NSS
Neutrosophic Sets and Systems, Vol. 85, 2025
University of New Mexico

Evaluation of Intrusion Detection Systems in Cyber Security using Fuzzy OffLogic and MCDM Approach

Zhengrui Yang*
School of Cyber Science and Engineering, Zhengzhou University, Zhengzhou, Henan, 450002, China
*Corresponding author: E-mail: yzrhw0218@163.com

Abstract
Modern cybersecurity infrastructures rely heavily on Intrusion Detection Systems (IDS) to detect and prevent malicious activities and unauthorized access. Given the growing complexity of network topologies and the rising frequency of cyber threats, evaluating IDS solutions requires a systematic and unbiased approach. In this study, thirteen widely used IDS models are assessed using a multi-criteria evaluation framework across four key dimensions: detection accuracy, resource efficiency, scalability, and false positive rate. The goal is to support informed, data-driven decision-making for stakeholders such as policymakers, IT administrators, and security analysts when selecting an appropriate IDS. The VIKOR method is employed to rank the IDS alternatives based on the assigned weights, while Fuzzy OffLogic is applied to integrate expert assessments expressed as intervals. The results reveal that modern AI-based IDS models demonstrate strong performance in scalability and resource utilization, and they outperform traditional systems in adaptability and detection accuracy.

Keywords: Fuzzy OffLogic; Security; Cyber-Security; Attacks; Intrusion Detection System; MCDM Approach.

1. Introduction

One popular cyber security technique is intrusion detection, which looks for hostile activity in host and/or network environments. Malicious activity detection allows for prompt action to, for instance, halt an ongoing attack. Numerous intrusion detection systems (IDS) have been devised and developed by the industrial and research communities due to the significance of intrusion detection[1], [2]. The creation of IDS evaluation approaches, techniques, and tools has emerged as a crucial area of research due to the growing diversity and complexity of IDSs. IDS assessment has several advantages. To implement an IDS that performs best in a particular environment and lowers the risks of a security breach, one could, for example, compare many IDS based on how well they identify attacks. Additionally, by altering its configuration parameters and examining their impact through assessment tests, one can fine-tune an IDS that has already been implemented[3], [4].

A wide range of tasks pertaining to IDS evaluation have been carried out by the industrial and scientific groups. A structured classification is required to enhance the general comprehension of the subject and to give an overview of the present status of the field, given the substantial amount of existing theoretical and practical work pertaining to IDS evaluation. Finding and comparing the benefits and drawbacks of various IDS evaluation techniques and procedures would be made easier with the help of such an overview. Additionally, it would assist in determining the best practices and requirements for assessing present and future IDS[5], [6].

The workload component of the design space for IDS evaluation. We require both harmful and benign workloads to assess an IDS. These can be used independently to gauge an IDS's capabilities (for example, as workloads that are pure malicious or pure benign). Workloads that only contain attacks are known as pure malicious workloads, while workloads that do not contain any attacks are known as pure benign workloads. As an alternative, one can expose an IDS under test to realistic threats by using mixed workloads, which are workloads that combine pure benign and pure malicious workloads[7], [8].

Workloads for IDS assessment often take the form of an executable for live IDS testing or a trace form, which is created by recording a live workload execution for subsequent playback. Tools made to analyze trace files are used to execute the trace replay; a popular combination is to utilize the utility TCP dump to record network traces for TCP replay to replay later.

The main benefit of using executable workloads is that they closely mimic actual workloads that are monitored by an intrusion detection system while it is in operation. However, setting up a malicious workload in executable form necessitates a particular victim environment, which can be costly and time-consuming. On the other hand, replaying workload does not always necessitate such an environment.

Furthermore, to guarantee the statistical significance of the observed system behavior, several evaluation runs are usually needed. Replicating assessment studies with executable malicious workloads is typically difficult, though, because the execution of assaults may cause the victim environment to crash or become unstable. Furthermore, the procedure to return the environment to its pre-attack form could take a long time[9].

2. Literature Review

If a system possesses the qualities of confidentiality, integrity, and availability of its data and services, it is deemed secure. Attacks are intentional attempts to compromise the security features listed above.

Kruegel et al. [10] categorize security measures that uphold the qualities using an attack-centric approach. The first class includes mechanisms that keep attackers from getting to the target and interacting with it, like access control; the second class includes mechanisms that change the data stored there so that an attacker cannot use it, like data encryption; and the third class includes mechanisms that identify ongoing attacks based on the presumption that an attacker can get to

the target and interact with it. Intrusion detection is a security mechanism that falls under this class.

In terms of computer security regulations, proper usage policies, or typical safety procedures, detection of breaches is defined as "the process for tracking occurrences happening on a network or computer environment and analyzing them for signs for potential incidents, which are infractions or immediate dangers of violation." According to the definition given above, an IDS is software that automates the detection of an attack process [11].

One area of active research is the assessment of computer intrusion detection systems, or intrusion detection systems for short. Milenkoski et al. [12] examined and organized standard procedures in the field of evaluating such systems. To do this, they established a design space that is divided into three sections: measuring methods, metrics, and workload. By reviewing the evaluation techniques and approaches associated with each area of the design space, they then gave a summary of the standard procedures in the assessment of intrusion detection systems. Lastly, they have gone over unresolved problems and difficulties with an emphasis on assessment techniques for innovative intrusion detection systems.

2.1 multi-criteria decision-making

Multi-criteria decision-making (MCDM) is a sophisticated process that uses straightforward algorithms to provide suitable conclusions with the right ranking. MCDM methods work effectively for managing massive volumes of data as well. Numerous algorithms, including VIKOR, and the average method, modify the fundamentals of MCDM. The selection of the best alternative based on VIKOR, and so on are examples of applications that apply MCDM approaches in the context of IDS[13].

2.2 Intrusion Detection Systems (IDS)

Information systems protection is greatly aided by intrusion detection systems (IDSs). Numerous human or humanoid assailants frequently target cyber-physical systems. The obnoxious, dangerous network traffic must be protected from the vital parts and resources of the network architecture. IDS plays a very sensitive function in this situation, and its ability to handle various attack types from many sources is desperately needed[13], [14].

