. Boehm, and B. Steece. (1999, Jul.). Bayesian Analysis of Empirical Software Engineering Cost Models. *IEEE Transactions on Software Engineering.* [Online]. 25(4), pp. 41–51. Available: 10.1109/32.799958
- [23] J. Li et al. and H. Liu. (2017, Dec.). Feature selection: A data perspective. *ACM computing surveys (CSUR).* [Online]. 50(6), pp. 1–45. Available: 10.1145/3136625
- [24] B. Boecking, W. Neiswanger, E. P. Xing, and A. Dubrawski, "Interactive Weak Supervision: Learning Useful Heuristics for Data Labeling," *ICLR 2021 - 9th Int. Conf. Learn. Represent.*, 2021, pp. 1–27.
- [25] S. Zhang, (2012, Nov.). Nearest neighbor selection for iteratively kNN imputation. *Journal of Systems and Software.* [Online]. 85(11), pp. 2541–2552. Available: 10.1016/j.jss.2012.05.073
- [26] M. Tan, L. Tan, S. Dara, and C. Mayeux, "Online Defect Prediction for Imbalanced Data," *IEEE/ACM 37th IEEE International Conference on Software Engineering*, Florence, Italy, 2015, pp. 99–108.
- [27] L. Li, T. F. Bissyandé, D. Octeau, and J. Klein, "Reflection-aware static analysis of android apps," *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, 2016, pp. 756–761.
- [28] A. S. Singh, M. B. Masuku, and Department. (2014, Nov.). Sampling techniques & determination of sample size in applied statistics research. *International Journal of economics, commerce and management.* [Online]. 2(11),

- pp. 32–33. Available: ijecm.co.uk
- [29] L. Buitinck et al. and R. Layton, “API design for machine learning software: experiences from the scikit-learn project,” European Conference on Machine Learning and Principles and Practices of Knowledge Discovery in Databases, 2013, pp. 1–15.
- [30] R. M. Dawes, “The robust beauty of improper linear models in decision making,” Rationality and Social Responsibility, 2008, pp. 321-344.
- [31] S. M. Lundberg and S. I. Lee, “A unified approach to interpreting model predictions,” Adv. Neural Inf. Process. Syst., 2017, pp. 4766–4775.
- [32] A. Sayers, Y. Ben-Shlomo, A. W. Blom, and F. Steele. (2016, Jun.). Probabilistic record linkage. International Journal of Epidemiology. [Online]. 45(3), pp. 954-964. Available: [10.1093/ije/dyv322](https://doi.org/10.1093/ije/dyv322)
-



Ferdowsi
University of
Mashhad

Journal of Computer and Knowledge Engineering

<https://cke.um.ac.ir>



Information and
Communication
Technology Association of
Iran

Description-based Post-hoc Explanation for Twitter List Recommendations*

Research Article

Havva Alizadeh Noughabi¹; Behshid Behkamal² ; Mohsen Kahani³

DOI: [10.22067/cke.2024.85185.1107](https://doi.org/10.22067/cke.2024.85185.1107)

Abstract Twitter List recommender systems can generate highly accurate recommendations, but since they employ heterogeneous information of users and Lists and apply sophisticated prediction models, they may not provide easy understandable intrinsic explanations. To address this limitation, Twitter List descriptions can play a critical role in providing post-hoc explanations that help users make informed decisions. In this paper, we present a model to provide relevant and informative explanations for recommended Twitter Lists by automatically generating descriptions for them. The model selects the most informative tweets from a List as its description to inform users more with the recommended List that positively contributes to the user experience. More specifically, the explanation model incorporates three categories of features: *content relevance* features, *tweet-specific* features and *publisher's authority* features that are used in a learning to rank model to rank the List's tweets in terms of their informativeness. Experimental results on a Twitter dataset validate the effectiveness of our proposed model in generating useful explanations for recommended Twitter Lists.

Keywords: *Index Terms-- Explainable recommender systems, Post-hoc explanation, Description generation, Twitter Lists.*

1. Introduction

With the substantial increase in user-generated content on social media, several platforms assist users in organizing related information into a single bin. For instance, Twitter introduced *Lists* as a solution to tackle the issue of information overload [1]. A Twitter List is a group of accounts that anyone with an interest in the topics covered by the List can subscribe to for free. While List recommender systems have been highly effective in using of different user and List features as well as advanced hybrid models to improve their performance [2], [3], their lack of explainability remains a significant challenge.

Nowadays, the user experience with social media platforms, which includes factors such as trust,

understandability, and satisfaction, is increasingly influenced by the availability of explainability in social recommender systems [4], [5]. This has motivated us to provide informative explanations for the recommended Twitter Lists. Users who were provided with explanations for recommended Lists were found to be more inclined to engage with them and to perceive the recommendations as relevant and useful.

On the other hand, the development of methods to provide post-hoc explanations, which are generated after recommendations have been made, has recently attracted a lot of interest in the field of recommender systems [6]–[8]. Post-hoc explanations can make it easier for users to make informed decisions about the recommendations by providing them with detailed information about the recommended item. This is especially useful for users who may not be familiar with the technical aspects of the recommender system and may find it difficult to understand intrinsic explanations, which can be too complex or technical for them. Our main goal is to provide a post-hoc explanation for the recommended Twitter List, which not only helps to maintain the predictive accuracy of the current complex recommender systems but also ensures that users can gain more information and insights into the recommended Lists.

Description of Twitter Lists can play an important role to provide post-hoc explanations for the recommended Lists. However, only a few popular Lists have a description written by the List owner on Twitter. For example, the List named *NTD* does not have any informative description on Twitter and it is quite hard for users to understand the main topics of the List or guess the content of it. For such Lists, the user needs to manually check the tweets of the List and read some of its recent tweets to figure out the topics of it which is tedious and highly time-consuming. Automatically generating an accurate and informative description about Lists on Twitter not only helps users make an informed decision but also improves the likelihood of subscribing to Lists on

* Manuscript received: 2023 November 4, Revised, 2024 May 19, Accepted, 2024 June 22.

¹ PhD, Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran.

² Corresponding author. Associate Professor, Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran. **Email:** behkamal@um.ac.ir.

³ Professor, Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran.

Twitter.

Recently, there has been increasing interest in automatic item description generation [9]–[11]. Some studies have utilized statistical frameworks, such as [12], that incorporate statistical methods with templates for generating product descriptions. Other research, like [13], have proposed to generate summarization of item reviews by applying a template based Natural Language Generation (NLG) framework. To overcome the need for handcrafted templates, researchers have turned to deep learning-based models and presented various conditions to the generation model [14], thereby addressing the limitations of traditional approaches. In our paper, to generate a Twitter List's description, our main idea is to select the most informative tweets of a List as its description to inform users more with the recommended List that improves their overall experience.

Specifically, compared to existing explainable models with various goals [14], [15], our work focuses on the recommended Twitter List to present an informative explanation with the aim of helping users to know more about the content of the recommendation. Therefore, providing a set of K tweets from the recommended List can give pertinent information to users, although despite noisy tweets, considering the quality of selected tweets is so important to obtain a relevant and informative tweet-level explanation. In this paper, to catch the high quality description for Twitter Lists, we use three feature categories to consider various perspectives: *Content Relevance Features*, *Tweet Specific Features* and *Publisher's Authority Features*. Then highly-useful tweets are obtained which provide explanations to help users know more about the List.

To make our idea more clear, we give an example. Consider a recommended Twitter List named 'Web Dev' that has many tweets within it that are posted by its members. Some of which are below and let's assume that one of these tweets is going to be presented as an explanation with the aim of helping users to know more about the content of the List.

A) "Roadmap for web development: 1-HTML 2-CSS 3-CSS frameworks 4-JS 5-DOM 6-Git and GitHub 7-React 8-Node.js 9-API 10-Database 11-Web3.js 12-Solidity."

B) "JavaScript is Awesome!"

C) "It's 5:30 and the sun is still up."

D) "Keep wearing masks if you want to survive. Dropping all the mitigation measures will bring another wave."

It is evident that tweets C and D are not associated with the primary topic of the 'Web Dev' List. For example, members posted tweet D due to trending topics of it (Covid19). Tweets A and B are related and can be considered as potential explanations to provide information about the List. Given that tweet A contains more detailed information that helps to clarify the recommended List, it is considered to be more informative than tweet B.

The main contributions of the paper are summarized as follows.

- We propose a method to give a post-hoc explanation

for the recommended List on Twitter to help users be more successful in decision making.

- We propose an explanation ranking model to catch a high quality description for a recommended List by utilizing three feature categories: content relevance features, tweet-specific features, and publisher's authority features.
- By conducting experiments on a Twitter dataset, we have shown that our tweet-level explanation can provide helpful information about the recommended List for users.

The structure of this paper is as follows. In the subsequent section, we provide an overview of related research on explainable social recommendation. We outline the proposed model that is designed to generate an informative post-hoc explanations for the recommended Twitter List in Section 3. Our evaluation of the model is presented in Section 4, and we conclude with Section 5.

2. Related Works

The explainability of recommendation models is considered so important in order to increase users' trust and encourage them to adopt recommender systems. These explanations may serve various purposes, such as transparency, effectiveness, trust, persuasiveness, satisfaction, scrutability, and efficiency [14], [16]. To achieve a specific purpose, it is essential to carefully consider what information should be conveyed to the user through the explanation facility [15].

Explainable recommendation models can be classified into two categories: *model-intrinsic* and *model-agnostic* [17]. Model-agnostic approaches generate explanations (often called the post-hoc explanation) using separate models or techniques that are independent of the recommendation algorithm. Post-hoc explanations are often more flexible and can be applied to a wider range of recommendation algorithms, but they may not be as accurate or specific compared to intrinsic explanations. Frequently employed techniques for post-hoc explanation generation include surrogate models [18],[19] and data mining methods such as association rule mining [20] or subgraph discovery [21]. Despite their approximate nature, post-hoc approaches are effective in maintaining the accuracy of the underlying model [22]. Intrinsic explanations are generated using features that are built into the recommendation algorithm itself. These features aim to provide a more detailed and specific explanation of why a particular item was recommended. Previous research has employed various explainable models to generate intrinsic explanations including factorization models [23],[24], knowledge graph based models [25]–[27], deep learning models [28]–[30], and rule-based models [31], [32].

The explainability of *social recommender systems* is vital in establishing users' trust in the recommendations, which is fundamental to maintain the sustainability of social networks [33]. In some previous studies on explainable social recommender systems, models provide

explanations based on the user's social network [34], [35]. For example, Wang et al. [35] have developed social explanations that follow the structure of "A and B also like the item". To generate the pertinent social explanation, they developed an algorithm for identifying the optimal set of users to include in the explanation. In some other works, a heterogeneous information network is created according to user and item information and then explanations are presented as paths connecting users and the recommended items [36], [37]. For instance, Zhang et al. [37] have introduced a knowledge distillation approach to explain black-box models for recommendation. Given an embedding-based model that generates black-box recommendations, their proposed approach explained recommended items based on differentiable paths on the knowledge graph. Recently, researchers in the field of social recommender systems have started to consider user-generated content such as reviews and posts as a form of explanation for the recommendations made by these systems [38]–[40]. For instance, Ren et al. [38] have presented the social collaborative viewpoint regression model, which utilize viewpoints as explanations. These viewpoints are represented by a combination of concept, topic, and sentiment label that is extracted from both user reviews and social connections.

Compared to explainable social recommendation models, our paper focuses on providing an informative post-hoc explanation for the recommended Twitter List by automatically creating descriptions based on the List's content. We believe that such explanations can assist users in making well-informed decisions.

3. Methodology

This section is devoted to the formulation of our model to generate an informative explanation for the recommended List. Formally, let L be the set of Twitter Lists and $l \in L$ is a recommended List, given $M_l = \{m_1, m_2, \dots, m_N\}$ as a tweet collection of l , we aim to identify top-K most informative and relevant tweets of l , as a post-hoc explanation. To address this problem, the proposed

approach, comprising several components, is illustrated in Figure 1.

Initially, a recommender system provides a recommended List, denoted as l . For the set of tweets in l (i.e., M_l), three categories of features are extracted within the *Feature Extraction* component. These tweets are subsequently ranked based on their informativeness and relevance to the main topic of the List in *Tweet Ranking* component. To train the ranking model, a learning-to-rank technique is applied, ensuring that the most relevant and informative tweets are prioritized. The following subsections provide a detailed introduction to these components.

A. Feature Extraction

Our emphasis when defining features is on *Informativeness* and *Relevance* of List's tweets that are utilized to provide an explanation. Adopted from [41], in our work, *Relevance* means the degree of content closeness to the main subject of recommended Twitter List and *Informativeness* means degree of information acceptability which can explain the recommended List understandable to maximum people. The explanation ranking model employs three distinct categories of features, which are: (1) *content relevance* features, (2) *tweet-specific* features and (3) *publisher's authority* features. We provide further details about these categories below.

1) Content Relevance Features

The features of this category are used to measure the relatedness of a tweet from the recommended List $\in L$, i.e., $m \in M_l$, to the main subject of l . According to our intuition, an explanation will be more beneficial to the user if it is more semantically comparable to the recommended List. Below is a description of two features belonging to this category.

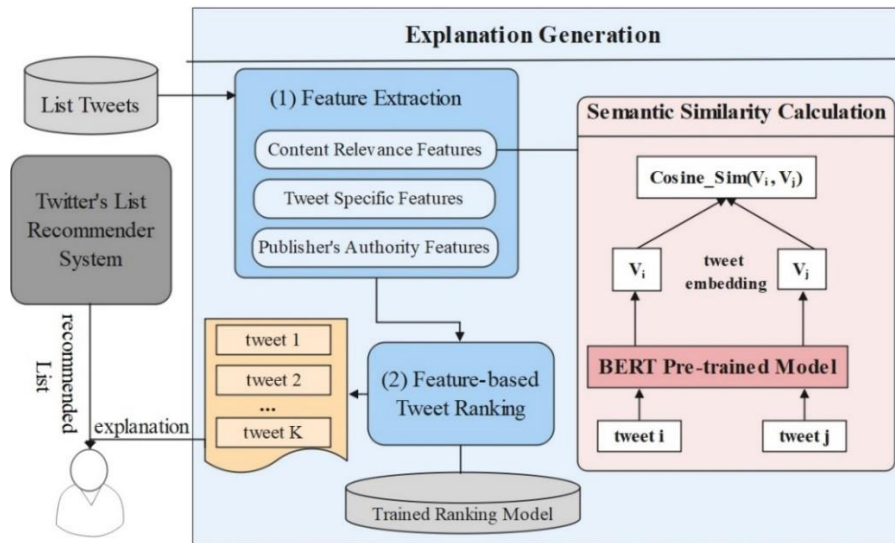


Figure 1. Overview of the proposed explanation generation model

1) *Semantic Relatedness*: we compute the semantic similarity between the tweet $m \in M_l$ and the l 's tweets (up to N:100 tweets) as a relatedness score. We use the sentence-BERT model [42] to get the embedding vectors and then compute the cosine similarity of the two embedding vectors. We calculate relatedness score as follows:

$$\text{relatedness_score}(m, l) = \frac{1}{N} \sum_{i=1}^N \text{cosine_similarity}(\vec{m}, \vec{m}_i)$$

Where m is a tweet of l which is compared with other tweets of l (i.e. m_i). If m is semantically related to the main subject of l , the relatedness score will be higher than if m is a noise tweet.

2) *Relevance to Hashtags*: A number of hashtags may be used in the $m \in M_l$ to highlight its main keywords. This feature determines the count of hashtags that are present in tweet m and are also included in the set of l 's top-10 hashtags (top-10 hashtags of l is specified through tweet history of it).

2) Tweet Specific Features

The quality of the tweet $m \in M_l$ regardless of its relatedness to the main subject of l , is measured by this category of features. Our hypothesis is that a tweet will be more helpful to serve as an explanation if it is more popular and informative than a tweet. The following features are included in this category, which were inspired by those mentioned in the literature on tweet ranking [43]–[46]:

- 1) *Length*: It is determined by how many words a tweet contains. Intuitively, a longer tweet is more likely to contain a greater amount of information than a shorter one.
- 2) *Retweet Count*: It is described as the quantity of retweets a tweet receives. The fundamental idea is that a tweet is more informative and useful if it is retweeted frequently.
- 3) *Favorite Count*: A tweet's quality and amount of appreciation may be suggested by the frequency with which users have expressed positive feelings about it.
- 4) *URL Count*: Publishers frequently augment their tweets with URLs that direct readers to more information on different web pages. As a result, the number of URLs in a tweet may affect how informative it is.
- 5) *Hashtag Count*: A tweet becomes more informative and useful the more hashtags it has.

3) Publisher's Authority Features

This particular group of features quantifies the level of authority of the individual who posted the tweet. According to our hypothesis, tweets that are shared on social media by

more authoritative users, are perceived to be of higher quality and more compelling, and thus may be more effective in providing a description for the recommended Twitter List to the user. Adopted from [43]–[45], the following features are considered as potential markers to determine a user's authority:

- 1) *Follower Count*: The amount of followers a user has, is recorded by this feature.
- 2) *Status Count*: This feature counts the total amount of tweets a user has ever posted.
- 3) *Mention Count*: This feature is used to estimate the number of times a user is mentioned in tweets.

B. Feature-based Tweet Ranking

Given M_l , the set of tweets within the recommended list l , our objective is to rank these tweets, identifying top-K most relevant and informative ones as the descriptive explanation. To achieve this, each $m \in M_l$ is represented by a feature vector, where each dimension of the vector quantifies the relevance and informativeness of m with respect to l . Specifically, utilizing the features extracted in the previous component (i.e., Feature Extraction), we apply a learning-to-rank model to efficiently rank the tweets. In the following, we describe the process of training the ranking model illustrated in Figure 1, titled *Trained Ranking Model*, which is used by the *Feature-based Tweet Ranking* component.