IDS provides cyber protection against a range of threats and weaknesses found in Internet applications. Any networked environment is full of threats. IDS protects networks by keeping an eye on and blocking the entry of anonymous or questionable network traffic. These days, a lot of IDS models are created using machine learning techniques. Generally speaking, anomaly detection and signature-based prediction are the two approaches used to identify attack trends[15], [16].

Since the majority of IDSs are used in heterogeneous, distributed environments, they must constantly process enormous amounts of data with redundant, superfluous features. IDSs

occasionally handle data with extraneous aspects as well. The model's training time will undoubtedly be slowed down in this scenario, resulting in decreased accuracy. The performance of the classifier model will deteriorate as a result. Another important consideration is that a model with fewer cum meaningful characteristics is less likely to make mistakes and thus require less training time[17], [18].

It is challenging to detect anomalies using network intrusion detection systems (NIDS) since vulnerable packets attempt to avoid the network and new attacks are created by different automated attack-triggering mechanisms.

2.3 Real-World Deployment Challenges of IDS Models

Although this study focuses on evaluating IDS performance using structured decision-making techniques, actual deployment of these systems in real-world environments comes with challenges that are often more complex than test conditions can reveal.

Many IDS models require specific tuning and configuration based on the network in which they are deployed. In practice, setting the right parameters for detection thresholds, processing capacities, and alert handling can take significant time and effort. In cloud-based or hybrid environments, IDS solutions also face the challenge of integration with virtual infrastructure, where data packets may not follow consistent paths, and encryption can obscure inspection.

Another challenge is related to maintenance. Some models, especially AI-based ones, need continuous retraining as attack patterns evolve. Without regular updates, these systems may become outdated quickly, losing their detection capabilities. This requires access to updated datasets and adequate computational resources, which may not always be available.

Moreover, system administrators often struggle with balancing detection sensitivity and false alarm rates. If the IDS is too sensitive, it may flood the team with alerts. If it's too lenient, real threats might be missed. These trade-offs must be handled in living environments under pressure, which adds another layer of difficulty compared to controlled simulations.

2.4 Comparative Summary of Classic vs. AI-based IDS

The evaluation in this study includes both traditional IDS like Snort and Zeek and newer AI-driven systems such as QRadar and Darktrace. While the numerical analysis shows differences in detection accuracy, resource efficiency, and scalability, a broader view helps explain why AI-based systems often rank higher.

Classic IDS models generally rely on pre-defined rules and signature databases. Their strength lies in predictability and low resource usage. They are open-source and flexible, often used by organizations with tight budgets or high customization needs. However, these models may struggle when faced with unknown or fast-changing threats.

In contrast, AI-based IDS solutions adapt more quickly to new patterns. They analyze behavior rather than signatures, which helps detect previously unseen attacks. These systems also scale more easily in distributed environments and can process high volumes of data in real time.

However, AI-based models are not without drawbacks. They typically require more system resources and a complex initial setup. They may also act like "black boxes," making it harder for teams to understand how decisions are made. Despite this, the overall analysis from this study suggests that their performance advantages in scalability and accuracy often outweigh these limitations.

2.5. Evaluation of Trade-Offs in Multi-Criteria Decision-Making

The study uses four criteria to evaluate IDS: detection accuracy, resource efficiency, scalability, and false positive rate. Each of these metrics reflects a different aspect of system performance, but choosing the right balance between them is not always straightforward.

For instance, systems with high detection accuracy may also generate more false positives. This can lead to alert fatigue, where security teams begin ignoring warnings. On the other hand, reducing the false positive rate too much might result in more attacks slipping through undetected. Similarly, some models perform well in terms of accuracy and low false positives but consume significant computational resources, making them unsuitable for low-power environments like IoT or edge computing.

The VIKOR method, supported by Fuzzy OffLogic, helps manage these trade-offs by assigning weights to each criterion. However, it's important to recognize that these weights reflect expert opinions, which may differ between organizations based on their specific needs. A system chosen by one company as "best" may not be the top choice in another with different operational priorities.

This trade-off analysis highlights the need for flexible evaluation models that can be adapted to various contexts, rather than promoting one-size-fits-all solutions.

3. Fuzzy OffLogic

In 2007, Smarandache extended the concept of the uncertain set to include three generalized forms: the uncertain OverSet, the uncertain UnderSet, and the uncertain OffSet [19], [20]. The OverSet accounts for cases in which some components have values greater than 1. For instance, an employee who works overtime may justifiably receive a degree of membership greater than 1, compared to a full-time employee whose degree is equal to 1. The UnderSet, on the other hand, applies when a component has a value less than 0. As an example, an employee who causes more harm than benefit to their organization may be assigned a membership degree less than zero, in contrast to a productive employee with a positive membership value.

The OffSet is a more general case in which certain components lie outside the standard interval [0,1][0, 1][0,1], meaning some values are greater than 1 while others are less than 0. This allows

the model to represent more complex or extreme situations that cannot be captured using traditional fuzzy logic.

Smarandache also extended other uncertain frameworks—such as logic, measure, probability, and statistics—into their corresponding Over-, Under-, and Off- variants. By "uncertain," he refers to all generalizations of fuzzy logic, including intuitionistic fuzzy sets, neutrosophic logic, spherical fuzzy models, and plithogenic sets [21], [22].

To illustrate this concept in a practical context, consider a student who earns an A+ grade. The Fuzzy OffTruth value associated with their performance may fall in a range such as [1.0,1.1][1.0, 1.1][1.0,1.1], even though the overall evaluation scale spans from [−1.2,1.1][-1.2, 1.1][−1.2,1.1] [23]. This reflects performance that slightly exceeds the conventional maximum and demonstrates the flexibility of OffLogic in capturing overachievement.