First, to collect the training data, we randomly selected 100 Twitter Lists from our dataset and randomly chose 30 tweets from each List as potential explanations. These tweets were then manually annotated according to the specified annotation guidelines, resulting in the creation of an explanation pool. Each explanation was assessed by human annotators and assigned one of three points according to the following criteria:

- 0: *Explanation is unrelated to the main subject of the List.*
- 1: *Explanation is related to the List but lacks informativeness.*
- 2: *Explanation is both related to the List and informative.*

Table 1 presents examples of explanations for each annotation category. With this annotated data, we proceeded to train a feature-based learning-to-rank (LTR) model, employing LambdaMART specifically. The experiments aimed at identifying the optimal LTR method are detailed in Section 4.B.

Table 1. Explanation annotation examples for each category.

Human annotation	List name	Explanation
unrelated	virus scientists	"What'd you eat for breakfast?"
	crypto currency	"The weather is perfect for along run."
related and non-informative	virus scientists	"Wearing a mask isn't fun."
	crypto currency	"Don't trip, buy the dip ⁴ !"
related and informative	virus scientists	"New on 3rd shot (booster) effectiveness vs Omicron infections from Spain in 7 million people: 51%, across all age groups, Moderna 13% better than Pfizer."
	crypto currency	"At the very least, one should be expecting a #Bitcoin bounce right around now. STH-SOPR has fallen below 1, which means top-buyers are spending their \$BTC at a realized loss. When top buyers capitulate, it is historically a local bottom."

⁴ dip in the world of cryptocurrency stands for 'Drop In Price'

This trained ranking model enables the systematic ranking of tweets based on their feature vectors, ensuring that the most relevant and informative tweets are prioritized effectively. This capability represents a critical step towards enhancing the quality and relevance of descriptive explanations for recommended Twitter Lists. Indeed, the *Feature-based Tweet Ranking* component, leveraging the trained ranking model, has the capacity to organize input tweets according to their quality, with a focus on relatedness and informativeness.

4. Experiments

A. Dataset

We collected a dataset from Twitter using Tweepy⁵. Similar to [47], the crawling process began with the Lists of ‘Ashton Kutcher’, a well-known user on Twitter. Given his Lists, we initially gathered all users who have subscribed to these Lists in order to expand the set of users. Then, we added more Lists to the collection by gathering every List that these users had subscribed to. After four iterations, our final dataset includes roughly 17,000 Lists covering a diverse range of topics. By investigating the random subset of 3000 Lists, it is realized that 61% of them do not have a description written by the List's owner. For the remaining 39%, we depict the Twitter Lists distribution by the number of description tokens in Figure 2, which states that the length of List's description is short in most cases. Therefore, providing a description with the amount of related information is important to help users be more successful in decision making.

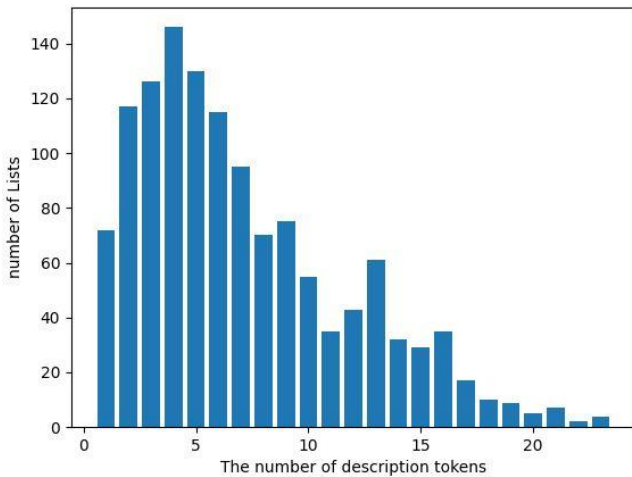


Figure 2. Twitter Lists distribution by count of description tokens

B. Experimental Settings

1) **BERT-Sentence.** As explained in Section 3, we utilize BERT-Sentence to measure the relatedness between a tweet and the main subject of the recommended List. Before applying BERT-Sentence, we slightly preprocessed the tweets by removing special characters, URLs and mentions. We apply the pre-trained model (i.e., "all-

MiniLM-L6-v2") to transform the tweets into dense vector embeddings in our experiments.

2) **Learn to Rank.** In our experiment, we used the RankLib⁶ library for learning to rank (LTR). We used three machine learning techniques to train the explanation ranking model, consisting of one linear method (Coordinate Ascent [48]) and two non-linear methods (MART [49] and LambdaMART [50]). To select the best-performing LTR method, given the train dataset, a 5-fold cross validation approach is applied to evaluate various ranking models in terms of Expected Reciprocal Rank (ERR) and Normalized Discounted Cumulative Gain (NDCG). Considering the results in Table 2, LambdaMART is selected for the rest of our experiments.

Table 2. NDCG and ERR reported by different LTR models

Model	NDCG@10	ERR@10
Coordinate Ascent	0.921	0.832
MART	0.884	0.798
LambdaMART	0.928	0.840

C. Evaluation

Similar to [51], we evaluated two aspects of our proposed model: (1) the generated explanations' quality with regards to relatedness and informativeness to the recommended List through a user study, (2) the significance of the ranking component by a pairwise analysis. As explained in Section 3, we identify top-K most informative and relevant tweets of the recommended List, as a post-hoc explanation. We set K to one in our experiments.

1) Explanations' Quality Evaluation by User Study

In order to assess the quality of the generated explanations, we initially selected 100 Lists at random and then the proposed model generated an explanation for each one. These explanations were then annotated by human annotators according to the guidelines introduced in Section 3.B.

The results of user study are reported in Figure 3, which shows the percentage of explanations annotated by each label. Based on our findings, we observed that the number of explanations that were both informative and relevant was higher than the number of explanations belonging to other categories.

2) Analysis on the Ranking Component

As discussed in Section 3, the tweets of recommended List *l* as potential explanations are ranked by the ranking component based on their relatedness to the main subject of the recommended List and their informativeness. To investigate the impact of ranking component on the quality of the final explanation, we design a pairwise evaluation. In detail, for randomly selected 100 Lists, the explanation generated by our model named *A* and the randomly selected potential explanation named *B*. Human annotators conducted pairwise evaluations between two explanations, using one of

⁵ <https://www.tweepy.org/>

⁶ <https://sourceforge.net/p/lemur/wiki/RankLib/>

the following points:

- 1: *A is more related and informative than B.*
- 2: *B is more related and informative than A.*
- 3: *A and B are almost the same, both related and equally informative.*
- 4: *A and B are almost the same, both unrelated.*

Figure 4 illustrates the results of the pairwise evaluation. For 52% of recommended Lists, top-1 ranked explanations are more related and informative than the randomly selected potential ones. In addition, for 39% of recommended List, it is challenging for annotators to determine which one is better because both explanations are useful. We conclude that in the majority of cases, top-1 ranked explanations perform as well as, or even better than, other potential explanations, which demonstrates the explanation ranking component's beneficial impact to improving explanation quality.

3) Feature Analysis

To specifically evaluate the effectiveness of each feature group (i.e., content-relevance, tweet-specific and publisher's authority), we performed an ablation study in which we removed the features of every category separately. The performance of LambdaMART as affected by the ablation study is presented in Table 3. In this table, symbols * shows statistical significance on a paired t-test with p -value less than 0.05. According to the findings, all three feature categories are effective on how well the explanation ranking model performs, and performance is decreased in terms of NDCG and ERR by removing them. The performance of the ranking component is demonstrated to be more significantly impacted by content relevance features and tweet-specific features than by publisher's authority features.

Furthermore, to assess the individual impact of each feature, we conducted an ablation study by systematically removing one feature at a time. The results of this study, presented in Table 4, provide valuable insights into the relative importance of features within each category. The findings highlight *semantic relatedness* as the pivotal feature within the content relevance category, with its removal resulting in a significant decrease in NDCG, from 0.928 to

0.889. *Tweet length* emerges as the primary feature within the tweet-specific category. Conversely, *follower count* within the publisher's authority category exhibits a negative impact on NDCG. Its removal leads to a slight increase in NDCG, from 0.928 to 0.931. Overall, our analysis demonstrates that the majority of features significantly contribute to the effectiveness of ranking explanations.

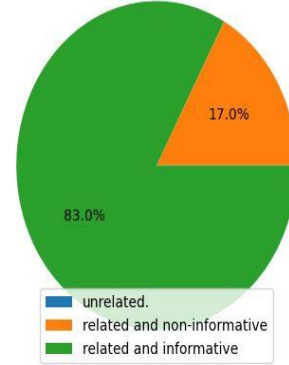


Figure 3. The results of user study

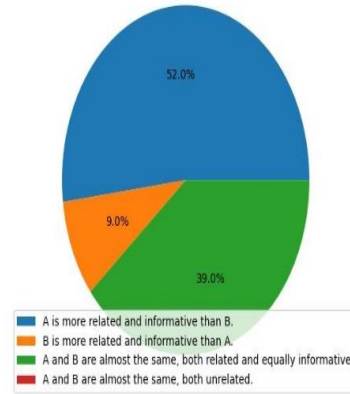


Figure 4. Pairwise annotation results

Table 3. Ablation study results (feature category removal).

	NDCG@10	▼	ERR@10	▼
All Features	0.928		0.840	
- Content Relevance Features	0.879	5.28 % *	0.812	3.33 % *
- Tweet Specific Features	0.868	6.47 % *	0.693	17.5 % *
- Publisher's Authority Features	0.926	0.22 %	0.833	0.83 %

Table 4. Ablation study results (feature removal).

	NDCG@10	ERR@10
All Features	0.928	0.840
- Semantic Relatedness	0.889	0.828
- Relevance to Hashtags	0.923	0.824
- Length	0.894	0.807
- Retweet Count	0.926	0.839
- Favorite Count	0.924	0.831
- URL Count	0.921	0.832
- Hashtag Count	0.916	0.829
- Follower Count	0.931	0.838
- Status Count	0.925	0.839
- Mention Count	0.924	0.830

5. Conclusion

In this paper, in order to help users to make an informed decision on social media, we proposed a post-hoc explanation model for List recommendations on Twitter. The proposed model provides a high quality description as explanation using the content of the recommended List. More specifically, by using three feature categories from different aspects, our model ranks explanations according to their relatedness and informativeness. In our experiments, the quality of final explanations are evaluated by user study. Also, the importance of the explanation ranking component is investigated.

In the current work, we only use one related and informative tweet of the recommended List as a final explanation. In future studies, our aim is to determine the minimum number of tweets required to provide reliable indicators of the usefulness of an explanation.

6. References.


- [1] S. de la Rouviere and K. Ehlers, "Lists as coping strategy for information overload on Twitter," Proceedings of the 22nd International Conference on World Wide Web, 2013, pp. 199–200.
- [2] V. Rakesh, D. Singh, B. Vinzamuri, and C. K. Reddy, "Personalized recommendation of twitter lists using content and network information," Proceedings of the International AAAI Conference on Web and Social Media, 2014, pp. 416–425.
- [3] L. Chen, Y. Zhao, S. Chen, H. Fang, C. Li, and M. Wang, "iplug: Personalized list recommendation in twitter," Web Information Systems Engineering–WISE 2013: 14th International Conference, Nanjing, China, 2013, pp. 88–103.
- [4] C.-H. Tsai and P. Brusilovsky. (2020, Oct.). The effects of controllability and explainability in a social recommender system. *User Modeling and User-Adapted Interaction*. [Online]. 31, pp. 591–627. Available: 10.1007/s11257-020-09281-5
- [5] A. Papadimitriou, P. Symeonidis, and Y. Manolopoulos. (2012, May.). A generalized taxonomy of explanations styles for traditional and social recommender systems. *Data Mining and Knowledge Discovery*. [Online]. 24, pp. 555–583. Available: 10.1007/s10618-011-0215-0
- [6] Y. Zhang and X. Chen. (2020, Mar.). Explainable recommendation: A survey and new perspectives. *Foundations and Trends® in Information Retrieval*. [Online]. 14(1), pp. 1–101. Available: 10.1561/15000000066
- [7] M. T. Ribeiro, S. Singh, and C. Guestrin, "Model-agnostic interpretability of machine learning," *arXiv preprint arXiv:1606.05386*, 2016.
- [8] D. Shmoryahu, G. Shani, and B. Shapira, "Post-hoc Explanations for Complex Model Recommendations using Simple Methods," *IntRS@ RecSys*, 2020, pp. 26–36.
- [9] Q. Chen, J. Lin, Y. Zhang, H. Yang, J. Zhou, and J. Tang, "Towards knowledge-based personalized product description generation in e-commerce," Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 3040–3050.
- [10] A. Nadamoto, K. Fukumoto, R. Takeuchi, H. Terada, and M. Bato. (2023). Automatic generation of product description using deep learning methods. *Journal of Data Intelligence*. [Online]. 4(1 & 2), pp. 134–148. Available: rintonpress.com
- [11] Q. Zhang, B. Guo, S. Liu, J. Liu, and Z. Yu. (2022, Apr.). CrowdDesigner: information-rich and personalized product description generation. *Frontiers of Computer Science*. [Online]. 16(6), p. 166339. Available: 10.1007/s11704-022-1193-7
- [12] J. Wang, Y. Hou, J. Liu, Y. Cao, and C.-Y. Lin, "A statistical framework for product description generation," Proceedings of the Eighth International Joint Conference on Natural Language Processing, 2017, pp. 187–192.
- [13] S. Gerani, Y. Mehdad, G. Carenini, R. Ng, and B. Nejat, "Abstractive summarization of product reviews using discourse structure," Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1602–1613.
- [14] K. Balog and F. Radlinski, "Measuring recommendation explanation quality: The conflicting goals of explanations," Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval, 2020, pp. 329–338.
- [15] I. Nunes and D. Jannach. (2017, Oct.). A systematic review and taxonomy of explanations in decision support and recommender systems. *User Modeling and User-Adapted Interaction*. [Online]. 27, pp. 393–444. Available: 10.1007/s11257-017-9195-0
- [16] N. Tintarev and J. Masthoff, "A survey of explanations in recommender systems," IEEE 23rd international conference on data engineering workshop, Istanbul, Turkey, 2007, pp. 801–810.
- [17] M. Caro-Martínez, G. Jiménez-Díaz, and J. A. Recio-García. (2021, Jul.). Conceptual modeling of explainable recommender systems: an ontological formalization to guide their design and development. *Journal of Artificial Intelligence Research*. [Online]. 71, pp. 557–589. Available: 10.1613/jair.1.12789
- [18] C. Nóbrega and L. Marinho, "Towards explaining recommendations through local surrogate models," Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, 2019, pp. 1671–1678.
- [19] B. C. G. Lee, K. Lo, D. Downey, and D. S. Weld, "Explanation-based tuning of opaque machine learners with application to paper recommendation," *arXiv preprint arXiv:2003.04315*, 2020.
- [20] G. Peake and J. Wang, "Explanation mining: Post hoc interpretability of latent factor models for recommendation systems," Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 2060–2069.
- [21] C. Lonjarret, C. Robardet, M. Plantervit, R. Auburtin, and M. Atzmueller, "Why should i trust this item? explaining the recommendations of any model," IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA), Sydney, NSW, Australia, 2020, pp. 526–535.
- [22] M. Du, N. Liu, and X. Hu. (2019, Dec.). Techniques for interpretable machine learning. *Communications of the ACM*. [Online]. 63(1), pp. 68–77. Available: 10.1145/3359786
- [23] Y. Zhang, G. Lai, M. Zhang, Y. Zhang, Y. Liu, and S. Ma, "Explicit factor models for explainable recommendation based on phrase-level sentiment analysis," 37th international ACM SIGIR conference on Research & development in information retrieval, 2014, pp. 83–92.
- [24] N. Wang, H. Wang, Y. Jia, and Y. Yin, "Explainable recommendation via multi-task learning in opinionated text data," 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, 2018, pp. 165–174.
- [25] Q. Ai, V. Azizi, X. Chen, and Y. Zhang. (2018, Sep.). Learning heterogeneous knowledge base embeddings for explainable recommendation. *Algorithms*. [Online]. 11(9), p. 137. Available: 10.3390/a11090137
- [26] H. Wang et al. and M. Guo, "Ripplenet: Propagating user preferences on the knowledge graph for recommender systems," Proceedings of the 27th ACM international conference on information and knowledge management, 2018, pp. 417–426.

- [27] Y. Xian, Z. Fu, S. Muthukrishnan, G. De Melo, and Y. Zhang, "Reinforcement knowledge graph reasoning for explainable recommendation," *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, 2019, pp. 285–294.
- [28] F. Fusco, M. Vlachos, V. Vasileiadis, K. Wardatzky, and J. Schneider, "RecoNet: An Interpretable Neural Architecture for Recommender Systems.," *International Joint Conference on Artificial Intelligence*, 2019, pp. 2343–2349.
- [29] Y. Lu, R. Dong, and B. Smyth, "Why I like it: multi-task learning for recommendation and explanation," *Proceedings of the 12th ACM Conference on Recommender Systems*, 2018, pp. 4–12.
- [30] Z. Chen *et al.* and E. Chen, "Co-attentive multi-task learning for explainable recommendation.," *International Joint Conference on Artificial Intelligence*, 2019, pp. 2137–2143.
- [31] X. Wang, X. He, F. Feng, L. Nie, and T.-S. Chua, "Tem: Tree-enhanced embedding model for explainable recommendation," *Proceedings of the 2018 world wide web conference*, 2018, pp. 1543–1552.
- [32] W. Ma *et al.* and X. Ren, "Jointly learning explainable rules for recommendation with knowledge graph," *The world wide web conference*, 2019, pp. 1210–1221.
- [33] W. Sherchan, S. Nepal, and C. Paris. (2013, Aug.). A survey of trust in social networks. *ACM Computing Surveys (CSUR)*. [Online]. 45(4), pp. 1–33. Available: 10.1145/2501654.250166
- [34] A. Sharma and D. Cosley, "Do social explanations work? Studying and modeling the effects of social explanations in recommender systems," *Proceedings of the 22nd international conference on World Wide Web*, 2013, pp. 1133–1144.
- [35] B. Wang, M. Ester, J. Bu, and D. Cai, "Who also likes it? generating the most persuasive social explanations in recommender systems," *Proceedings of the AAAI Conference on Artificial Intelligence*, 2014.
- [36] C. Shi, Z. Zhang, P. Luo, P. S. Yu, Y. Yue, and B. Wu, "Semantic path based personalized recommendation on weighted heterogeneous information networks," *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, 2015, pp. 453–462.
- [37] Y. Zhang, X. Xu, H. Zhou, and Y. Zhang, "Distilling structured knowledge into embeddings for explainable and accurate recommendation," *Proceedings of the 13th international conference on web search and data mining*, 2020, pp. 735–743.
- [38] Z. Ren, S. Liang, P. Li, S. Wang, and M. de Rijke, "Social collaborative viewpoint regression with explainable recommendations," *Proceedings of the tenth ACM international conference on web search and data mining*, 2017, pp. 485–494.
- [39] J. Zheng, Q. Li, J. Liao, and S. Wang. (2021, Oct.). Explainable link prediction based on multi-granularity relation-embedded representation. *Knowledge-Based Systems*. [Online]. 230, p. 107402. Available: 10.1016/j.knosys.2021.107402
- [40] J. Zheng, Z. Qin, S. Wang, and D. Li. (2022, Jun.). Attention-based explainable friend link prediction with heterogeneous context information. *Information Sciences*. [Online]. 597, pp. 211–229, 2022. Available: 10.1016/j.ins.2022.03.010
- [41] D. Rudrapal, A. Das, and B. Bhattacharya. (2019, Jun.). Ranking of event-focused english tweets based on relevance and informativeness. *Computación y Sistemas*. [Online]. 23(2), pp. 491–500. Available: 10.13053/cys-23-2-3011
- [42] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," *arXiv preprint arXiv:1904.09675*, 2019.
- [43] K. Chen, T. Chen, G. Zheng, O. Jin, E. Yao, and Y. Yu, "Collaborative personalized tweet recommendation," *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, 2012, pp. 661–670.
- [44] C. De Maio, G. Fenza, M. Gallo, V. Loia, and M. Parente. (2019, Apr.). Time-aware adaptive tweets ranking through deep learning. *Future Generation Computer Systems*. [Online]. 93, pp. 924–932. Available: 10.1016/j.future.2017.07.039
- [45] Y. Duan, L. Jiang, T. Qin, M. Zhou, and H. Y. Shum, "An empirical study on learning to rank of tweets," *Proceedings of the 23rd international conference on computational linguistics (Coling 2010)*, 2010, pp. 295–303.
- [46] K. Sailunaz, J. Kawash, and R. Alhaji. (2022, Apr.). Tweet and user validation with supervised feature ranking and rumor classification. *Multimedia Tools and Applications*. [Online]. 81(22), pp. 31907–31927. Available: 10.1007/s11042-022-12616-6
- [47] D. Kim, Y. Jo, I.-C. Moon, and A. Oh, "Analysis of twitter lists as a potential source for discovering latent characteristics of users," *ACM CHI workshop on microblogging*, Citeseer, 2010.
- [48] D. Metzler and W. Bruce Croft. (2007, Jan.). Linear feature-based models for information retrieval. *Information Retrieval*. 10, pp. 257–274. Available: 10.1007/s10791-006-9019-z
- [49] J. H. Friedman. (2001, Oct.). Greedy function approximation: a gradient boosting machine. *Annals of statistics*. [Online]. 29(5), pp. 1189–1232. Available: jstor.org/stable/2699986
- [50] Q. Wu, C. J. Burges, K. M. Svore, and J. Gao. (2009, Sep.). Adapting boosting for information retrieval measures. *Information Retrieval*. [Online]. 13, pp. 254–270. Available: 10.1007/s10791-009-9112-1
- [51] C. Chen, M. Zhang, Y. Liu, and S. Ma, "Neural attentional rating regression with review-level explanations," *Proceedings of the 2018 world wide web conference*, 2018, pp. 1583–1592.



Machine Learning Classifiers and Data Synthesis Techniques to Tackle with Highly Imbalanced COVID-19 Data

Research Article

Avaz Naghipour¹ , Mohammad Reza Abbaszadeh Babil Soflaei², Mostafa Ghaderi-Zefrehei³

DOI: [10.22067/cke.2024.88940.1121](https://doi.org/10.22067/cke.2024.88940.1121)

Abstract The COVID-19 pandemic has highlighted the urgent need for rapid and accurate diagnostic methods. In this study, we evaluate three machine learning models—Random Forest (RF), Logistic Regression (LR) and Decision Tree (DT)—for detecting COVID-19 trained on preprocessed imbalanced datasets. The dataset used in this study is heavily imbalanced, with 5086 negative and 558 positive cases, posing a significant challenge for effective model training. To this end, we demonstrate the capability of two advanced data synthesis algorithms, Conditional Tabular Generative Adversarial Network (CTGAN) and Tabular Variational Autoencoder (TVAE), in addressing the class imbalance inherent in the dataset. The classifiers trained on the original as well as the balanced datasets were evaluated for comparison. Our findings reveal that RF obtains the highest accuracy of 98.83% on the CTGAN-balanced dataset. In conclusion, our results verify the potential of coupling data synthesis with traditional machine learning for the diagnosis of COVID-19. We hope that we become a valuable contributor to the ongoing AI for pandemic.

Keywords COVID-19 Detection, Machine Learning, CTGAN, TVAE, Class Imbalance.

1. Introduction

In late 2019, a pneumonia outbreak originated in Wuhan, China, which was subsequently identified as being caused by the SARS-CoV-2 virus by the World Health Organization (WHO) [1]. SARS-CoV-2 is an enveloped virus with a positive-sense, single-stranded RNA genome [2]. This virus primarily targets the human respiratory system and is highly transmissible through respiratory droplets from coughing, sneezing, and direct physical contact [3]. Additionally, it can spread via contact with contaminated surfaces, where the virus can persist for several days depending on environmental conditions [4].

Common symptoms of the infection include fever, dry cough, loss of taste and smell, sore throat, and muscle pain [2]. The pandemic has had widespread impacts, leading to the postponement of school examinations, closure of offices, and widespread layoffs [5]. Estimates from the World Health Organization (WHO) show that the full death toll associated directly or indirectly with the COVID-19 pandemic between 1 January 2020 and 31 December 2021 was approximately 14.9 million [6]. The recent surge in data science has empowered healthcare professionals by providing tools to analyze massive datasets of health information for disease detection. This progress is driven by various techniques like deep learning, data mining, and especially machine learning (ML). However, a key challenge remains: selecting the most appropriate ML algorithms that can learn effectively from existing data and make accurate predictions for entirely new cases [2]. ML is crucial in the healthcare sector, particularly for diagnosing diseases, detecting outbreaks, and preventing illnesses. ML algorithms are employed for numerous purposes, including predicting diabetes [7], forecasting the progression of Alzheimer's disease [8], heart disease [9], and other medical conditions. Due to the scarcity of tabular data on COVID-19, we tested our hypothesis using a dataset available on Kaggle (at this link), which clearly represents the clinical symptoms of COVID-19. This dataset, like many others in the field of ML, is heavily imbalanced, containing 5086 negative cases and 558 positive cases, resulting in a 1:9 ratio. Training ML models on imbalanced datasets poses several challenges: the models tend to be biased towards the majority class, leading to poor performance in detecting the minority class [10]. This imbalance can result in lower recall for the minority class, skewed accuracy metrics, and an overall decrease in the model's ability to generalize well to new, unseen data [11]. Our study seeks to improve the classification of COVID-19 by conducting a thorough

* Manuscript received: 2024 July 15, Revised, 2024 October 4, Accepted, 2024 November 6.

¹ Corresponding Author. Assistant Professor, Department of Computer Engineering, University College of Nabi Akram, Tabriz, Iran. Email: naghipour@ucna.ac.ir.

² Master of Artificial Intelligence, Department of Computer Engineering, University College of Nabi Akram, Tabriz, Iran.

³ Associate Professor, Department of Animal Science, University of Yasouj, Yasouj, Iran.

comparative analysis of three machine learning (ML) models: Logistic Regression (LR), Random Forest (RF), and Decision Tree (DT). Additionally, to mitigate the effects of imbalanced dataset, we investigate two advanced data augmentation techniques: Conditional Tabular Generative Adversarial Network (CTGAN) [12] and Tabular Variational Autoencoder (TVAE) [12], to balance the dataset. To boost classification accuracy and evaluate the effectiveness of different oversampling techniques, we trained each ML model on data balanced using these techniques. This allowed us to compare the models' performance based on the quality of the synthetic data generated by each technique. Additionally, we ensured the effectiveness of data balancing by comparing the performance of these models against models trained on both the original unbalanced dataset and a baseline model. Our results show exceptional accuracy and performance metrics, exceeding those reported in other studies.

The structure of the study is outlined as follows: Section 2 provides an overview of the relevant literature related to our study. In Section 3, the proposed methodology is detailed, with a thorough explanation of its components and techniques like CTGAN and other used methods. Section 4 presents the experimental findings, including the results of the proposed method. Section 5 is the discussion section where the findings of the study are analyzed and compared with other studies, along with an evaluation of the proposed model's strengths and limitations. Finally, the article is concluded in Section 6, with a summarization of the key findings and discussion about implications for future research in the field.

2. Literature Review

Machine learning has been extensively employed in various domains to address challenges posed by the COVID-19 pandemic. This section reviews existing literature in the fields of COVID-19 vaccine uptake prediction, COVID-19 detection through medical images, and techniques to address class imbalance in datasets. These studies are grouped based on the methodologies they use, highlighting the relevance of each to our research, and comparing the datasets and performance metrics where appropriate.

a. Machine Learning for COVID-19 Vaccine Acceptance and Uptake Prediction

Within the realm of machine learning applied to vaccination studies, recent research has explored predicting vaccine acceptance and identifying key factors influencing uptake. For instance, a study [13] investigates barriers to COVID-19 vaccine uptake in Ghana using a cross-sectional survey and machine learning algorithms. The study identifies significant factors, such as the type of medical facility visited and the presence of underlying conditions, with the random forest model emerging as the most effective predictor. Similarly, another study [14] applied machine learning algorithms to assess COVID-19 vaccine acceptance in countries where residents had already been vaccinated. This study differs from [13] by focusing on vaccine acceptance in different regions and contexts, showcasing machine learning's versatility in vaccine-related studies. Further, a study [15] applied

machine learning algorithms to analyze vaccination rates across states in the United States. While both [14] and [15] utilize machine learning, [14] targets acceptance while [15] investigates actual vaccination rates, highlighting the diverse applications of machine learning in vaccine-related health studies. However, none of these studies address the issue of class imbalance in datasets, which is a crucial aspect of improving predictive accuracy, especially in health-related research. Our study seeks to extend this work by incorporating advanced data balancing techniques like CTGAN and TVAE to tackle this imbalance.

b. Machine Learning and Deep Learning for COVID-19 Detection Using X-ray Images

A significant body of work has focused on using machine learning and deep learning techniques to detect COVID-19 through medical images, particularly X-rays. One such study [2] employed machine learning algorithms to detect lung changes associated with COVID-19 from X-ray images. The models classified X-ray images into categories such as COVID-19 patients, pneumonia patients, and healthy individuals. Among the models tested, VGG-19 with augmentation achieved the best performance, with 99% training accuracy and 98% testing accuracy. This approach shows great potential for enhancing patient prognosis tracking and supporting treatment efficacy studies. Several other studies have also demonstrated the efficacy of deep learning techniques in detecting COVID-19. For instance, a study [16] utilized Convolutional Neural Networks (CNNs) combined with a filter family and the weight-sharing feature extractor SqueezeNet. The study achieved high detection rates using deep learning for COVID-19 cases, illustrating the power of CNN features and neural network classifiers. Similarly, [17] and [18] applied CNN models for COVID-19 detection, achieving accuracies of 90% and 97%, respectively. In particular, [18] proposed a CNN model for detecting COVID-19-associated changes from X-ray images and demonstrated high accuracy across various classes. In another study [19], the authors employed several deep learning models, including CNNs, long short-term memory (LSTM) networks, GANs, and residual neural networks (ResNets), for classifying COVID-19 from other pneumonia causes using chest X-ray images. Among these, the CNN-based approach showed the highest accuracy of 99%. Despite their promising results, these studies predominantly focus on deep learning methods without addressing class imbalance, which can skew results, especially in smaller datasets with uneven class distributions. In contrast, our study applies data balancing techniques such as CTGAN and TVAE, which allow us to effectively balance the dataset and improve the robustness of machine learning models.

c. Handling Class Imbalance in COVID-19 Datasets

Class imbalance is a recurring issue in medical datasets, particularly in COVID-19 studies, as the number of positive cases is often significantly lower than the number of negative cases. Addressing this issue is crucial for improving model performance and ensuring that predictions are not biased toward the majority class. A notable study [5] tackled this challenge by employing

various oversampling techniques, including the Synthetic Minority Oversampling Technique (SMOTE). SMOTE generates synthetic samples of the minority class to balance the dataset, and this study further enhanced it by introducing a modified version called Outlier-SMOTE, which focuses on data points that are farther from others. The proposed method improved performance across several benchmark datasets, including a COVID-19 dataset. While SMOTE and its variants are widely used, they may not always be sufficient to handle more complex data distributions, particularly in tabular data. Our study builds on this by implementing advanced generative approaches such as Conditional Tabular Generative Adversarial Networks (CTGAN) and Tabular Variational Autoencoders (TVAE) to synthesize realistic samples from the minority class. These techniques have shown superior performance in balancing datasets, and our results indicate that CTGAN, in particular, outperforms traditional methods like SMOTE. For instance, our CTGAN-balanced dataset, trained with the Random Forest model, achieved accuracy levels comparable to deep learning models, underscoring the effectiveness of GAN-based approaches in handling class imbalance.

d. Machine Learning for Predicting COVID-19 in Vulnerable Populations and Other Domains

In addition to detection and vaccination prediction, machine learning has been applied to address the unique challenges posed by COVID-19 in vulnerable populations and other sectors. One study [20] evaluated a machine learning model's ability to predict COVID-19 diagnosis among individuals with intellectual and developmental disabilities (IDD). The random forest model, trained on over 700 variables from three major IDD-specific datasets, achieved an accuracy of 62.5%. This demonstrates the challenges in applying machine learning to vulnerable populations where data availability and quality may be limited. Furthermore, machine learning has been applied beyond healthcare. For example, [21] developed a predictive model using routine clinical laboratory test data to forecast patient survival outcomes. The combination of Lasso and SVM algorithms produced an ROC curve area of 0.9277 using just eight clinical parameters. In a non-health-related study [22], machine learning was applied to a global aviation dataset to predict financial distress. This study highlighted the potential of machine learning to provide accurate predictions in industries heavily impacted by the pandemic.

e. Deep Learning for Automatic COVID-19 Diagnosis Using Chest X-rays

Several studies have employed deep learning models to automatically diagnose COVID-19 from chest X-rays. A study [23] modified deep learning architectures such as VGG16, VGG19, ResNet50, and InceptionV3 to classify COVID-19 cases. These models, collectively termed "COV-DLS," achieved high classification accuracies, with Modified-VGG16 achieving the highest at 98.61%. Another study [24] applied transfer learning to automatically detect COVID-19 from chest X-ray images, with the VGG16 model achieving 98% testing accuracy. Finally, another deep learning study [25] explored pre-

trained CNN models for the automatic diagnosis of COVID-19 from chest X-rays, using a dataset of over 1,200 CXR images from COVID-19 patients. The VGG16 model achieved the highest accuracy of 98.28%, demonstrating the potential of CNN-based models for rapid and accurate COVID-19 detection. However, similar to the other deep learning approaches reviewed, these models did not consider class imbalance, which can lead to overfitting in imbalanced datasets. Our study, by contrast, addresses this issue by implementing CTGAN and TVAE techniques, providing a more robust solution for handling class imbalance.

In summary, our study fills a critical gap in the literature by focusing on addressing class imbalance in COVID-19 datasets using advanced data synthesis techniques such as CTGAN and TVAE. By comparing the performance of Random Forest, Logistic Regression, and Decision Tree models, our research offers a comprehensive evaluation of machine learning models trained on both balanced and imbalanced datasets, with results that are comparable to or exceed those of deep learning models.

3. Methodology

This section outlines the methodology employed in our study for COVID-19 detection using the COVID-19 dataset. We began by conducting rigorous data preprocessing procedures to ensure data quality and prepare the dataset for input into machine learning models. Preprocessing steps included feature selection, column dropping, label encoding, train/test splitting, and feature standardization. All experiments were performed using Python 3.9.18 on a system running Windows 11 with 16GB of RAM, an NVIDIA RTX3070TI graphics card, and an AMD Ryzen™ 9 6900HX CPU, ensuring computational efficiency and accuracy in our analyses. Detailed descriptions of each step in our methodology are provided in the following subsections. Figure 1 provides a visual diagram of the proposed ensemble classifier.

a. Dataset Description

The dataset used in this study is the COVID-19 dataset which is a publicly available dataset at this link, and it is an invaluable resource in the realm of healthcare and machine learning. This dataset comprises anonymized data from patients at Hospital Israelita Albert Einstein in São Paulo, Brazil, who underwent SARS-CoV-2 RT-PCR and additional laboratory tests during their hospital visits [5]. It includes 5644 test samples from various patients, evaluated across 111 attributes including Haemoglobin, Platelets, and Arterial Blood Gas Analysis. Among these samples, only 553 individuals tested positive for COVID-19. This reveals a notable class imbalance, with a ratio of 1:9 (minority to majority class).

b. Preprocessing

In this section, we provide a comprehensive overview of the preprocessing techniques employed to prepare the dataset for machine learning analysis. Each step is meticulously detailed, covering approaches for feature selection, label encoding, train/test partitioning, and feature standardization. We also explain the significance

and reasoning behind hyperparameter tuning, using `GridSearchCV` to optimize the performance of our classifiers.

c. Feature Selection

Given the heavily imbalanced nature of our dataset, the dataset was also significantly influenced by null values. This required additional feature selection and reduction to ensure the quality of data and performance of our models. Initially, features that had a null value percentage of more than 90% were removed. These features are likely to have very few and they are too insubstantial to contribute any inference into the modeling procedure. It is best to remove them to ensure a higher overall quality of the dataset. Next, other features that were taken out are 'Patient ID', 'Patient admitted to regular ward (1=yes, 0=no)', 'Patient admitted to semi-intensive unit (1=yes, 0=no)' and 'Patient admitted to intensive care unit (1=yes, 0=no)' as these features were not informative and not related to our objective of

predicting whether a patient was positive or not. Subsequently, features with zero variance, particularly feature 'Parainfluenza 2', was dropped out. This meticulous feature selection ensured that only the most relevant and informative features were retained for subsequent analysis. Furthermore, according to study [5], we also conducted a similar preprocessing step. The last 19 columns in our dataset were associated with the presence of antigens, represented as binary values (0 or 1). Due to significant null values in these columns, we calculated a row-wise sum. Subsequently, all 19 columns were merged into a single column named 'other_disease'. It was determined by the authors of [5] that 13% of the patients tested positive for at least one antigen. Any remaining scattered null values were replaced with the mean of their respective non-null values.