3.1 Illustrative Examples of Fuzzy OffLogic Calculations

Consider the following examples that demonstrate how to compute the OffLogic value when evaluating outcomes that may extend beyond the conventional [0, 1] interval.

Example 1

Suppose a student receives a result represented by the interval [1.0,1.1], within a full range that spans from −1.2- to 1.11 The Fuzzy OffLogic value can be calculated using the following expression:

$$F_{Offlogic} = \frac{1.1 - 1.0}{1.1 - (-1.2)} = \frac{0.1}{2.3} = 0.043 \quad (1)$$

This small value indicates a slight overperformance relative to the upper bound of the standard fuzzy scale.

Example 2

In another scenario, if the outcome interval lies between [9.0,1.1] to the test out of [-1.2,1.1].

$$F_{Offlogic} = \frac{1.1 - 9.0}{1.1 - (-1.2)} = \frac{0.2}{2.3} = 0.086 \quad (2)$$

This significantly negative value reflects a performance that extends far beyond the typical bounds and is interpreted as an extreme deviation from the expected range.

Example 3

Similarly, if another individual scores between [8.0,1.1] to the test out of [-1.2,1.1].

$$F_{Offlogic} = \frac{1.1 - 8.0}{1.1 - (-1.2)} = \frac{0.3}{2.3} = 0.130 \quad (5)$$

This also indicates a substantial departure from the neutral range, with the score falling well outside conventional expectations.

Example 4

if another individual scores between [7.0,1.1] to the test out of [-1.2,1.1].

$$F_{Offlogic} = \frac{1.1 - 7.0}{1.1 - (-1.2)} = \frac{0.4}{2.3} = 0.1739 \quad (6)$$

Example 5

if another individual scores between [6.0,1.1] to the test out of [-1.2,1.1].

$$F_{Offlogic} = \frac{1.1 - 6.0}{1.1 - (-1.2)} = \frac{0.5}{2.3} = 0.217 \quad (7)$$

Example 6

if another individual scores between [5.0,1.1] to the test out of [-1.2,1.1].

$$F_{Offlogic} = \frac{1.1 - 5.0}{1.1 - (-1.2)} = \frac{0.6}{2.3} = 0.260 \quad (8)$$

Example 7

if another individual scores between [4.0,1.1] to the test out of [-1.2,1.1].

$$F_{Offlogic} = \frac{1.1 - 4.0}{1.1 - (-1.2)} = \frac{0.7}{2.3} = 0.304 \quad (9)$$

Example 8

if another individual scores between [3.0,1.1] to the test out of [-1.2,1.1].

$$F_{Offlogic} = \frac{1.1 - 3.0}{1.1 - (-1.2)} = \frac{0.8}{2.3} = 0.347 \quad (10)$$

Example 9

if another individual scores between [2.0,1.1] to the test out of [-1.2,1.1].

$$F_{Offlogic} = \frac{1.1 - 2.0}{1.1 - (-1.2)} = \frac{0.9}{2.3} = 0.391 \quad (11)$$

Example 10

if another individual scores between [0.1,1.1] to the test out of [-1.2,1.1].

$$F_{Offlogic} = \frac{1.1 - 0.1}{1.1 - (-1.2)} = \frac{1}{2.3} = 0.434 \quad (12)$$

These examples demonstrate how Fuzzy OffLogic flexibly handles values outside the standard fuzzy range. It allows for more expressive evaluations of performance, especially when considering overachievement or underperformance beyond the limits of traditional logic.

3.2 Enhancing Fuzzy OffLogic Interpretability

Fuzzy OffLogic was used in this study to convert expert opinions, expressed as intervals, into crisp values for ranking. While this approach improves flexibility and allows for uncertainty, understanding how these values are derived can be challenging for users unfamiliar with the logic behind it.

The core idea of OffLogic is that decision scores can extend beyond the traditional [0,1] range allowing some values to be greater than 1 or less than 0. This helps model situations where an option is considered better than "perfect" or worse than "completely unacceptable." For example, if one IDS detects attacks before they even reach the system, its value for accuracy might logically exceed 1 in a relative sense.

However, such concepts can be difficult to interpret for practitioners who are used to conventional logic systems. One way to make this clearer is by providing visual aids or simple examples during decision-making. A dashboard that maps OffLogic scores to understandable categories like "very strong," "satisfactory," or "needs review" could help bridge the gap between complex calculations and practical decisions.

Ultimately, Fuzzy OffLogic adds depth to the evaluation, but it also requires careful explanation if it's going to be adopted outside of academic or highly technical settings.

3.3 Applying VIKOR under Fuzzy OffLogic for Multi-Criteria IDS Evaluation

In this section, the VIKOR method is applied in combination with Fuzzy OffLogic to evaluate and rank intrusion detection systems (IDS) based on multiple performance criteria. The integration of these two techniques allows decision-makers to work with uncertain and imprecise input from experts while still producing reliable and consistent rankings.

The process begins with experts constructing a decision matrix by evaluating each IDS alternative against the defined criteria. These evaluations are not expressed as fixed numbers but rather as intervals to reflect uncertainty or varying confidence levels. Fuzzy OffLogic is used to convert these intervals into crisp values. This method allows components to exceed traditional boundaries, meaning values may fall outside the standard range of [0, 1], depending on the strength or weakness of a given performance assessment.

After obtaining crisp values, the average method is used to calculate the weights of the criteria. These weights reflect the relative importance of each criterion in the evaluation process and are critical for establishing a balanced decision model.