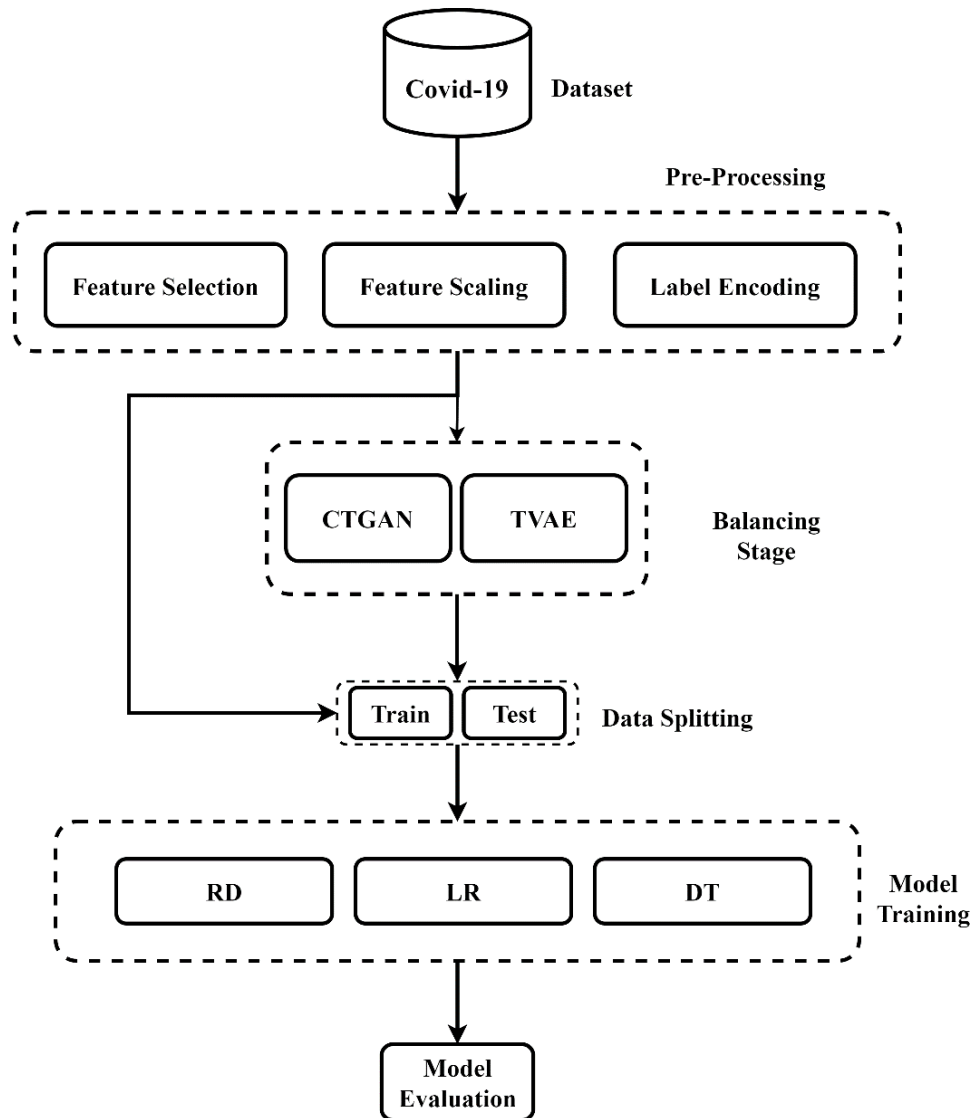


Figure 1. Flowchart of the Proposed Method

3.2.2 Data Splitting

Data splitting is a fundamental technique in ML that involves dividing the dataset into separate subsets for training, validation, and testing. This method facilitates model evaluation and ensures that classifiers are assessed on unseen data, thereby improving their generalizability. In our study, we partitioned the dataset into training and testing sets using the `train_test_split` function from the Python scikit-learn [26] package. Specifically, we adopted an 80:20 split ratio for the classifiers in our proposed method.

3.2.3 Feature Scaling

Feature scaling is a crucial component in machine learning pipelines, ensuring that the range of features is standardized. This standardization reduces the impact of varying magnitudes and enhances the convergence of optimization algorithms. In our study, we utilized the StandardScaler module from the scikit-learn [26] library to perform feature scaling. The standardization process is mathematically expressed as (1)

$$Z = \frac{x - \mu}{\sigma}$$

In this transformation, Z represents the standardized value of the feature, while x denotes its original value. The parameters μ and σ represent the mean and standard deviation of the feature, respectively. Through this mathematical process, the distribution of the feature is effectively centered around a mean of zero with a standard deviation of one.

d. Hyperparameter Tuning

Hyperparameter tuning is a meticulous process aimed at selecting the optimal hyperparameters for a machine learning model, enhancing its functionality and generalizability. This comprehensive search across specified parameter values refines model performance and tailors it to the dataset's nuances. In our research, we employed *GridSearchCV* [26] from the scikit-learn library to determine the best hyperparameters for all the classifiers

used. The results of this grid search, detailing the optimal parameters for each model, are summarized in Table 1.

e. Machine Learning Classifiers

Machine learning (ML) algorithms represent a significant advancement over classical algorithms, possessing the ability to learn from data and improve their performance autonomously [27]. By leveraging machine learning techniques, systems can enhance their capabilities and adapt to various environments without explicit programming. ML algorithms are broadly categorized into supervised and unsupervised learning approaches, each offering unique capabilities for data analysis and prediction [28]. The following section introduces the ML algorithms used as classifiers in our methodology, implemented with the scikit-learn [26] Python package, specifically for the detection of COVID-19. The selection of Random Forest (RF), Logistic Regression (LR), and Decision Tree (DT) classifiers was driven by their complementary strengths, simplicity, and ease of implementation. RF is robust and handles imbalanced datasets effectively, providing high accuracy and insights into feature importance. LR offers simplicity and interpretability, serving as a solid baseline for comparison. DTs are not only easily interpretable but also capable of capturing non-linear relationships. Using these classifiers allows us to evaluate the impact of data synthesis techniques like CTGAN and TVAE comprehensively, ensuring a thorough analysis of COVID-19 detection performance.

Random Forest (RF). Random Forest is an ensemble learning technique that involves training multiple decision trees during the training phase [29]. In this method, each tree within the forest independently predicts the target variable, and the final prediction is determined through majority voting among the predictions of all trees. The primary advantage of Random Forests lies in their ability to reduce classification errors compared to traditional classifiers, while also being less prone to overfitting [7].

Table 1. Best Hyperparameters Obtained through GridSearchCV

Hyperparameters with			
Model Name	Original Data	CTGAN-Balanced	TVAE-Balanced
LR	C:1, penalty:l2, solver:liblinear	C:1, penalty:l2, solver:liblinear	C:1, penalty:l2, solver:liblinear
DT	max_depth:10, min_samples_split:5	max_depth:10, min_samples_split:5	max_depth:5, min_samples_split:2
RF	max_depth:None, min_samples_split:2, n_estimators:100	max_depth:None, min_samples_split:5, n_estimators:200	max_depth:None, min_samples_split:2, n_estimators:100

Decision Trees (DTs) are constructed, and their majority voting is calculated as follows (2) [7]: Let $Cq(p)$ represent the class prediction of the q th Random Forest, and MV denotes the majority vote of the constructed Decision Trees.

This approach combines the predictions of multiple decision trees to produce a final prediction that is generally more robust and accurate than the prediction of any individual tree.

$$c(p) = MV\{C_q(p)\} \quad (2)$$

Logistic Regression (LR). Logistic Regression is a linear classification algorithm that models the probability of a binary outcome using a logistic function. It is employed to estimate the likelihood of an event occurring based on predictor variables [30,31]. The algorithm utilizes the sigmoid function (3), which maps the input to the range between 0 and

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad (3)$$

Decision Tree (DT). Decision Tree is a versatile and intuitive machine learning algorithm used for both classification and regression tasks. It operates by recursively partitioning the feature space into regions, guided by the values of input features, in a hierarchical manner. At each node of the tree, a decision is made based on a feature's value, aiming to maximize information gain or minimize impurity, such as entropy or Gini impurity. The decision tree algorithm selects the feature and threshold that optimize information gain or minimize impurity at each node, resulting in a hierarchical structure that facilitates interpretation and decision-making. Decision trees are favored for their simplicity, interpretability, and capability to handle both numerical and categorical data. Additionally, ensemble methods like Random Forests and Gradient Boosting Trees extend the capabilities of decision trees, enhancing their predictive power and robustness.

f. Our Approach to Balancing the Dataset

In addressing the class imbalance in our dataset, we employed two advanced data synthesis techniques: Conditional Tabular Generative Adversarial Network (CTGAN) and Tabular Variational Autoencoder (TVAE). Our primary goal was to synthesize additional data to equalize the number of instances between the negative and positive classes. Initially, our dataset comprised 5086 negative and 558 positive cases, creating a significant imbalance that posed a challenge for effective machine learning model training. To mitigate this issue, we adopted the following approach:

- I. **Training on Positive Cases:** We exclusively trained both CTGAN and TVAE on the 558 positive cases. This step was crucial to ensure that the models could accurately learn the data distribution specific to the positive class.
- II. **Synthesizing Data:** After training, both CTGAN and TVAE were utilized to generate synthetic data. The

aim was to create enough synthetic positive cases to balance the dataset. Specifically, we synthesized 4528 additional positive cases. This number was calculated to match the number of negative cases (5086), ensuring an equal distribution.

- III. **Combining Data:** The 4528 synthetic positive cases generated by CTGAN and TVAE were then added to the original dataset. This resulted in a balanced dataset with 5086 instances each of negative and positive cases.
- IV. **By employing this method,** we ensured that our machine learning models were trained on a balanced dataset, thereby enhancing their ability to accurately detect COVID-19 cases without bias towards the majority class. This approach demonstrates the effectiveness of CTGAN and TVAE in generating synthetic data to address class imbalances in datasets. Figure 2 demonstrates the class distribution of both balanced and imbalanced versions of the dataset.

g. Utilization of CTGAN for Data Balancing

In numerous real-world scenarios, datasets often exhibit significant class imbalance, where one class greatly outweighs the others [32]. This class imbalance poses challenges for machine learning models, as they typically struggle to perform well on the minority class due to a lack of sufficient examples for learning. CTGAN (Conditional Tabular Generative Adversarial Network) [12] emerges as a powerful solution for generating synthetic data that accurately replicates the distribution of real-world data, leveraging the capabilities of Generative Adversarial Networks (GANs) [33]. CTGAN was specifically developed to address various challenges encountered in generating synthetic tabular data in the field of Informatics in Medicine, including handling mixed data types, non-Gaussian and multi-modal distributions, and highly imbalanced categorical columns. Additionally, the algorithm combines softmax (4) and tanh functions in its output to effectively generate a blend of discrete and continuous columns simultaneously [34].

$$f(x)_i = \frac{e^{x_i}}{\sum_{j=1}^K e^{x_j}} \quad (4)$$

Tanh activation function is shown in Eq. (5)

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (5)$$

CTGAN workflow is listed in three steps below [10]:

- **Identification of continuous columns:**

CTGAN initially identifies each continuous column and determines the number of modes present within them. This is achieved by fitting the modes into a Gaussian mixture using the Variational Gaussian Mixture model (VGM) [35].

- **Computing probability density:**

Then, CTGAN computes the probability density for each column value in every row, thereby assessing the likelihood of each value occurring.

- **Sampling and normalization:**

Finally, CTGAN samples a mode based on the calculated

probability density and uses this sampled mode to normalize the value in the new row. This process ensures that the generated data accurately reflects the distribution observed in the original dataset.

Figure 2 visually illustrates both the imbalanced and balanced datasets, vividly depicting the skewed class distribution in the imbalanced scenario. However, after applying CTGAN, the balanced version emerges, ensuring equitable representation of both classes. This balance not only addresses the skewed distribution but also contributes to enhanced model performance by providing sufficient examples for learning from each class. Additionally, Table 2 represents the hyperparameters used in CTGAN.

h. Other Balancing Techniques

In our study, instead of solely relying on CTGAN, we adopted another approach by evaluating a widely adopted data synthesis technique: Tabular Variational Autoencoder (TVAE) [12]. TVAE represents an established method in the field for addressing class imbalance within datasets. The same mentioned classifiers were also trained on data balanced by TVAE. The results of TVAE are discussed in the next sections.

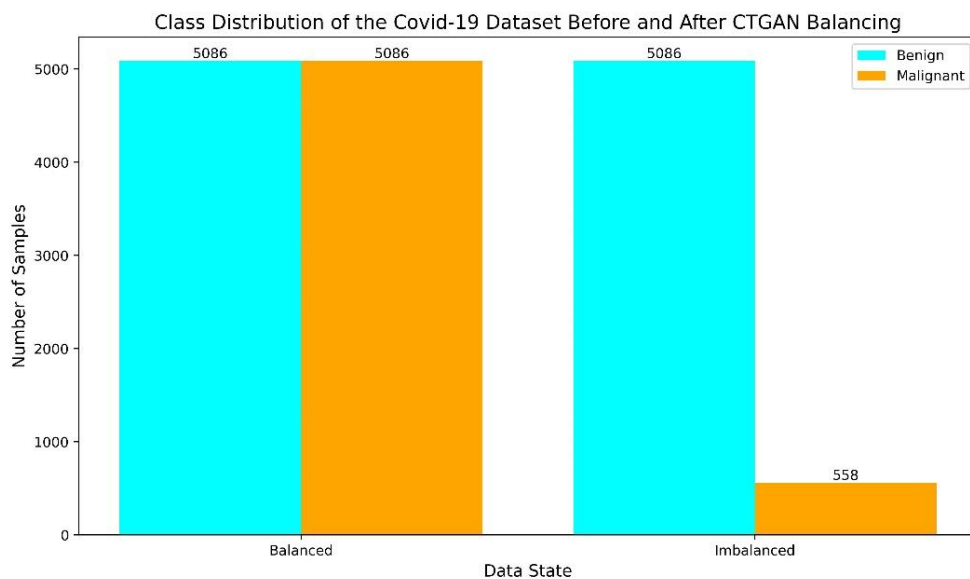
i. Tabular Variational Autoencoder

In addition to evaluating CTGAN, we conducted thorough comparative analysis by testing another widely utilized method in balancing and oversampling techniques, namely Tabular Variational Autoencoder (TVAE). Variational Autoencoders (VAEs) are advanced deep generative models widely used for generating synthetic data from real datasets. VAEs consist of two main components: The Encoder and the Decoder. The Encoder compresses the input data into a latent probability distribution, while the Decoder generates new instances based on this inferred latent space. This approach allows VAEs to learn and recreate the original input data from the learned probability

distribution. In recent research, Xu et al. [12] introduced the Tabular Variational Autoencoder (TVAE), a variant of VAE specifically tailored for tabular data. This study applied TVAE to generate synthetic COVID-19 data. The TVAE model takes real COVID data and analyzes feature variables based on their statistical and probabilistic distributions. The sensitivity of synthetically generated data is crucial in biomedical domains, especially in disease research. The TVAE model addresses this sensitivity by incorporating the Evidence Lower Bound Loss (ELBO) [36]. To illustrate the computational process, the TVAE model is represented by the following Eq. (6):

$$G(x) = T(\text{Decoder}(\text{Encoder}(x))) \quad (6)$$

Here, $G(x)$ represents the generated synthetic data of COVID-19 instances, x represents a real data instance of COVID-19 instances, and T is the function of the tabular variational autoencoder that takes x as input and generates $G(x)$. The $\text{Encoder}()$ function learns the latent distribution from real data, while the $\text{Decoder}()$ function generates synthetic data by analyzing these latent distributions. The TVAE method operates in a semi-supervised manner for the synthetic generation of COVID-19 data. The model first learns the latent space distribution of real data and then replicates this data while minimizing the loss function. This semi-supervised nature makes the TVAE model well-suited for use in the biomedical domain, particularly for generating synthetic data that accurately reflects the characteristics of real-world datasets. In this study, TVAE was used to produce synthetic samples representing both negative and positive for the COVID-19 dataset.



Figure

CTGAN-Balanced and Imbalanced versions of the Covid-19 Dataset.

Table 2. Hyperparameters of CTGAN.

Hyperparameters	Values
embedding_dim	128
generator_lr	0.0001
generator_decay	0.00001
discriminator_lr	0.00001
discriminator_decay	0.001
discriminator_steps	3
epochs	500

4. Experimental Results

This section evaluates the effectiveness of the oversampling method using various well-known machine learning metrics, including accuracy, precision, recall, F1-score, and ROC/AUC. Subsection 4.1 provides detailed explanations of these metrics. Additionally, we perform a comparative analysis between the results obtained from the balanced data approach and those from standalone models, highlighting the improvements achieved solely through dataset balancing. The outcomes for the baseline classifiers trained on the original dataset are presented in subsection 4.2. Subsections 4.3 and 4.4 detail the findings for the classifiers trained on CTGAN-balanced data and TVAE-balanced data, respectively.

a. Performance Metrics

Before delving into the evaluation metrics employed in this study, it is imperative to understand key terms such as True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). These terms are fundamental in assessing the performance of classification models. In this paper, all performance metrics are explained alongside their corresponding formulas, ensuring clarity and enabling a comprehensive understanding of the evaluation process.

- True Positive (TP): Instances correctly predicted as positive by the model.
- True Negative (TN): Instances correctly predicted as negative by the model.
- False Positive (FP): Instances incorrectly predicted as positive by the model.
- False Negative (FN): Instances incorrectly predicted as negative by the model.

b. Accuracy

Accuracy, a crucial metric in classification tasks, quantifies the proportion of correctly predicted cases, encompassing both true positives and true negatives, out of all instances. The formula for Accuracy is provided in (7)

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \quad (7)$$

c. Precision

Precision, a vital metric in classification assessment, emphasizes the quality of positive predictions by determining the proportion of correctly identified positive

instances out of all instances predicted as positive. A higher precision value signifies a lower rate of false positives. The formula for Precision is provided in (8)

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \quad (8)$$

d. Recall

Recall (9), often referred to as sensitivity or true positive rate, assesses the model's capability to identify positive cases accurately. It quantifies the proportion of actual positive instances that the model correctly predicts. A higher recall value indicates a lower rate of false negatives. The formula for Recall is expressed as follows:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \quad (9)$$

e. F1-Score

F1-Score (10), a harmonic mean of precision and recall, provides a balanced evaluation by considering both false positives and false negatives. This metric is especially valuable in scenarios where class distribution is imbalanced. By combining precision and recall, F1-Score offers a comprehensive assessment of the model's performance. The formula for F1-Score is defined as:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (10)$$

f. ROC/AUC

The trade-off between true positive rate (recall) and false positive rate is represented graphically by the ROC curve. It provides an overview of the model's overall performance across several thresholds.

g. Baseline Results: Training Results on the Original Dataset

In this subsection, Table 3 presents the results of the three classifiers trained on the original dataset. This table provides a comprehensive analysis of performance metrics, including accuracy, precision, recall, F1-score, and ROC/AUC, for each classifier. It's worth noting that these classifiers were trained on the original, pre-processed COVID-19 dataset. Among these models, LR emerges as the top performer, achieving an accuracy of 88%. Additionally, Figure 3 represents the baseline model accuracies and ROC curves, providing a comprehensive comparison of the models' performance before applying data balancing techniques.

Table 3. Performance of Classifiers, Trained on Original COVID-19 Dataset, with 80:20 Split Ratio.

Model	Label	Precision	Recall	F1-Score	Accuracy	AUC
RF	0	0.90	0.97	0.94	0.88	0.66
	1	0.67	0.35	0.46		
LR	0	0.92	0.96	0.94	0.89	0.72
	1	0.67	0.47	0.55		
DT	0	0.93	0.93	0.93	0.89	0.81
	1	0.59	0.59	0.59		

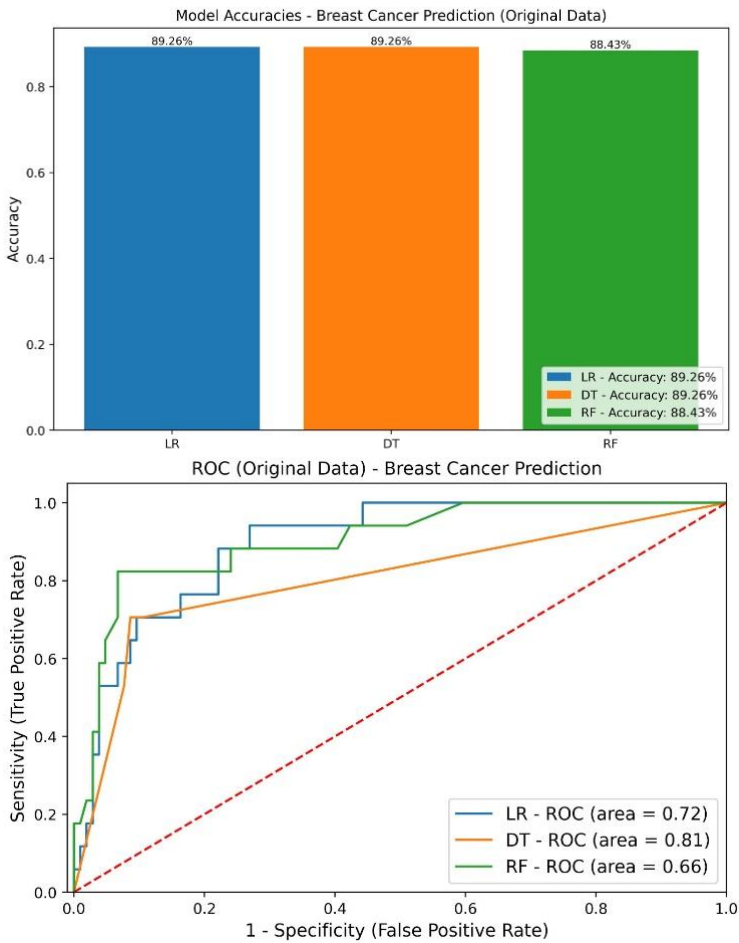


Figure 3. Accuracies and ROC Curves of Models Trained on Original Data.

Table 4. Performance of Classifiers Trained on CTGAN-Balanced Data, with 80:20 Split Ratio.

Model	Label	Precision	Recall	F1-Score	Accuracy	AUC
RF	0	0.93	0.96	0.94	0.99	0.97
	1	1.00	0.99	0.99		
LR	0	0.81	0.33	0.47	0.92	0.65
	1	0.93	0.99	0.96		
DT	0	0.91	0.83	0.86	0.97	0.90
	1	0.98	0.99	0.99		

h. CTGAN-Balanced Dataset Results

In this subsection, Table 4 presents the results of training classifiers with CTGAN-balanced data. The classifiers used are Random Forest (RF), Logistic Regression (LR), and Decision Tree (DT). The same performance metrics are calculated and reported to assess the classifiers' effectiveness. As shown in Table 4, RF trained on the CTGAN-balanced dataset is the top performer, achieving an impressive accuracy of 0.9883 and an AUC of 0.97. Additionally, other classifiers also showed improved accuracy by balancing their input data with synthetic data.

i. TVAE-Balanced Dataset Results

In this subsection, the performance of classifiers trained on TVAE-balanced data is presented in Table 5. The same set of metrics is used to evaluate the classifiers' performance. From Tables 5 and 4, it is evident that the RF trained on CTGAN-balanced data outperformed all other RF models trained on both the original data and TVAE-balanced data, achieving an accuracy of 98.83%

Table 5. Performance of Classifiers Trained on TVAE-Balanced Data, with 80:20 Split Ratio.

Model	Label	Precision	Recall	F1-Score	Accuracy	AUC
RF	0	0.92	1.00	0.96	0.94	0.90
	1	1.00	0.81	0.90		
LR	0	0.94	0.85	0.89	0.86	0.86
	1	0.74	0.88	0.80		
DT	0	0.94	0.88	0.91	0.88	0.87
	1	0.78	0.88	0.82		

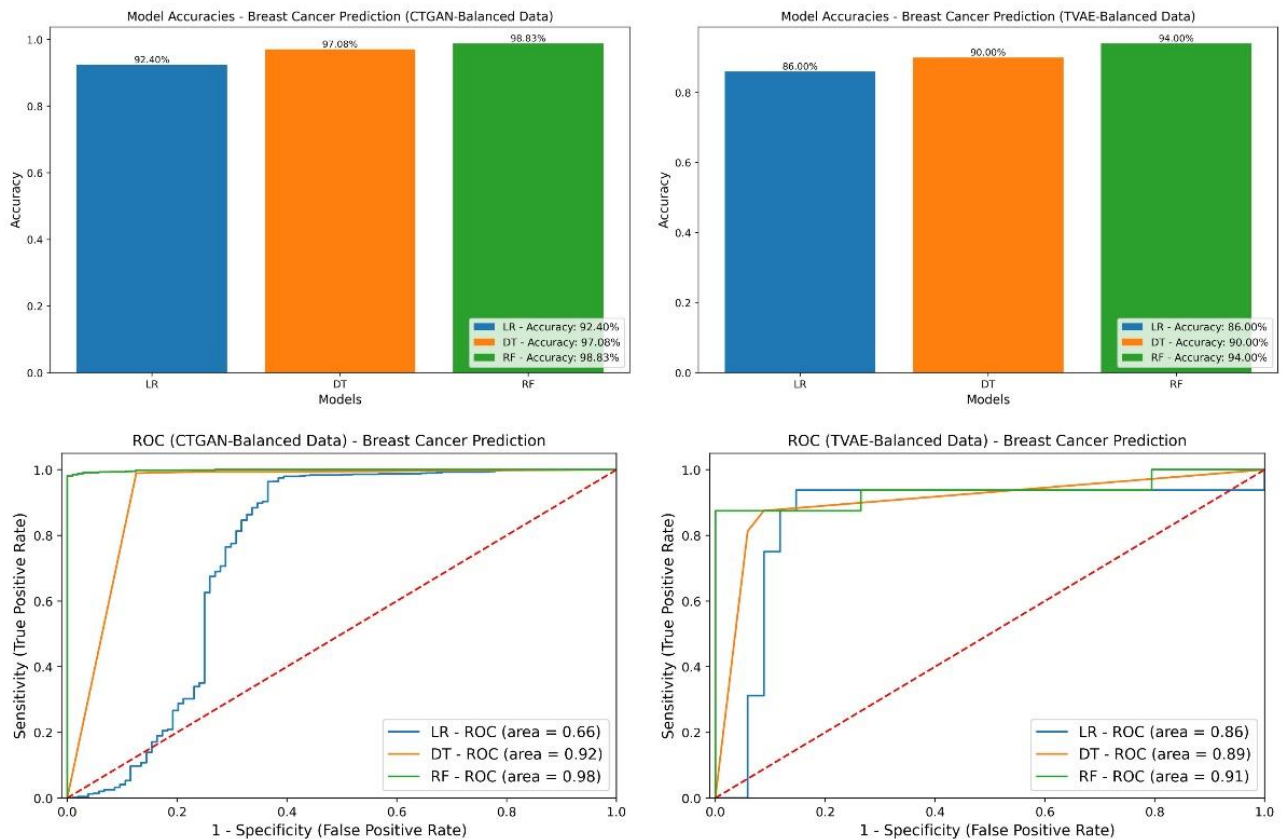


Figure 4. Accuracies and ROC Curves of Models Trained on both CTGAN and TVAE-Balanced Dataset.

Table 6. Comparison with Similar Methods in the Literature

References	Methods	Accuracy
Ahmed et al. [16]	Decision Tree, Random Forest, Neural Network (NN), Naive Bayes, Logistic Regression, and k-nearest neighbor	97.24%
Hemdan et al. [17]	VGG-19	90%
Kumar et al. [23]	Modified-VGG16, Modified-VGG19, Modified-ResNet50, and Modified-InceptionV3	98.61%
Hafeez et al. [18]	CODISC-CNN (CNN based Coronavirus Disease Prediction System for Chest X-rays)	89%
Taresh et al. [25]	Deep transfer learning algorithms including VGG16 and MobileNet	98.28%
Turlapati et al. [5]	Outlier-SMOTE, SMOTE, ADASYN	89-90%
Proposed	Balancing techniques (CTGAN, TVAE) + LR, RF, DT,	98.83%

Furthermore, Figure 4 shows the accuracies and ROC curves of models trained on both CTGAN and TVAE-balanced datasets. Notably, the RF classifier trained on the CTGAN-balanced data achieves a higher area under the curve (AUC) compared to the RF classifiers trained with other methods.

5. Discussion

The findings of this study provide significant insights that can enhance the field of COVID-19 detection. By assessing the performance of different machine learning models, specifically Random Forest (RF), Logistic Regression (LR), and Decision Tree (DT), on the Kaggle COVID-19 dataset, we add to the existing body of research and broaden the understanding of machine learning approaches for COVID-19 detection. Our results underscore the effectiveness of data synthesis and oversampling techniques, such as Conditional Tabular Generative Adversarial Network (CTGAN) and Tabular Variational Autoencoder (TVAE), in mitigating dataset imbalances. These insights are highly valuable for healthcare professionals and researchers aiming to improve the accuracy and efficiency of COVID-19 detection and diagnosis.

a. Comparative Analysis with other Studies

In this section, a comparative analysis of our study's findings with those reported in previous research are presented in Table 6.

b. Balanced Dataset Effects on Dataset

Imbalanced datasets have a substantial effect on the performance of machine learning techniques. When data is skewed, ML algorithms tend to favor the majority classes, overlooking those with fewer instances. This bias can

undermine the overall performance and quality of the ML model [37]. The dataset used in this study (COVID-19) is heavily imbalanced, with a ratio of 1:9, intensifying the associated challenges. In the field of machine learning, addressing class imbalance has led to the development of various data oversampling and under sampling techniques, such as SMOTE [38] (Synthetic Minority Oversampling Technique), RUS [39] (Random Under-Sampling), and ROS [40] (Random Over-Sampling). However, recent research [35] indicates that Conditional Tabular Generative Adversarial Network (CTGAN) [35] outperforms these traditional methods in data synthesis [12,41]. As demonstrated in our study, employing CTGAN-based data oversampling yields superior results. When utilizing the same classifiers—Random Forest (RF), Logistic Regression (LR), and Decision Tree (DT)—trained on the CTGAN-balanced dataset, we observed an overall enhancement of 5% to 10% in the mentioned evaluation metrics. Studies like current research on pandemics like COVID-19 can be instrumental in enhancing future preparedness and containment strategies. By focusing on early detection, surveillance, and monitoring of potential pathogens in both animal and human populations, promoting wildlife and ecosystem protection, implementing biosecurity measures, and strengthening healthcare infrastructure, we can improve our ability to prevent, detect, and respond to future pandemics effectively.

6. Conclusion and Future Work

The COVID-19 pandemic ravaged the globe, and overwhelmed healthcare systems on a mass scale, exacting an enormous number of lives. Prompt and precise diagnostic approaches have been a critical need. In our study, we evaluated three machine learning models—

Random Forest (RF), Logistic Regression (LR), and Decision Tree (DT)—for detecting COVID-19 using preprocessed datasets. The dataset used in this study is heavily imbalanced, with 5086 negative and 558 positive cases, posing a significant challenge for effective model training. Therefore, we demonstrated the capability of two advanced data synthesis algorithms, Conditional Tabular Generative Adversarial Network (CTGAN) and Tabular Variational Autoencoder (TVAE), in addressing the class imbalance inherent in the dataset. The performance of the models trained on both the original and balanced datasets was then compared. Our findings reveal that RF obtains the highest accuracy of 98.83% on the CTGAN-balanced dataset. We believe that exploiting data synthesis along with classical machine learning approaches holds promise for enhancing the accuracy of COVID-19 diagnosis. This approach could be particularly beneficial in resource-limited settings and developing countries. Moving forward, we recommend the adoption of balanced datasets in training high-performance systems to support effective pandemic response.

Scope for future research lies in incorporating multi-modal data sources, such as merging chest X-ray images with demographic information, comorbidities, and laboratory investigations. Prediction of COVID-19 diagnosis with diverse clinical attributes would only further strengthen the diagnostic model. Also, explainable machine learning endeavors should be initiated to understand the black-box AI decision. Investigation of the approach of model explainability techniques and feature importance methods may help interpret the model decisions in a clinical context, ensuring patient safety, privacy, comfort and their rights are held and protected.

Data Availability

The dataset used in this study is the COVID-19 dataset, which is a publicly available dataset at this link: <https://www.kaggle.com/datasets/einsteindata4u/COVID19>.

7. Funding

This research has not received any external funding.

Conflict of interest: The writers do not have any relevant conflicts of interest to disclose about the subject matter of this work.

Ethical Approval: Not applicable

Consent to Participate: Not applicable.

Consent to Publish: Not applicable.