Once the decision matrix and weights are finalized, the VIKOR method is applied through a series of structured steps. The first step involves normalizing the decision matrix for both benefit and cost-type criteria using equations (13) and (14), as shown in Figure 1. This ensures that all values are comparable across different scales. Then, the weighted normalized decision matrix is calculated using equation (15), as illustrated in Figure 2.

$$Y_{ij} = \frac{\max_i x_{ij} - x_{ij}}{\max_i x_{ij} - \min_i x_{ij}} \tag{13}$$

$$Y_{ij} = \frac{\min_i x_{ij} - x_{ij}}{\min_i x_{ij} - \max_i x_{ij}} \tag{14}$$

$$U_{ij} = w_j Y_{ij} \tag{15}$$

Next, two key indicators S (representing the overall utility) and R (representing the individual regret)—are computed using equations (16) and (17). These indicators capture both the collective performance of each alternative and its weakest performance under any specific criterion. The values are shown in Figure 3.

$$S_i = \sum_{j=1}^{n} U_{ij} \tag{16}$$

$$R_i = \max_j U_{ij} \tag{17}$$

Using equation (18), the final index Pi is computed for each alternative. This value combines both S and R, weighted by a parameter H, which was set to 0.5 in this study to reflect equal importance between group utility and individual regret. The final scores are presented in Figure 4.

$$P_i = H \times \left(\frac{S_i - \min_i S_i}{\max_i S_i - \min_i S_i} \right) + (1 - H) * \left(\frac{R_i - \min_i R_i}{\max_i R_i - \min_i R_i} \right) \tag{18}$$

Finally, the alternatives are ranked based on their Pi values in ascending order, where lower values indicate better overall performance. The resulting rankings are shown in Figure 5, providing a clear and structured assessment of which IDS solutions offer the best balance across all evaluation dimensions.

By integrating Fuzzy OffLogic with the VIKOR method, this approach captures expert uncertainty while supporting precise decision-making. It offers a robust framework for evaluating complex systems like IDS in a flexible yet mathematically rigorous manner.

4. Experimental Results

This section presents a detailed evaluation of thirteen intrusion detection system (IDS) models using a structured decision-making framework. The aim is to assess each model based on its performance across four core criteria that reflect the essential capabilities of IDS in real-world cybersecurity environments. To ensure fairness and objectivity, the evaluation process was conducted in collaboration with three domain experts who contributed their judgments based on their experience.

The experts first defined the decision matrix by assigning values to each criterion for all 13 IDS models. Instead of using fixed values, they worked with performance intervals that reflect the uncertainty often present in subjective evaluations. These intervals were then processed using Fuzzy OffLogic, which translates imprecise values into crisp numbers. This method allows more flexible handling of expert input while maintaining mathematical consistency.

The evaluation was based on the following four key criteria:

1. C1: Detection Accuracy: This measures how well the IDS can identify and correctly classify malicious activities. A higher accuracy means the system can detect threats more reliably, reducing the chances of undetected intrusions. As expected, this criterion received the highest priority, as accurate detection is the primary goal of any IDS.
2. C2: Resource Efficiency: This criterion evaluates the system's use of CPU, memory, and bandwidth. Efficient systems can operate without placing a heavy load on hardware resources, making them suitable for deployment in environments where computational capacity is limited.
3. C3: Scalability: Scalability reflects the system's ability to manage increased traffic and expand across distributed network environments. This is particularly important for larger organizations or systems operating in dynamic and growing infrastructures.
4. C4: False Positive Rate: A false positive occurs when legitimate activity is incorrectly flagged as a threat. While lower in priority compared to the other three, minimizing false positives is essential to avoid overwhelming system administrators with unnecessary alerts and to maintain trust in the system's outputs.

The weights assigned to each criterion were calculated using the average method based on the experts' input. The results of this weighting reflect the practical priorities in IDS performance:

1) Detection Accuracy (C1) received a weight of 0.2611, confirming its status as the most crucial aspect.
2) Resource Efficiency (C2) followed closely with 0.2500, indicating the high value placed on system performance and stability.
3) Scalability (C3) was assigned 0.2476, showing that the ability to adapt to larger or evolving networks is nearly as important.
4) False Positive Rate (C4) held a weight of 0.2412, still significant but slightly lower, recognizing its role in operational efficiency without overshadowing detection capabilities.

By using the VIKOR method alongside Fuzzy OffLogic, the evaluation was able to produce refined, balanced, and interpretable scores for each IDS model. These scores reflect how each system performs across a combination of technical and operational requirements, providing a clear foundation for comparing alternatives in a structured, data-driven way.

Table 1. The values of the decision matrix.

	C_1	C_2	C_3	C_4
A_1	([0.20, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])
A_2	([0.10, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])
A_3	([0.10, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])
A_4	([0.80, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])
A_5	([0.90, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])
A_6	([0.10, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])
A_7	([0.90, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])
A_8	([0.30, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])
A_9	([0.40, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])
A_{10}	([0.10, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])
A_{11}	([0.80, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])
A_{12}	([0.80, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])
A_{13}	([0.90, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])
	C_1	C_2	C_3	C_4
A_1	([0.90, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])
A_2	([0.10, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])
A_3	([0.20, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])
A_4	([0.30, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])
A_5	([0.40, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])
A_6	([0.70, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])
A_7	([0.80, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])
A_8	([0.30, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])
A_9	([0.20, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])
A_{10}	([0.10, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])
A_{11}	([0.70, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])
A_{12}	([0.10, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])
A_{13}	([0.90, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])
	C_1	C_2	C_3	C_4
A_1	([0.40, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])
A_2	([0.70, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])
A_3	([0.80, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])
A_4	([0.90, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])
A_5	([0.10, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])
A_6	([0.40, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])
A_7	([0.70, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])
A_8	([0.30, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])
A_9	([0.20, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])
A_{10}	([0.10, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])
A_{11}	([0.70, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])
A_{12}	([0.80, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])
A_{13}	([0.10, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])
	C_1	C_2	C_3	C_4
A_1	([0.90, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])
A_2	([0.90, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])
A_3	([0.80, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])
A_4	([0.40, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])
A_5	([0.30, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])
A_6	([0.30, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])
A_7	([0.20, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])
A_8	([0.30, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])
A_9	([0.20, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])
A_{10}	([0.10, 1.1] ,[-1.2, 1.1])	([0.90, 1.1] ,[-1.2, 1.1])	([0.80, 1.1] ,[-1.2, 1.1])	([0.70, 1.1] ,[-1.2, 1.1])
A_{11}	([0.70, 1.1] ,[-1.2, 1.1])	([0.40, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])
A_{12}	([0.90, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])
A_{13}	([0.10, 1.1] ,[-1.2, 1.1])	([0.20, 1.1] ,[-1.2, 1.1])	([0.30, 1.1] ,[-1.2, 1.1])	([0.10, 1.1] ,[-1.2, 1.1])

To proceed with the evaluation, the normalized decision matrix was computed for both positive and negative criteria using equations (13) and (14). This normalization step ensured that all criteria, regardless of their nature or units, could be compared on the same scale. The process allowed for a fair transformation of values across different metrics, whether they were to be maximized or minimized. The resulting normalized matrix is illustrated in Figure 1.

Once the criteria were normalized, the next step was to apply the previously determined weights to each criterion. Using equation (15), the weighted decision matrix was calculated, capturing the relative importance of each criterion as defined by the expert inputs. This step helped align the raw evaluation data with the practical priorities of the decision context. The final weighted values are displayed in Figure 2.

Following the construction of the weighted matrix, the VIKOR method was employed to calculate two performance indices, S and R, for each alternative. These values were computed using equations (16) and (17), as shown in Figure 3. The S index reflects the overall group utility, while the R index focuses on the regret measure, representing the worst-case performance relative to any specific criterion.

After obtaining the S and R values, the final score for each alternative was computed using equation (18). This equation integrates both S and R into a single measure, denoted as Pi, to determine the overall compromise solution. For this analysis, the value of H was set to 0.5 to give equal importance to both group utility and individual regret. The final calculated scores are presented in Figure 4. Based on the values of Pi, the alternatives were ranked from best to worst, where a lower Pi, value indicated a better overall performance. This final ranking, which reflects a balanced view of the alternatives across all criteria, is summarized in Figure 5.

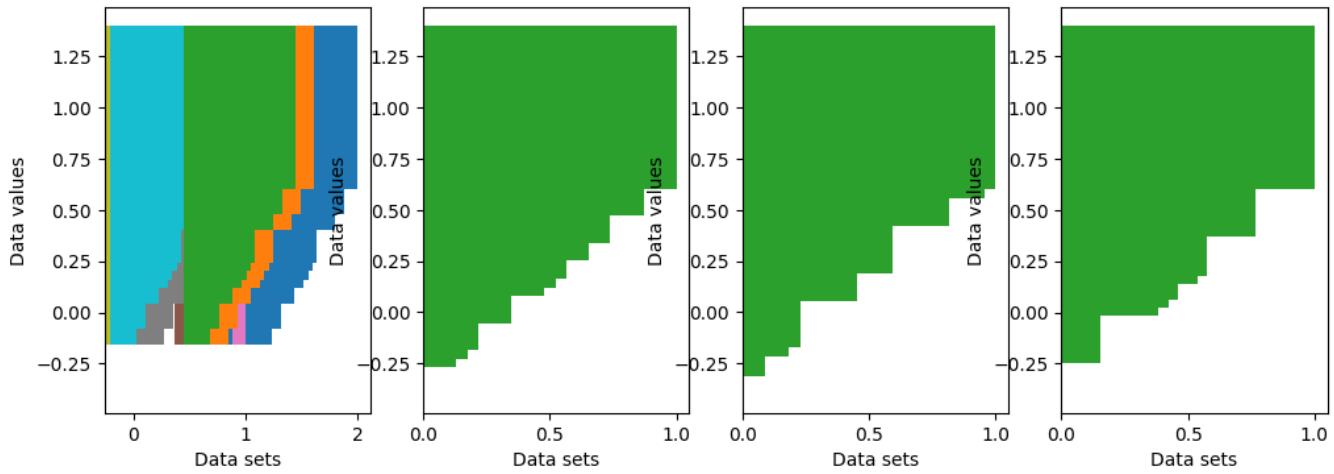

Fig 1. The normalized decision matrix.

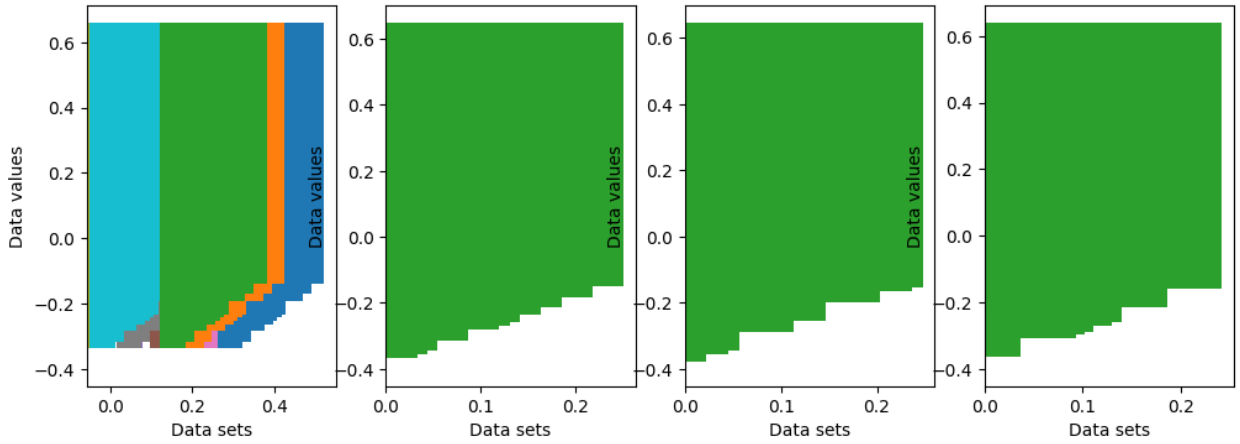

Fig 2. The weighted decision matrix.