8. References

- [1] *Naming the coronavirus disease (COVID-19) and the virus that causes it*, World Health Organization, 2020.
- [2] S. Hamal et al. and R. M. Gibson. (2024, Jun.). *A comparative analysis of machine learning algorithms for detecting COVID-19 using lung X-ray images*. Decision Analytics Journal. [Online]. 11, p. 100460. Available: 10.1016/j.dajour.2024.100460
- [3] Debata, B., P. Patnaik, and A. Mishra. (2020, Sep.). *COVID-19 pandemic! It's impact on people, economy, and environment*. Journal of public affairs. [Online]. 20(4), p. e2372. Available: 10.1002/pa.2372
- [4] Baker, C.A. and K.E. Gibson. (2022, Oct.). *Persistence of SARS-CoV-2 on surfaces and relevance to the food industry*. Current Opinion in Food Science. [Online]. 47, p. 100875. Available: 10.1016/j.cofs.2022.100875
- [5] Turlapati, V.P.K. and M.R. Prusty. (2020, Dec.). *Outlier-SMOTE: A refined oversampling technique for improved detection of COVID-19*. Intelligence-Based Medicine. [Online]. 3-4, p. 100023. Available: 10.1016/j.ibmed.2020.100023
- [6] *14.9 million excess deaths associated with the COVID-19 pandemic in 2020 and 2021*, WHO, 2022.
- [7] B. Amma. (2024, Mar.). *En-RfRsK: An ensemble machine learning technique for prognostication of diabetes mellitus*. Egyptian Informatics Journal. [Online]. 25, p. 100441. Available: 10.1016/j.eij.2024.100441
- [8] S. El-Sappagh et al. and H. Saleh. (2021, Sep.). *The role of medication data to enhance the prediction of Alzheimer's progression using machine learning*. Computational Intelligence and Neuroscience. [Online]. (1), 2021. Available: 10.1155/2021/8439655
- [9] M. A. Islam et al. and S. Jannaty. (2024, Jun.). *Precision Healthcare: A Deep Dive into Machine Learning Algorithms and Feature Selection Strategies for Accurate Heart Disease Prediction*. Computers in Biology and Medicine. [Online]. 176, p. 108432. Available: 10.1016/j.compbimed.2024.108432
- [10] M. R. A. B. Soflaei, A. Salehpour and K. Samadzamini. (2024, Apr.). *Enhancing network intrusion detection: a dual-ensemble approach with CTGAN-balanced data and weak classifiers*. The Journal of Supercomputing. [Online]. 80, pp. 16301-16333. Available: 10.1007/s11227-024-06108-7
- [11] A. Luque et al. and A. D. L. Heras. (2019, Jul.). *The impact of class imbalance in classification performance metrics based on the binary confusion matrix*. Pattern Recognition. [Online]. 91, pp. 216-231. Available: 10.1016/j.patcog.2019.02.023
- [12] L. Xu et al. and K. Veeramachaneni, "Modeling tabular data using conditional gan," Advances in neural information processing systems, 2019.
- [13] C. C. Doodoo et al. and J. Mensah. (2024, Jun.). *Using machine learning algorithms to predict COVID-19 vaccine uptake: A year after the introduction of COVID-19 vaccines in Ghana*. Vaccine: X. [Online]. 18, p. 100466. Available: 10.1016/j.jvacx.2024.100466
- [14] Oyewola, D.O., E.G. Dada, and S. Misra. (2022, Nov.). *Machine learning for optimizing daily COVID-19 vaccine dissemination to combat the pandemic*. Health and Technology. [Online]. 12, pp. 1277-1293. Available: 10.1007/s12553-022-00712-4

- [15] S. M. I. Osman and A. Sabit. (2022, Dec.). *Predictors of COVID-19 vaccination rate in USA: A machine learning approach*. Machine Learning with Applications. [Online]. 10, p. 100408. Available: 10.1016/j.mlwa.2022.100408
- [16] M. A. Ahmed et al. and M. Hamid Ali, "Automatic COVID-19 pneumonia diagnosis from x-ray lung image: A Deep Feature and Machine Learning Solution," Journal of Physics: Conference Series, 2021, p. 012099.
- [17] E. E. D. Hemdan, M. A. Shouman, and M. E. Karar, "Covidx-net: A framework of deep learning classifiers to diagnose covid-19 in x-ray images," arXiv preprint arXiv:2003.11055, 2020.
- [18] U. Hafeez et al. and H. A. Madni. (2022, Feb.). *A CNN based coronavirus disease prediction system for chest X-rays*. Journal of Ambient Intelligence and Humanized Computing. [Online]. 14(10), pp. 13179-13193. Available: 10.1007/s12652-022-03775-3
- [19] H. Mohammad-Rahimi et al. and S. Ghafouri-Fard. (2021, Mar.). *Application of machine learning in diagnosis of COVID-19 through X-ray and CT images: a scoping review*. Frontiers in cardiovascular medicine. [Online]. 8, p. 638011. Available: 10.3389/fcvm.2021.638011
- [20] M. D. Broda et al. and A. West. (2024, Jul.). *Understanding COVID-19 infection among people with intellectual and developmental disabilities using machine learning*. Disability and Health Journal. [Online]. 17(3), p. 101607. Available: 10.1016/j.dhjo.2024.101607
- [21] Y. Fu et al. and S. Wang. (2024, May.). *Using machine learning algorithms based on patient admission laboratory parameters to predict adverse outcomes in COVID-19 patients*. Heliyon. [Online]. 10(9): p. e29981. Available: 10.1016/j.heliyon.2024.e29981
- [22] K. Halteh et al. and K. Kumar. (2024, Mar.). *Using machine learning techniques to assess the financial impact of the COVID-19 pandemic on the global aviation industry*. Transportation Research Interdisciplinary Perspectives. [Online]. 24, p. 101043. Available: 10.1016/j.trip.2024.101043
- [23] V. Kumar et al. and O. Cheikhrouhou. (2022, Apr.). *COV-DLS: prediction of COVID-19 from X-rays using enhanced deep transfer learning techniques*. Journal of Healthcare Engineering. [Online]. (1), p. 6216273. Available: 10.1155/2022/6216273
- [24] A. Chen et al. and J. Chan, "Detecting Covid-19 in Chest X-Rays using Transfer Learning with VGG16," CSBio '20: Proceedings of the Eleventh International Conference on Computational Systems-Biology and Bioinformatics, Association for Computing Machinery: Bangkok, Thailand, 2020, pp. 93–96.
- [25] M. M. Taresh et al. and M. L. Mutar. (2021, May.). *Transfer Learning to Detect COVID-19 Automatically from X-Ray Images Using Convolutional Neural Networks*. International Journal of Biomedical Imaging. [Online]. 2021(1), p. 8828404. Available: 10.1155/2021/8828404
- [26] F. Pedregosa. (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research. [Online]. 12, pp. 2825-2830. Available: cir.nii.ac.jp
- [27] H. I. Sarker. (2021, Mar.). *Machine Learning: Algorithms, Real-World Applications and Research Directions*. SN Computer Science. [Online]. 2(3), p. 160. Available: 10.1007/s42979-021-00592-x
- [28] M. Alloghani et al. and A. J. Aljaaf. (2019, Sep.). *A Systematic Review on Supervised and Unsupervised Machine Learning Algorithms for Data Science*. Supervised and Unsupervised Learning for Data Science. [Online]. pp. 3-21.
- [29] G. Louppe, "Understanding random forests: From theory to practice," arXiv preprint arXiv:1407.7502, 2014.
- [30] Y. Belsti et al. and H. Teede. (2023, Nov.). *Comparison of machine learning and conventional logistic regression-based prediction models for gestational diabetes in an ethnically diverse population; the Monash GDM Machine learning model*. International Journal of Medical Informatics. [Online]. 179, p. 105228. Available: 10.1016/j.ijmedinf.2023.105228
- [31] P. Rajendra and S. Latifi. (2021, Jan.). *Prediction of diabetes using logistic regression and ensemble techniques*. Computer Methods and Programs in Biomedicine Update. [Online]. 1, p. 100032. Available: 10.1016/j.cmpbup.2021.100032
- [32] M. M. Chowdhury, R. S. Ayon and M. S. Hossain. (2024, Jun.). *An investigation of machine learning algorithms and data augmentation techniques for diabetes diagnosis using class imbalanced BRFS dataset*. Healthcare Analytics. [Online]. 5, p. 100297. Available: 10.1016/j.health.2023.100297
- [33] I. Goodfellow et al. and Y. Bengio, "Generative adversarial nets," Advances in neural information processing systems, 2014.
- [34] M. S. K. Inan, S. Hossain and M. N. Uddin. (2023, Jan.). *Data augmentation guided breast cancer diagnosis and prognosis using an integrated deep-generative framework based on breast tumor's morphological information*. Informatics in Medicine Unlocked. [Online]. 37, p. 101171. Available: 10.1016/j.imu.2023.101171
- [35] S. Bourou et al. and T. Zahariadis. (2021, Sep.). *A Review of Tabular Data Synthesis Using GANs on an IDS Dataset*. Information. [Online]. 12(9), p. 375. Available: 10.3390/info12090375.
- [36] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," arXiv preprint arXiv:1312.6114, 2013.
- [37] V. W. D. Vargas et al. and J. L. V. Barbosa. (2022, Nov.). *Imbalanced data preprocessing techniques for machine learning: a systematic mapping study*. Knowledge and Information Systems. [Online]. 65(1), pp. 31-57. Available: 10.1007/s10115-022-01772-8
- [38] N. V. Chawla et al. W. P. Kegelmeyer. (2002, Jun.). *SMOTE: synthetic minority over-sampling technique*.

- Journal of artificial intelligence research. [Online]. 16, pp. 321-357. Available: [10.1613/jair.953](https://doi.org/10.1613/jair.953)
- [39] T. Hasanin, T. Khoshgoftaar, "*The Effects of Random Undersampling with Simulated Class Imbalance for Big Data*," *IEEE International Conference on Information Reuse and Integration (IRI)*, Lake City, UT, USA, 2018, pp. 70-79.
- [40] *Imbalanced learn*. Available: imbalanced-learn.org
- [41] I. Ashrapov, "Tabular GANs for uneven distribution," arXiv preprint [arXiv:2010.00638](https://arxiv.org/abs/2010.00638), 2020.
-



**Ferdowsi
University of
Mashhad**

Journal of Computer and Knowledge Engineering


<https://cke.um.ac.ir>



**Information and
Communication
Technology Association of
Iran**

Anomaly Detection in IoMT Environment Based on Machine Learning: An Overview*

Research Article

Peyman Vafadoost Sabzevar¹, Hamidreza Rokhsati² , Alireza Chamansara³, Ahmad Hajipour⁴

DOI: [10.22067/cke.2024.89572.1127](https://doi.org/10.22067/cke.2024.89572.1127)

Abstract In today's era, the Internet of Things has become one of the important pillars in organizations, hospitals, and research circles and is recognized as an integral part of the Internet. One of the important areas that require online monitoring is medical imaging equipment, whose functional information is transmitted through the Internet of Things. Server security and intrusion prevention, along with anomaly detection, are critical requirements for these networks. The purpose of anomaly detection is to develop methods that can detect attackers' attacks and prevent them from happening again. Algorithms and methods based on statistics play an important role in predicting and diagnosing anomalies. In this article, the isolation forest algorithm was used for training on 80% of the dataset related to the data of the Internet of Medical Things network, and then this model was tested and evaluated on the remaining 20%. The results show 90.54% accuracy in detecting anomalies in the received data, which confirms the effective performance of this method in this field.

Keywords *Index Terms— Anomaly, Detection Anomaly, Internet of Medical Things, Isolation Forest, Unsupervised Learning.*

1. Introduction

The Internet of Things (IoT) has become one of the most prominent and transformative phenomena in the world of technology and communication since the beginning of 2000. This concept has widely been able to create a significant improvement in the way of communication and interaction between different objects and devices and thus, improve efficiency, efficiency, and communication in various systems. The IoT, which refers to the communication and interaction between various types of objects including sensors, home devices, and sophisticated gadgets, allows us to collect, send, and receive information to facilitate various processes in various societies and industries such as improve agriculture, healthcare, and

smart homes [1].

Currently, one of the fields that has been deeply affected by the Internet of Things is the field of medicine and health. The Internet of Medical Things (IoMT), which specifically relates to the use of sensors, wearable devices, smart medical devices, and other advanced technologies, has been able to help improve disease monitoring and management and improve the quality of healthcare services. This technology not only helps in early diagnosis of diseases and better management of patients' condition, but also generally increases the quality and accuracy of health services.

Also, the rapid growth of IoT devices, which includes smart home appliances and various industrial technologies, has led to increased communication between objects and improved efficiency and flexibility in various industries and daily life. Statistics and surveys show that the connection of devices to the IoT is increasing dramatically [2]. This rapid growth, shown in Table 1, shows the importance and necessity of paying attention to security and management issues in this area, because with the increase in the number of connected devices, there is a need for effective solutions to maintain security and data quality.

Table 1 Internet Use and World Population [2].

	2010	2015	2020
World Population	6.8 billion	7.2 billion	7.6 billion
Connected Devices	12.5 billion	25 billion	50 billion
Device for Each Person	1.84	3.47	6.58

* Manuscript received 2024 July 15, Revised, 2024 October 4, accepted 2024 November 6.

¹ Master of Biomedical Engineering, Electrical and Computer Engineering Department, Hakim Sabzevari University, Sabzevar, Iran.

² Corresponding Author. Master of Engineering in Computer Science, Department of Computer, Control and Management Engineering, Sapienza University of Rome, Rome, Italy. **Email:** rokhsati.1960699@studenti.uniroma1.it

³ PhD in Biomedical Engineering, Department of Biomedical Engineering, Materials and Energy Research Center, Tehran, Iran.

⁴ Assistant Professor, Electrical and Computer Engineering Department, Hakim Sabzevari University, Sabzevar, Iran.

Data available on the internet, especially medical data, are of particular importance. Medical data includes sensitive and vital information such as disease history, test results, and medical center device parameters. Breach of security or abnormality in this data can lead to serious problems, including unauthorized access to information, misuse, malfunction of monitoring systems, and even illegal sale of information. These issues can have serious consequences for the privacy and security of healthcare facilities and potentially affect public trust in health and medical systems [3, 4]. Figure 1 shows the IoMT architecture. This architecture includes various layers, including sensors and IoMT devices layer, network and communication layer, and data processing and analysis layer, which are connected to each other. This layered structure enables the communication and exchange of information between different IoMT devices and enables the processing and analysis of collected data.

Hence, the field of IoMT has become a complex and fully integrated ecosystem that includes interconnected medical devices and systems. This ecosystem is specifically designed to enhance the quality of patient care, optimize healthcare practices, and ultimately improve overall health outcomes [5, 6]. Considering the high importance of medical data and the complexities related to their security, it is essential to develop and implement effective solutions to protect these data and ensure their accuracy and security.

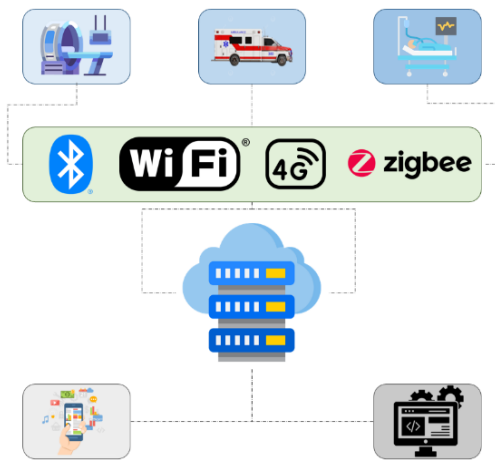


Figure 1. IoMT Reference Architecture [7].

Due to the emerging phenomenon of the IoT and its combination with new methods, the activities and researches conducted in this field are expanding rapidly. This innovative combination has not only increased researchers' interest in studying and developing the IoT, but also led to significant improvements in various applications of this technology, especially in medical fields.

Evidence shows that the number of scientific articles published in this field is increasing day by day. In particular, articles published in prestigious journals such as ScienceDirect show a significant growth as shown in Figure 2 and Figure 3, and pay a lot of attention to topics

related to IoT and IoMT. This growth shows that researchers are seriously investigating and developing this technology, working on its various aspects from improving the security and efficiency of systems to expanding new applications in clinical and health care.

The quick adoption of the IoMT, in the healthcare industry has transformed how patients are cared for. Has also brought about security hurdles to overcome. The application of machine learning (ML) for anomaly detection shows promise in recognizing and addressing these risks. This article gives an outline of existing ML methods, for detecting anomalies in IoMT settings. Emphasizes how they could bolster the security and dependability of networks. Our goal is to help tackle these obstacles and aid in the advancement of robust IoMT systems.

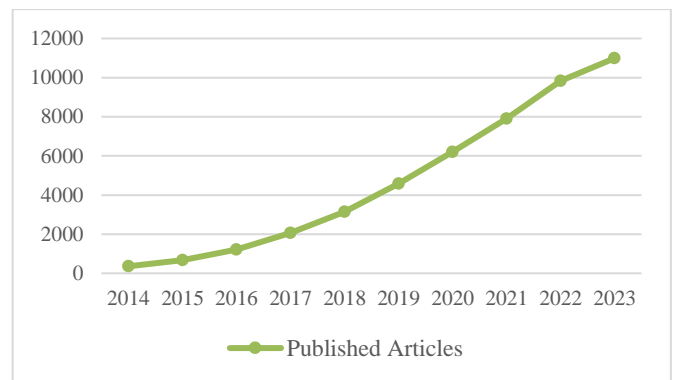


Figure 2. The Number of Articles Published in The Field of IoT

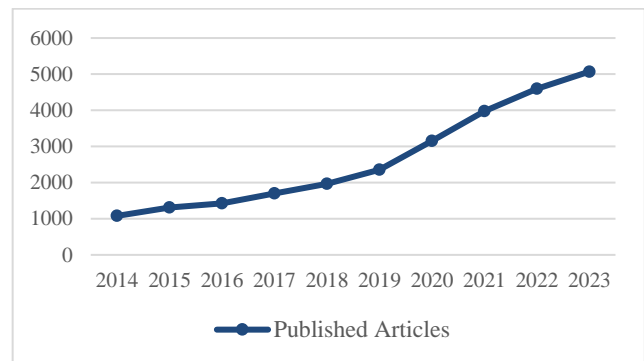


Figure 3. The Number of Articles Published in The Field of IoMT

A. IoT security challenges

In the field of IoT, due to its complex nature and architecture, this technology has many security challenges, each of which can create serious risks for the systems. Developers of this technology should avoid these damages by using different techniques. One of the most important security challenges in this field is the wireless nature of IoT networks [8]. This feature exposes the network to the risk that it can gain access to sensitive data and compromise or alter it. Another important security challenge is network topology. Networks can compromise the entire system by sending fake data to the network and

disrupt its core functionality. This type of operation can have serious consequences, especially in sensitive environments such as medicine and healthcare, because in these systems it can contribute to life-threatening injuries and serious problems for patients. In addition, data heterogeneity is one of the other main challenges of the IoT. Data generated by different sources, which are created separately and with different patterns, make the management and analysis of this data more complex. This diversity in data can cause IoT systems to face more challenges in maintaining security and integrity [9, 10].

Due to these challenges, the need to develop multilayered and comprehensive IoT security in and implement these systems to the users to ensure their integrity, security, and efficiency against various threats. These predictions are especially important in fields such as medicine that deals with human lives [11].

In this regard, exploiting vulnerabilities in IoT-based networks can lead to specific attacks that put these networks at risk. For example, the CyberMDX report [12] shows that nearly half of IoMT devices are vulnerable to anomalies. These vulnerabilities can lead to malicious attacks that not only disrupt the performance of systems, but can also have serious consequences for the health of patients.

IoMT systems are fundamentally different from other systems because they directly affect patients' lives. Serious privacy concerns arise if patients' identities and sensitive information are exposed. Additionally, the high value of healthcare data on the black market adds to these concerns. In particular, the average cost of healthcare-related data is approximately 50 times higher than credit card information, making this information a very valuable target for attackers [13].

These conditions show that providing security in IoMT systems is not only necessary for data protection and privacy, but also for protecting patients' lives and preventing financial abuses. Therefore, paying special attention to the security of these systems and implementing advanced solutions to deal with possible threats is an inevitable necessity [14].