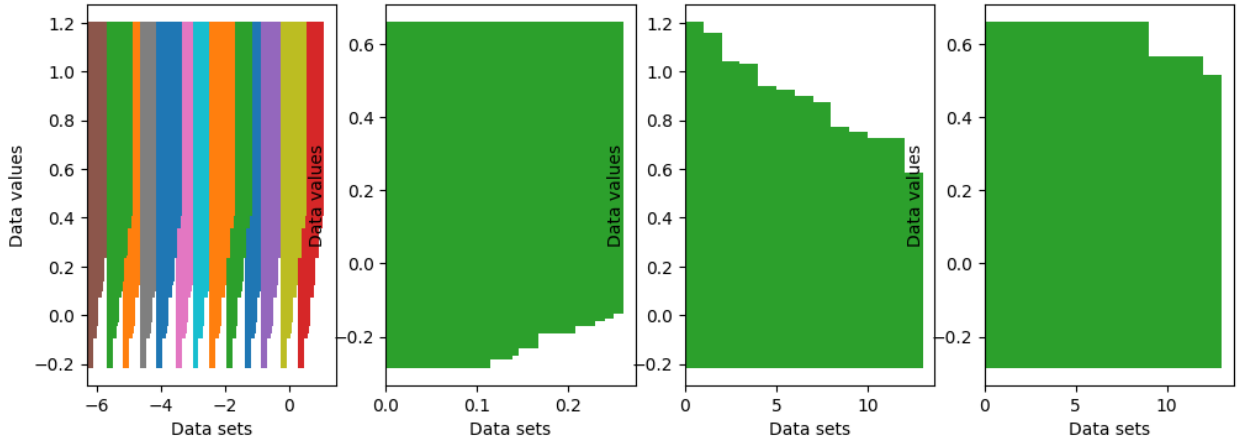

Fig 3. The values of S and R.

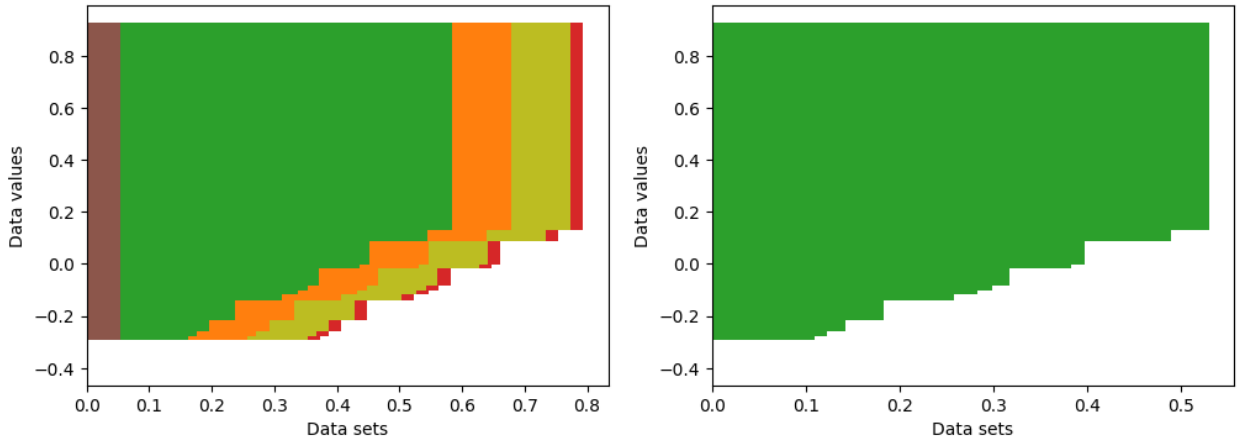

Fig 4. The values of P_i.

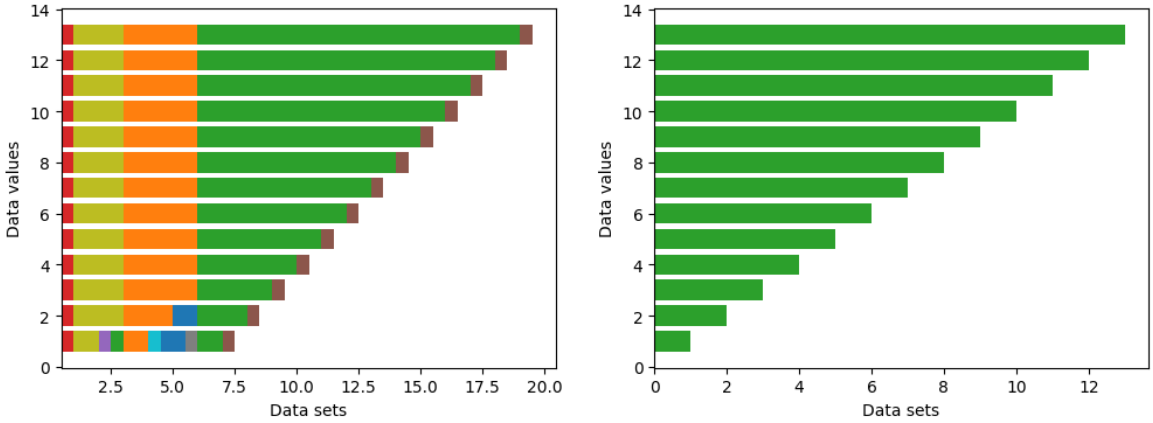

Fig 5. The ranks of alternatives.

5. Sensitivity Testing and Ranking Stability

This section aims to vary the parameter H in the VIKOR method between 0 and 1 and to rank the alternatives based on the resulting values to assess the stability of the rankings. Figure 6 illustrates the different Pi values obtained across the range of H, while Figure 7 presents the corresponding ranks of the alternatives under these varying conditions. The results indicate that the rankings remain stable across different values of H, demonstrating the robustness of the proposed evaluation approach.

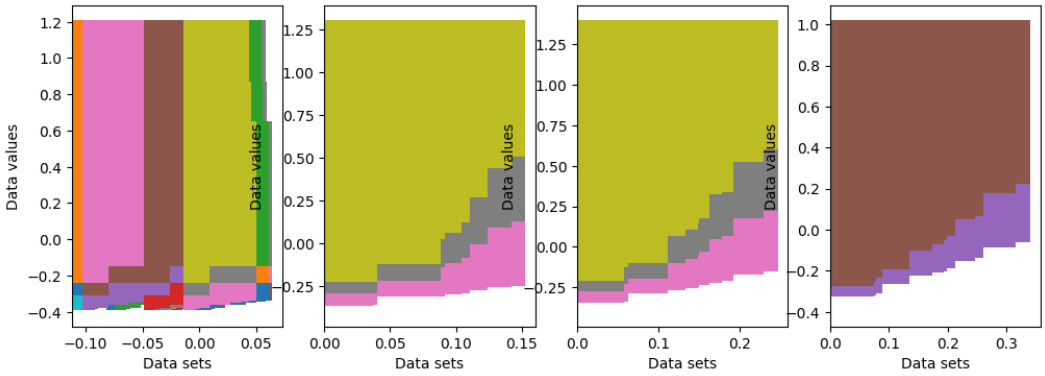

Fig 6. The new values of P_i.

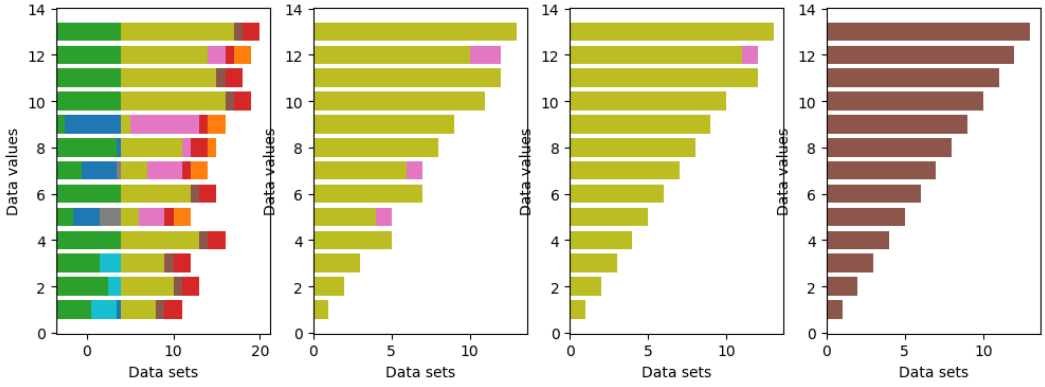

Fig 7. The new values rank.

As part of the final evaluation using the VIKOR method, the ranking of the thirteen IDS alternatives was analyzed across different values of the parameter H, which represents the balance between group utility and individual regret. This analysis provides deeper insight into how each system responds to changes in the decision-makers' preference for compromise versus strict performance consistency.

IDS1 showed a notable improvement in ranking as H increased. Although it began in ninth place when H was set to zero, it quickly rose and stabilized at the fourth position from H = 0.2 onward. This shift reflects the model's ability to deliver more balanced and reliable results when preferences are considered, indicating a solid performance across multiple criteria without major weaknesses.

In contrast, IDS2 remained consistently near the bottom, occupying either the eleventh or twelfth position across all H values. This suggests that its performance did not improve even when criteria weights were adjusted, indicating a general lack of adaptability or deficiency in key areas such as detection accuracy or resource efficiency.

IDS3 started strong with a second-place ranking but gradually settled in fifth place, where it remained steady. This pattern implies that while IDS3 performs very well in raw, unweighted conditions, it does not gain further advantage when additional preference intensity is introduced, perhaps due to limited flexibility in handling varied priorities.

IDS4 demonstrated the most consistent and high-ranking performance in the study. It started at fourth position and quickly rose to the top, maintaining first place from H = 0.1 to H = 1.0. Such stability at the top of the ranking indicates that IDS4 excels in all evaluated criteria and adapts effectively to shifts in decision-making emphasis, making it a strong candidate for deployment.

IDS5 maintained a position between third and fifth across all values of H, reflecting consistent and reliable performance. Although it did not reach the top rank at any point, its steady results indicate that it handles all criteria reasonably well, with only minor limitations preventing it from outperforming the leading systems.

IDS6 remained in last place across all evaluations, consistently ranking thirteenth. This points to a lack of strength in every evaluated area and suggests that the model struggles to meet the minimum expectations for a reliable IDS under varying weight distributions.

IDS7 showed no significant movement in its ranking, holding eighth place through the entire range of H values. This reflects a system with average performance that is steady but lacks competitiveness to challenge higher-performing alternatives.

IDS8 remained in the lower tier, shifting slightly between the tenth and twelfth positions. Although it showed a minor early improvement, it was not enough to escape the bottom ranks, indicating that the model does not align well with the criteria given the weightings used in this analysis.

IDS9 began in sixth place but quickly rose to second and stayed there from H = 0.1 onward. This jump suggests that IDS9 aligns strongly with the weighted preferences, demonstrating strength across the most important evaluation dimensions, and positioning it as one of the most capable systems in the group.

IDS10 began with a promising third-place rank but gradually dropped to seventh, where it stabilized. This decline implies that while the system performs well under balanced criteria, it loses effectiveness as more weight is placed on specific performance aspects, indicating limitations in scalability or adaptability.

IDS11 had an unusual pattern. It was ranked first when all criteria were treated equally at H = 0 but fell to ninth place as H increased. This change indicates that the system initially appeared strong across general criteria, but its performance did not hold up under more focused weightings, suggesting a lack of depth in specific areas.

IDS12 shifted between sixth and eighth place, showing modest capability but without any moment of breakthrough. Its performance was stable but lacked any standout feature that would push it into the top ranks.

Finally, IDS13 began in tenth place and dropped further to twelfth, suggesting weak performance and poor adaptability across all weight conditions. Like IDS2 and IDS6, its position at the lower end of the ranking highlights consistent underperformance across the board.