Based on these cases, first the researches and works done in this field and similar methods are discussed. Then, in the next section, the research methodology, the algorithms used, and the available data are examined in detail. In the fourth part, the results obtained from the analyzes and experiments will be evaluated and analyzed. Finally, conclusion will be presented and the achievements of this research will be comprehensively discussed.

2. Literature review

So far, a lot of research has been done in the field of anomaly detection in order to prevent cyber-attacks in this field. These researches have developed advanced algorithms and models that are able to identify unusual behaviors and prevent security threats in IoT systems and especially IoMT.

Zachos et al. [15] propose an efficient and effective anomaly-based intrusion detection (AIDS) system for IoMT networks. The proposed idea aims to use host-based and network-based techniques to reliably collect log files from IoMT and gateway devices, as well as traffic from

the IoMT edge network, while considering the computational cost. The proposed idea relies on ML techniques to detect anomalies in the collected data and thereby detect malicious events in the IoMT network, taking into account the computation overhead. Tamara et al. [16] addresses this issue by building a Kalman filter and Cauchy clustering algorithm for anomaly detection and applying them to authenticate nodes in IoMT using the Extreme ML classifier. For anomaly detection, a critical aspect in various applications including security, healthcare, and network monitoring, Alsaman [17] has introduced FusionNet, an innovative ensemble model that leverages the strengths of multiple ML algorithms, namely random forest, K-nearest neighbor (KNN), It combines support vector machine (SVM) and multilayer perceptron (MLP) for advanced anomaly detection. The FusionNet architecture uses a variety of these algorithms to achieve high precision and accuracy. And the evaluation results on two sets of data show the high accuracy of this architecture compared to classical methods.

Wang et al. [18] addressed that IoMT devices lack security authentication mechanisms and trust between these devices is highly dependent on centralized third-party services. To solve this problem, they used blockchain technology as a secure interactive environment for IoMT. However, security issues are also increasingly raised in the IoMT-Blockchain environment. Cyber-attacks targeting IoMT-Blockchain not only compromise the security of IoT devices, but can also seriously affect the overall security of the Internet. Therefore, abnormal traffic detection becomes especially important in the IoMT-Blockchain environment. In their proposed method, an abnormal traffic detection model using deep neural network is designed to detect abnormal traffic in the IoMT-Blockchain environment. Their proposed algorithm uses Residual Learning to perform more optimally in identifying abnormal traffic.

On the other hand, Wagan et al. [19] used a variety of customized security tools and frameworks to counter several common attacks, including botnet-based distributed denial-of-service (DDoS) attacks and zero-day network attacks. They proposed a new approach called Duo-Secure IoMT, which uses data from multimodal sensory signals to distinguish between attack patterns and routine data from IoMT devices. Their proposed model employs a combination of two techniques including C-Means dynamic fuzzy clustering and a customized version of the Bi-LSTM technique. This approach securely processes sensory medical data and detects attack patterns in the IoMT network.

In their research, Awotunde and his colleagues [20] used a model based on swarm neural networks to detect intruders in IoMT data-driven systems. Their proposed model is capable of detecting intruders during data transmission and can provide efficient and accurate analysis of healthcare data at the edge of the network. The performance of this system was tested using the NF-ToN-IoT real-time dataset, which is designed for IoT applications and includes telemetry data, operating systems and network data. The results of these tests showed the high accuracy of the model in identifying anomalies in the system.

In another research, Kumar et al. [21] proposed an approach to combat cloud architecture-based cyber-attacks in IoMT networks based on the 2018 Ransomware cyber-attack on the Indiana hospital system. Their method is a combination of three classification algorithms: decision tree, naive bayes, and random forest, which is used to identify anomalies. In the next step, the results obtained from the classifications are processed by the XGBoost algorithm to accurately identify normal samples and samples under attack. This multilayered combination of machine learning techniques helps to improve the accuracy and efficiency of anomaly detection in IoMT networks.

Al-Qatf et al. [34] used the KDDTrain dataset and using a hybrid approach including deep learning and SVM to detect infiltration in the network. Their proposed method has provided higher accuracy in intrusion detection compared to the classical SVM method. By introducing an intelligent intrusion detection system based on automatic encoder (AE), Ieracitano et al. [35] have presented a new statistical analysis. The salient feature of their proposed method is the use of data analysis for more optimal and robust feature extraction, which improves the accuracy and efficiency of the intrusion detection system. Javid et al. [36] have also proposed a deep learning approach for developing an intrusion and anomaly detection system. In this method, Self-Taught Learning (STL), which is an advanced technique in the field of deep learning, is used on the NSL-KDD dataset.

The motivation for writing this article is to address the growing concerns that have formed around cyber attacks on healthcare networks. Considering the high sensitivity of medical data and their vital importance in providing services to patients, any security flaw or cyber attack can have irreparable consequences for patients and health organizations. This issue not only threatens the security of information, but can lead to the disclosure of sensitive data, interruption of medical services, and even endangering the lives of patients. Therefore, investigating and identifying anomalies in the traffic data of these networks is very important in order to prevent such attacks.

Focusing on these challenges, this article tries to identify and analyze anomalies in the traffic data of healthcare networks using statistical methods. The proposed system in this paper specifically addresses one of the important problems in IoMT data security. These data are always exposed to security threats due to their high importance in controlling and monitoring the condition of patients and medical equipment. Threats that, if not detected in time, can disrupt the performance of health systems and cause irreparable damage.

For this reason, the proposed system not only identifies anomalies in traffic data, but also tries to help improve security and reduce risks associated with these threats by providing advanced solutions. Using advanced statistical techniques, this system is able to identify anomalies that may indicate a cyber-attack with acceptable accuracy. This identification allows organizations to take the necessary steps to prevent potential attacks and maintain the security of medical data. Finally, this article, with a detailed review and comprehensive analysis, not only solves one of the

basic problems in IoMT data security, but also helps to provide solutions for the overall improvement of security in healthcare networks. In this study, isolated forest machine learning algorithm was used. This algorithm, unlike other machine learning methods, categorizes data based on the detection and isolation of abnormal data. This feature makes the isolated forest algorithm work faster and more accurately than other methods and has better efficiency in isolating and identifying anomalies.

The following topics will be covered in the following articles:

- First, the NSL-KDD dataset is examined.
- Then, the proposed algorithm of this research will be introduced.
- After that, the results of this research are analyzed.
- then this method is compared with other existing methods.
- At the end, the conclusion of the article will be presented.

3. Methodology

In this section, we begin by providing a detailed description of the dataset utilized in this research. This includes an overview of its structure, the types of data it contains, and the relevance of these data points to our study. Understanding the dataset is crucial as it forms the foundation upon which our analysis and anomaly detection efforts are built. Following this, we delve into the theoretical framework of the Isolation Forest algorithm. This involves a comprehensive examination of its initial formulation, including the key mathematical principles and assumptions that underpin the model. We will explore how the Isolation Forest algorithm identifies anomalies, its unique approach to data partitioning, and why it is particularly well-suited for detecting outliers in large datasets. By first establishing a clear understanding of the dataset and then the methodology, this section sets the stage for the subsequent analysis and findings presented in this research.

A. NSL-KDD Dataset

The NSL-KDD dataset is a modified and improved version of the KDD Cup 99 dataset, which is used for research purposes and the development of intrusion detection systems [22]. This dataset has become one of the main sources of research in the field of anomaly detection and network traffic analysis due to its special features, including reducing redundant data repetition and improving the quality of samples [23]. So far, many researchers have used this data set to conduct extensive research in the field of anomaly detection and have tried to develop effective intrusion detection systems using various techniques and tools. Deep analysis of this data set using different machine learning algorithms and advanced data mining methods has been done in numerous researches and their results have led to a significant improvement in identifying security threats and improving the efficiency of cyber security systems [24-30].

In this research, among the files available in NSL-KDD dataset, the KDDTrain+.TXT file has been selected as the desired dataset. This file contains the complete set of data in CSV format, and in each record there are 41 different

attributes that describe different aspects of the data flow. These features provide important information about network activities and communications. In addition, each record has a label that specifies the type of attack or designates it as normal data. Table 3 contains the names and data types of all 41 features in the NSL-KDD dataset.

These tags are critical for anomaly detection and analysis, as they help distinguish normal data from potential attacks. Table 2 in this research shows the amount of data in this dataset as well as the distribution of data related to each class, which includes normal data and different types of attacks. This table helps to better understand the composition of data and how they are scattered among different classes, and is the basis for evaluating the performance of anomaly detection models.

The data recorded in this dataset are generally divided into four main categories [23]:

- DoS (Denial of Service): This type of attack is designed to occupy the victim's system resources by sending a large volume of fake requests. This large volume of requests makes the system unable to provide services to real users and as a result the service is disrupted. DoS attacks can be carried out in different ways, but the ultimate goal of all of them is to disable the target system.
- Probing: The goal of this attack is to probe and examine the victim's system to discover vulnerabilities and sensitive information such as open ports, running services, and connection duration. By using the information obtained from these scans, the attacker can make a more detailed plan for his next attacks. This type of attack allows the attacker to gain sufficient knowledge of the target environment before performing more serious attacks.
- U2R (User to Root): In this type of attack, the attacker enters the victim's system through a normal user account and then tries to gain higher level privileges such as root or administrator privileges by exploiting existing vulnerabilities. This type of attack allows the attacker to take full control of the system and execute commands as an elevated user.
- R2L (Remote to Local): This type of attack involves unauthorized remote access to the victim's system. The attacker enters the victim's system by guessing or breaking the password, and after gaining access, he tries to carry out destructive operations or steal information. This attack usually involves logging into the system through the network without the need for the attacker to be physically present at the victim's location.

Table 2 Details of The Normal and Anomalous Data in the KDDTrain+.TXT Dataset [23].

Dataset Type	Record	Normal Class	Dos Class	Probe Class	U2R Class	R2L Class
KDDTrain+	125,973	67,343 53.39%	45,927 36.65%	11,656 9.09%	52 0.04%	995 0.79%

Table 3 NSL-KDD Dataset Features [26].

No	Features	Type	No	Features	Type
0	duration	int64	21	is_guest_login	int64
1	protocol_type	object	22	count	int64
2	service	object	23	srv_count	int64
3	flag	object	24	serror_rate	float64
4	src_bytes	int64	25	srv_serror_rate	float64
5	dst_bytes	int64	26	rerror_rate	float64
6	land	int64	27	srv_rerror_rate	float64
7	wrong_fragment	int64	28	same_srv_rate	float64
8	urgent	int64	29	diff_srv_rate	float64
9	hot	int64	30	srv_diff_host_rate	float64
10	num_failed_logins	int64	31	dst_host_count	int64
11	logged_in	int64	32	dst_host_srv_count	int64
12	num_compromised	int64	33	dst_host_same_srv_rate	float64
13	root_shell	int64	34	dst_host_diff_srv_rate	float64
14	su_attempted	int64	35	dst_host_same_src_port_rate	float64
15	num_root	int64	36	dst_host_srv_diff_host_rate	float64
16	num_file_creations	int64	37	dst_host_serror_rate	float64
17	num_shells	int64	38	dst_host_srv_serror_rate	float64
18	num_access_files	int64	39	dst_host_rerror_rate	float64
19	num_outbound_cmds	int64	40	dst_host_srv_rerror_rate	float64
20	is_host_login	int64			

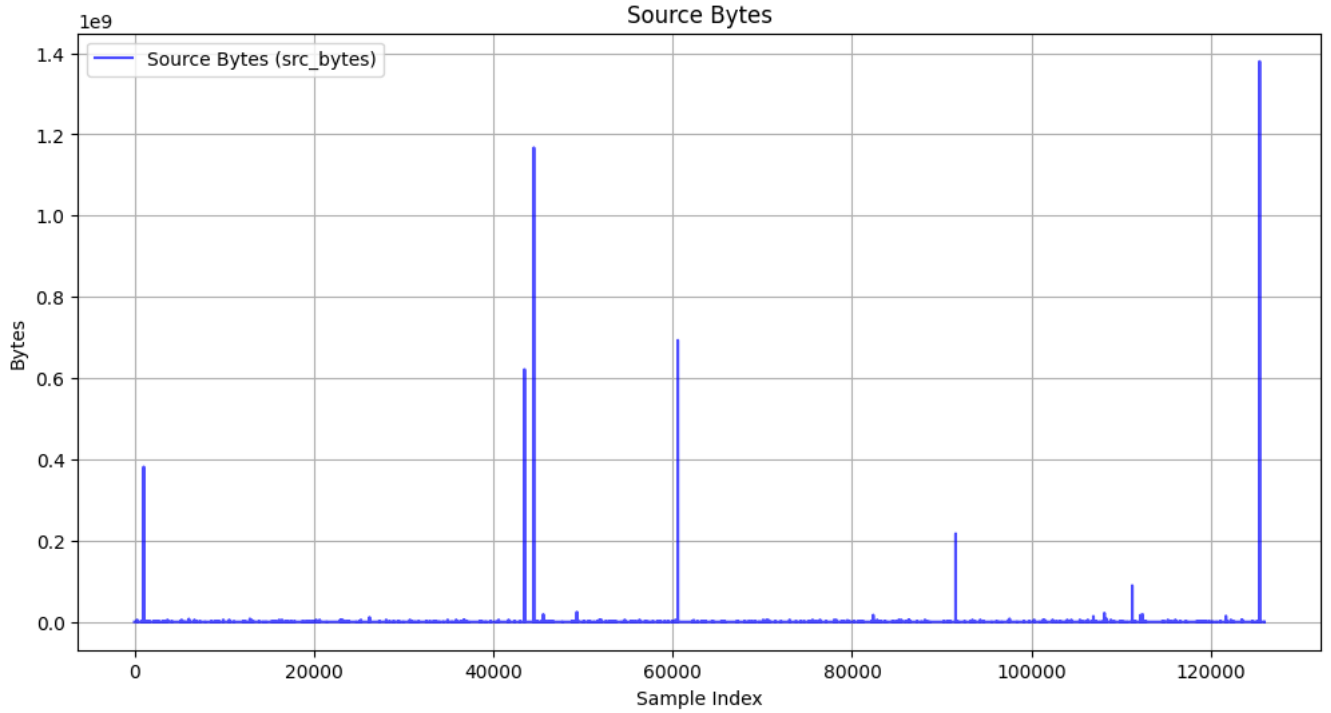


Fig 4. Data Traffic in Bytes from Source to Destination.

B. Isolation Forest

ML algorithms are divided into three main categories: supervised learning (SL), unsupervised learning (UL), and reinforcement learning (RL). Each of these categories have different uses and are designed to solve specific problems.

In SL, the model is trained using data, each of which is assigned a specific label or output. These labels indicate the correct answer or the correct classification of the data. The model learns from this labeled data how to relate inputs to the correct outputs and then uses this learning to evaluate and test on new data. UL is used where the data is not labeled. In this method, algorithms try to divide the data into different clusters or categories without having a specific output. This clustering is done based on various criteria and parameters that algorithms use to identify patterns and relationships in the data. Finally, RL is based on the trial and error of an agent in the environment. In this method, by performing various actions in the environment and receiving feedback (reward or punishment) from these actions, the agent gradually learns to adopt the best policies to achieve its goals. This type of learning is especially effective in situations where decisions are made in successive stages and long-term results are important [30].

The algorithm used in this research is called Isolation Forest as a branch of UL that is used to detect anomalies and patterns that have different characteristics from normal examples. Detecting abnormalities in this particular instance could mean an attacker could launch an attack on medical and healthcare systems that would threaten people's health [31].

Most ML-based methods for anomaly detection first train the model using normal samples. Then, using this trained model, they evaluate and test the new data. In this process, data that do not conform to normal patterns are

identified as anomalies. However, using these methods has two major problems [32, 33].

The first problem is that the anomaly detection may not achieve the desired results and the system will face an increase in error, so that a small number of anomalies are correctly detected. This issue can lead to unfavorable performance of the system in identifying security threats. The second problem is related to the high computational complexity of some of these methods, which makes these methods limited to small data and the learning process faces serious limitations. This can greatly affect the performance of the model, especially when there is a need to process a large amount of data.

In the proposed method, an approach that separates anomalies from normal data is used. This method identifies each abnormality separately by using the tree structure, the limited amount of abnormal data and the large difference between their values and normal data. This scheme not only reduces the computational complexity, but also increases the accuracy of anomaly detection and enables the system to detect anomalies more reliably.

Unlike many algorithms that first model normal behavior and then detect anomalies, Isolation Forest directly isolates abnormal points, thereby increasing detection accuracy and reducing processing time. Its special binary structure, called iTree, allows the algorithm to detect anomalous data at shallower depths, as abnormal points are quickly isolated at lower depths. In addition, this method is insensitive to the data scale due to the reliance on relative comparisons and does not require normalization or rescaling of the data, so changes in the data scale will not significantly affect the results. The random feature of feature selection in the separation forest also provides the possibility of effective identification of anomalies in multidimensional data with a high number of

features, and there is no need to reduce the dimensions or select specific features. In addition, the simple setting of the parameters of this algorithm has made it a favorable option for fast and extensive applications and has made it popular in various fields, especially anomaly detection and data security.

In order to build the iTree, having the data set $X = \{x_1, x_2, \dots, x_n\}$, the algorithm works recursively. This process starts by randomly selecting a feature named q and a threshold value p . These random selections are used to segment the data. Segmentation is done by dividing the data into two parts: those in which the feature value q is less than or equal to p , and those in which the value q is greater than p . This segmentation process continues recursively until one of the stopping conditions is met:

The node contains only one point: in other words, the data set has shrunk so much that only one data remains.

All node data have the same values: in the sense that no further partitioning is possible because all data is the same.

When the iTree is fully grown, each point of the data set X is placed in one of the external nodes (leaves). The length of the path traveled by each point in the tree to reach a leaf is determined as a criterion.

According to this algorithm, the points with shorter path length in the tree are identified as unusual and anomalous points. This is because anomalous data tends to be separated in the early stages of segmentation and, therefore, takes a shorter path to reach the leaf. On the other hand, normal data is more likely to travel longer paths because it is grouped with more similar data. The path length of a point in the tree is denoted by $h(x_i)$. This value is the number of sides that the point x_i travels from the root node to reach the leaf node. In other words, $h(x_i)$ indicates the depth of the node where the point x_i is located.

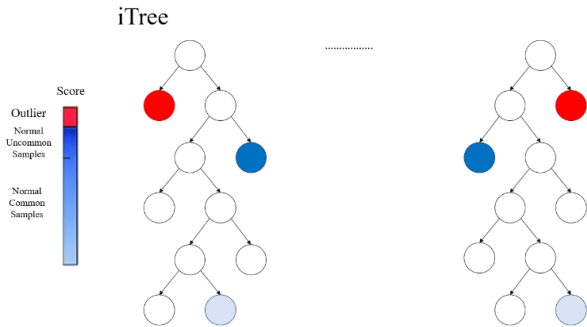


Figure 5. Isolation Forest Structure.

Abnormality detection generally consists of two main steps. In the first step, the training data set is used to build iTree trees. In this step, trees are created based on random feature partitions and threshold values to form a suitable tree structure for the data. In the second step, the test data is entered into the iTree trees, and each data sample passing through the trees is assigned an abnormality score. This abnormality score is calculated based on the length of the path that each sample takes in the tree.

To diagnose abnormality, calculating the abnormality score is a critical step. In methods that use a tree structure such as iTree, this score is calculated based on the length of the path that each data sample takes in the tree. Since the iTree has the same

structure as the binary search tree (BST), the estimation of the average path length $h(x)$ for external nodes is done in the same way as in the failed search in BST. In this analysis, the average iTree path length to terminate at external nodes is estimated using binary search tree features. This analysis, based on BST features, helps researchers estimate the average iTree path length and assign an anomaly score to each data sample based on that.

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n} \quad (1)$$

Concepts such as harmonic number $H(i)$ and average path length $c(n)$ are used in calculating the abnormality score. The harmonic number $H(i)$ is one of the important tools in these calculations and is defined as follows:

$$H(i) = \ln(i) + \gamma \quad (2)$$

γ is equivalent to Euler-Mascheroni constant.

Equations 1 and 2 can be used for normalization and estimate the abnormality score for a given sample.

$$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}} \quad (3)$$

Here $E(h(x))$ represents the average path length $h(x)$ in a set of iTree trees.. In equation (3):

- If the value of s is close to 1, it can be concluded that x is an anomalous point with high probability. This means that the point x behaves differently and unusually from other data points and is identified as an anomaly.
- On the other hand, the proximity of the value s to 0.5 indicates the norm or normality of the point. In this case, the point x has the same characteristics as other points in the dataset and is likely not anomalous.
- If for all values of a random sample, the score or value s is close to 0.5, it can be concluded that all points in the data set behave normally and there is probably no anomaly in the data set. This result shows that the data follows a stable pattern and no significant deviation from this pattern is observed.

4. Results

Before starting the clustering operation on the data, we perform two pre-processing steps so that the data is fully prepared for the clustering process. These pre-processing steps play a key role in the efficiency of the clustering algorithm, because clean and standardized data can lead to better and more meaningful results. In the first step, the protocol_type, service and flag batch features, which are part of the batch features, are converted into numbers. Also, if there are text values in each column, they are converted into numbers.

Data standardization is critical in algorithms that are sensitive to the distance or scale of features, such as distance-based algorithms. This process ensures that all features are considered equally in the analysis and predictions and that each feature has an equal impact on the final result. Standardization is calculated using the following equation:

$$X_{scaled} = \frac{X - \mu}{\sigma} \quad (4)$$

where X is the original value of the feature. μ is the mean of that feature in the data. σ is the standard deviation of that feature. Using this relationship, all features are converted to a common scale (mean zero and standard deviation one), which improves the performance of scale-sensitive algorithms.

Also, in this process, 80% of the data is considered as training data, while the remaining 20% is used as testing data. The training data is used to train the model, while the test data is used to evaluate the performance of the model. This segmentation helps us to test the model on new data that has not been trained in the process and evaluate its accuracy and efficiency in real conditions.

Also, the following relationship is used to calculate the accuracy of the model:

$$Accuracy (\%) = \left(\frac{TP + TN}{TP + TN + FP + FN} \right) \times 100 \quad (5)$$

where TP is the number of positive samples that are correctly predicted as positive. TN is the number of negative samples correctly predicted as negative. FP is the number of negative samples that are falsely predicted as positive. FN is the number of positive samples that are falsely predicted as negative.

To evaluate this method, in addition to the main criterion, three other criteria have also been used:

Precision measures the ratio of the number of correct positive predictions to the total number of positive predictions (both true and false).

$$Precision = \left(\frac{TP}{TP + FP} \right) \quad (6)$$

Recall calculates the ratio of the number of correct positive predictions to the total number of true positive samples and indicates the rate of correct identification of true positive samples.

$$Recall = \left(\frac{TP}{TP + FN} \right) \quad (7)$$

F1-Score is the harmonic mean of Precision and Recall and is generally used to evaluate models that require a balance between Precision and Recall.

$$F1 - Score = 2 \times \left(\frac{Precision \times Recall}{Precision + Recall} \right) \quad (8)$$

Table 4 depicts the performance of the model in terms of data learning and clustering. This table examines and evaluates the model based on four key parameters and also pays attention to the analysis of the relationships between these parameters. In this evaluation, all the different aspects of the model's performance have been carefully examined in order to achieve a more comprehensive understanding of the model's efficiency in identifying and classifying data.

Table IV Model Performance Results.

	Accuracy	Precision	Recall	F1-Score
Performance	90.54%	94.53%	88.51%	91.2%

Also, the table below shows the correct and incorrect predictions of the samples in the evaluation process.

Table V Model Predictions in the Evaluation Process.

	TP	FP	TN	FN
Test Data	11,097	1,614	11,714	770

In this research, we evaluated our proposed method using four key criteria and compared its results with the methods presented in various articles that worked on the same dataset. This comparison includes the accuracy of the criteria tested on this data set. The results of these evaluations show the significant and improved performance of our proposed method compared to other existing methods. In other words, our proposed model has been able to properly provide predictions in terms of accuracy and quality compared to previous methods, and this issue is well evident in the comparisons and analyzes performed.

Table VI Performance Comparison with Other Approaches.

	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
[34]	84.96	96.23	76.57	85.28
[35]	84.21	87	80.37	81.98
[36]	88.39	85.44	95.95	90.4
Suggested Method	90.54	94.53	88.51	91.2

5. Conclusion

In this article, considering the increasing importance of data in the context of IoMT, we investigated the anomalies and security threats in the traffic of this field. For this purpose, we used the NSL-KDD dataset to identify and analyze intrusions, attack risks, and the probability of failure of healthcare systems. Our proposed method consists of several main steps. First, pre-processing and normalization of the data is done in order to prepare them to enter the model. In this step, the raw data is converted to the appropriate format and scale so that the model is able to accurately analyze them. After this step, we used the cluster method called isolated forest for clustering. This method is used to identify anomalies and unusual data and is especially useful in identifying unusual patterns that may be related to security attacks. In this approach, anomalous data are presented to the model as training samples, and the model performs the clustering process based on these samples. The isolated forest model has a high ability to identify abnormal points among the data and is especially useful in complex and noisy environments such as medical network traffic. Our experimental results and its comparison with existing methods and previous studies conducted by other researchers show significant

accuracy in predicting and correct clustering of samples. These results clearly indicate the model's ability to identify abnormal network traffic patterns that deviate from the normal state, thus contributing to a more accurate analysis of security threats. However, it should be noted that the characteristics of intrusion samples may appear differently in different datasets. Therefore, to achieve optimal results, we need detailed investigations and the use of different tools to work with different data. In addition, the effectiveness of this method should be tested and evaluated in real conditions and in medical environments, including hospitals and medical centers. These tests and evaluations will help to prove the actual application and basic functionality of the model in these environments and confirm its capabilities to improve the security and efficiency of healthcare systems.

6. Authorship contribution statement

Peyman Vafadoost: basic assistance in the concept or design of the article; Implementation of article design, analysis or interpretation of article data; Alireza Chamansara: revision of the article for important intellectual content; Hamidreza Rokhsati: writing the article, validating the results of the article; Ahmad Hajipour: revision of the article for important intellectual content; validating the results of the article.

7. References

- [1] M. Chun, S. Weber and H. Tewari, "A Lightweight Encryption Scheme for IoT Devices in the Fog," *Proceedings of the Future Technologies Conference*, 2022, pp. 147-161.
- [2] Singh, Davinder, "Internet of things," *Factories of the Future: Technological Advancements in the Manufacturing Industry*, 2023, ch. 9, pp. 195-227.
- [3] M. M. Salim et al. and J. H. Park. (2021, Sep.). Homomorphic encryption based privacy-preservation for iomt. *Applied Sciences*. [Online]. 11(18), p. 8757. Available: 10.3390/app11188757
- [4] H. Mazi, F. N. Arsene and A. M. Dissanayaka, "The influence of black market activities through dark web on the economy: a survey," In *The Midwest Instruction and Computing Symposium (MICS)*, Milwaukee School of Engineering and Northwestern Mutual, Milwaukee, Wisconsin, 2020.
- [5] S. Gandhi, T. Poongodi and K. S. Kumar. (2024). Original Research Article Enhancing data security of cardiac patients in IoMT with Twin-Shield Encryption. *Journal of Autonomous Intelligence*. [Online]. 7(2). Available: 10.32629/jai.v7i2.1322
- [6] K. T. Kadhim, A. M. Alsahlany, S. M. Wadi, H. T. Kadhun. (2020, May.). An Overview of Patient's Health Status Monitoring System Based on Internet of Things (IoT). *Wireless Personal Communications*. [Online]. 114(3), pp. 2235-2262. Available: 10.1007/s11277-020-07474-0
- [7] A. Rghioui and A. Oumnad. (2018, Oct.). Challenges and Opportunities of Internet of Things in Healthcare. *International Journal of Electrical & Computer Engineering*. [Online]. 8(5), pp. 2753-2761. Available: 10.11591/ijece.v8i5.pp2753-2761
- [8] P. Kasinathan et al. and M. A. Spirito, "An IDS framework for internet of things empowered by 6LoWPAN" *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, 2013, pp. 1337-1340.
- [9] M. Xie. S. Han, B. Tian and S. Parvin. (2011, Jul.). Anomaly detection in wireless sensor networks: A survey. *Journal of Network and computer Applications*. [Online]. 34(4), pp. 1302-1325. Available: 10.1016/j.jnca.2011.03.004
- [10] S. Rajasegarar, C. Leckie and M. Palaniswami. (2008, Aug.). Anomaly detection in wireless sensor networks. *IEEE Wireless Communications*. [Online]. 15(4), pp. 34-40. Available: 10.1109/MWC.2008.4599219
- [11] M. Behniafar, A. Nowroozi and H. R. Shahriari. (2018, Jul.). A Survey of Anomaly Detection Approaches in Internet of Things. *IseCure*. [Online]. 10(2). Available: sid.ir
- [12] CyberMDX, "2020 Vision: A Review of Major IT & Cyber Security Issues Affecting" Healthcare, 2020.
- [13] W. MADDOX, "Why Medical Data is 50 Times More Valuable Than a Credit Card," Healthcare, 2020.
- [14] A. Ghubaish, T. Salman, M. Zolanvari, D. Unal, A. Al-Ali and R. Jain. (2020, Dec.). Recent advances in the internet-of-medical-things (IoMT) systems security. *IEEE Internet of Things Journal*. [Online]. 8(11), pp. 8707-8718. Available: 10.1109/IIOT.2020.3045653
- [15] G. Zachos et al. and J. Rodriguez. (2021, Oct.). An anomaly-based intrusion detection system for internet of medical things networks. *Electronics*. [Online]. 10(21), p. 2562. Available: 10.3390/electronics10212562
- [16] T. S. Mohamed, S. Aydin, A. Alkhayyat and R. Q. Malik. (2022, Jul). Kalman and Cauchy clustering for anomaly detection based authentication of IoMTs using extreme learning machine. *IET Communications*. [Online]. Available: 10.1049/cmu2.12467
- [17] D. Alsalman. (2024, Jan.). A Comparative Study of Anomaly Detection Techniques for IoT Security using AMoT (Adaptive Machine Learning for IoT Threats). *IEEE Access*. [Online]. 12, pp. 14719-14730. Available: 10.1109/ACCESS.2024.3359033
- [18] J. Wang et al. and K. Zhong. (2022, Dec.). Anomaly detection in Internet of medical Things with Blockchain from the perspective of deep neural network. *Information Sciences*. [Online]. 617, pp. 133-149. Available: 10.1016/j.ins.2022.10.060
- [19] S. H. Wagan et al. and D. R. Shin. (2023, Jan.). A fuzzy-based duo-secure multi-modal framework for IoMT anomaly detection. *Journal of King Saud University-Computer and Information Sciences*. [Online]. 35(1), pp. 131-144. Available: 10.1016/j.jksuci.2022.11.007
- [20] J. B. Awotunde et al. and R. G. Jimoh, "A deep learning-based intrusion detection technique for a secured IoMT system," *International Conference on Informatics and Intelligent Applications*, Cham: Springer International Publishing, 2021, pp. 50-62.

- [21] P. Kumar, G. P. Gupta and R. Tripathi. (2021, Jan.). An ensemble learning and fog-cloud architecture-driven cyber-attack detection framework for IoMT networks. *Computer Communications*. [Online]. 166, pp. 110-124. Available: 10.1016/j.comcom.2020.12.003
- [22] M. Tavallaee, E. Bagheri, W. Lu and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," *IEEE symposium on computational intelligence for security and defense applications*, Ottawa, ON, Canada, 2009, pp. 1-6.
- [23] L. Dhanabal and S. P. Shantharajah. (2015, Jun.). A study on NSL-KDD dataset for intrusion detection system based on classification algorithms. *International journal of advanced research in computer and communication engineering*. [Online]. 4(6), pp. 446-452. Available: 10.17148/IJARCC.2015.4696
- [24] T. Su, H. Sun, J. Zhu, S. Wang and Y. Li. (2020, Feb.). BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset. *IEEE*. [Online]. 8, pp. 29575-29585. Available: 10.1109/ACCESS.2020.2972627
- [25] S. Kavitha and N. U. Maheswari. (2021, Mar.). Network anomaly detection for NSL-KDD dataset using deep learning. *Information Technology*. [Online]. 9(2), pp. 821-827.
- [26] W. Xu and F. Sabrina. (2021, Sep.). Improving performance of autoencoder-based network anomaly detection on nsl-kdd dataset. *IEEE*. [Online]. 9, pp. 140136-140146. Available: 10.1109/ACCESS.2021.3116612
- [27] F. Türk. (2023, Jun.). Analysis of intrusion detection systems in UNSW-NB15 and NSL-KDD datasets with machine learning algorithms. *Bitlis Eren Üniversitesi Fen Bilimleri Dergisi*. [Online]. 12(2), pp. 465-477. Available: 10.17798/bitlisfen.1240469
- [28] K. S. Bhuvaneshwari et al. and P. Prabu. (2022, Jul.). Improved Dragonfly Optimizer for Intrusion Detection Using Deep Clustering CNN-PSO Classifier. *Computers, Materials & Continua*. [Online]. 70(3). Available: 10.32604/cmc.2022.020769
- [29] S. S. Kaushik and P. R. Deshmukh. (2011). Detection of attacks in an intrusion detection system. *International Journal of Computer Science and Information Technologies (IJCSIT)*. [Online]. 2(3), pp. 982-986. Available: citeseerx.ist.psu.edu
- [30] B. Mahesh. (2020, Jan.). Machine learning algorithms-a review. *International Journal of Science and Research (IJSR)*. [Online]. 9(1), pp. 381-386. Available: 10.21275/ART20203995
- [31] F. T. Liu, K. M. Ting and Zhi-Hua Zhou, "Isolation forest," *eighth ieee international conference on data mining*, Pisa, Italy, 2008, pp. 413-422.
- [32] N. Abe, B. Zadrozny and J. Langford, "Outlier detection by active learning," *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006, pp. 504-509.
- [33] Z. He, X. Xu and S. Deng. (2003, Jun.). Discovering cluster-based local outliers. *Pattern recognition letters*. [Online]. 24(9-10), pp. 1641-1650. Available: 10.1016/S0167-8655(03)00003-5
- [34] M. Al-Qatf, Y. Lasheng, M. Al-Habib and K. Al-Sabahi. (2018, Sep.). Deep learning approach combining sparse autoencoder with SVM for network intrusion detection. *Ieee Access*. [Online]. 6, pp. 52843-52856. Available: 10.1109/ACCESS.2018.2869577
- [35] C. Ieracitano et al. and A. Hussain. (2020, Apr.). A novel statistical analysis and autoencoder driven intelligent intrusion detection approach. *Neurocomputing*. [Online]. 387, pp. 51-62. Available: 10.1016/j.neucom.2019.11.016
- [36] A. Javaid, Q. Niyaz, W. Sun and M. Alam, "A deep learning approach for network intrusion detection system," *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, 2016, pp. 21-26.

CONTENTS

Enhancing Channel Selection in 5G with Decentralized Federated Multi-Agent Deep Reinforcement Learning	1
Taghi Shahgholi- Keyhan Khamforoosh- Amir Sheikhamadi- Sadoon Azizi	
An Enhanced Sine Cosine Algorithm for Feature Selection in Network Intrusion Detection	17
Zahra Asghari Varzaneh- Soodeh Hosseini	
Evaluating Developers' Expertise in Serverless Functions by Mining Activities from Multiple Platforms	27
Aref Talebzadeh Bardsiri- Abbas Rasoolzadegan	
Description-based Post-hoc Explanation for Twitter List Recommendations	43
Havva Alizadeh Noughabi- Behshid Behkamal- Mohsen Kahani	
Machine Learning Classifiers and Data Synthesis Techniques to Tackle with Highly Imbalanced COVID-19 Data	51
Avaz Naghipour- Mohammad Reza Abbaszadeh Babil Soflaci- Mostafa Ghaderi-Zefrehei	
Anomaly Detection in IoMT Environment Based on Machine Learning: an Overview	65
Peyman Vafadoost Sabzevar- Hamidreza Rokhsati- Alireza Chamansara- Ahmad Hajipour	