6. Conclusions and Future Studies

This study provided a comprehensive framework for evaluating intrusion detection systems (IDS) based on four critical criteria. The comparison of thirteen IDS solutions demonstrated that while traditional models such as Snort and Zeek remain valuable due to their open-source nature and strong community support, AI-driven systems like Darktrace and IBM QRadar tend to perform better in terms of detection accuracy and scalability. The findings emphasize the importance of a balanced selection process that aligns with an organization's threat landscape, operational goals, and technical infrastructure. To manage multiple evaluation factors, a multi-criteria decision-making (MCDM) approach was adopted. The VIKOR method was specifically applied to rank the IDS alternatives and identify the most suitable solution. A sensitivity analysis was conducted by varying key parameters within the VIKOR method to observe their impact on system rankings. The results confirmed that the proposed evaluation approach is robust, with rankings remaining stable under different parameter settings.

For future work, it is recommended to incorporate real-time adaptive learning capabilities into the evaluation criteria. Additionally, deeper integration with security information and event management (SIEM) platforms could enhance the practical relevance of IDS assessments and support more dynamic and intelligent threat response systems.

References

[1] T. Bouyahia, N. Cuppens-Boulahia, F. Cuppens, and F. Autrel, "Multi-criteria recommender approach for supporting intrusion response system," in *Foundations and Practice of Security: 9th International Symposium, FPS 2016, Québec City, QC, Canada, October 24-25, 2016, Revised Selected Papers 9*, Springer, 2017, pp. 51–67.

[2] Y. B. Abushark *et al.*, "Cyber security analysis and evaluation for intrusion detection systems," *Comput. Mater. Contin*, vol. 72, no. 1, pp. 1765–1783, 2022.

[3] J. Wang, X. Xiong, G. Chen, R. Ouyang, Y. Gao, and O. Alfarraj, "Multi-Criteria Feature Selection Based Intrusion Detection for Internet of Things Big Data," *Sensors*, vol. 23, no. 17, p. 7434, 2023.

[4] G. Kou, Y. Peng, Z. Chen, and Y. Shi, "Multiple criteria mathematical programming for multi-class classification and application in network intrusion detection," *Inf. Sci. (Ny).*, vol. 179, no. 4, pp. 371–381, 2009.

[5] S. Guan, J. Wang, C. Jiang, J. Tong, and Y. Ren, "Intrusion detection for wireless sensor networks: a multi-criteria game approach," in *2018 IEEE wireless communications and networking conference (WCNC)*, IEEE, 2018, pp. 1–6.

[6] S. M. H. Bamakan, B. Amiri, M. Mirzabagheri, and Y. Shi, "A new intrusion detection approach using PSO based multiple criteria linear programming," *Procedia Comput. Sci.*, vol. 55, pp. 231–237, 2015.

[7] G. Bernieri, S. Damiani, F. Del Moro, L. Faramondi, F. Pascucci, and F. Tambone, "A Multiple-Criteria Decision Making method as support for critical infrastructure protection and Intrusion Detection System," in *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society*, IEEE, 2016, pp. 4871–4876.

[8] V. Upendran and R. Gopinath, "Feature selection based on multi-criteria decision making for intrusion detection system," *Int. J. Electr. Eng. Technol.*, vol. 11, no. 5, pp. 217–226, 2020.

[9] M. Zbakh, K. Elmahdi, R. Cherkaoui, and S. Enniari, "A multi-criteria analysis of intrusion detection architectures in cloud environments," in *2015 International Conference on Cloud Technologies and Applications (CloudTech)*, IEEE, 2015, pp. 1–9.

[10] C. Kruegel, F. Valeur, and G. Vigna, *Intrusion detection and correlation: challenges and solutions*, vol. 14. Springer Science & Business Media, 2004.

[11] K. Scarfone and P. Mell, "Guide to intrusion detection and prevention systems (idps)," *NIST Spec. Publ.*, vol. 800, no. 2007, p. 94, 2007.

[12] A. Milenkoski, M. Vieira, S. Kounev, A. Avritzer, and B. D. Payne, "Evaluating computer intrusion detection systems: A survey of common practices," *ACM Comput. Surv.*, vol. 48, no. 1, pp. 1–41, 2015.

[13] K. D. Devprasad, S. Ramanujam, and S. B. Rajendran, "Context adaptive ensemble classification mechanism with multi-criteria decision making for network intrusion detection," *Concurr. Comput. Pract. Exp.*, vol. 34, no. 21, p. e7110, 2022.

[14] L. Diana, P. Dini, and D. Paolini, "Overview on Intrusion Detection Systems for Computers Networking Security," *Computers*, vol. 14, no. 3, p. 87, 2025.

[15] I. Makris *et al.*, "A comprehensive survey of Federated Intrusion Detection Systems: techniques, challenges and solutions," *Comput. Sci. Rev.*, vol. 56, p. 100717, 2025.

[16] A. Naghib, F. S. Gharehchopogh, and A. Zamanifar, "A comprehensive and systematic literature review on intrusion detection systems in the internet of medical things: current status, challenges, and opportunities," *Artif. Intell. Rev.*, vol. 58, no. 4, pp. 1–88, 2025.

[17] Q. A. Al-Haija and A. Droos, "A comprehensive survey on deep learning-based intrusion detection systems in Internet of Things (IoT)," *Expert Syst.*, vol. 42, no. 2, p. e13726, 2025.

[18] U. Zukaib and X. Cui, "Mitigating backdoor attacks in Federated Learning based intrusion detection systems through Neuron Synaptic Weight Adjustment," *Knowledge-Based Syst.*, vol. 314, p. 113167, 2025.

[19] F. Smarandache, *Degrees of membership> 1 and< 0 of the elements with respect to a neutrosophic offset*. Infinite Study, 2016.

[20] F. Smarandache, "Interval-Valued Neutrosophic Oversets, Neutrosophic Understes, and Neutrosophic Offsets," *Int. J. Sci. Eng. Investig.*, vol. 5, no. 54, pp. 1–4, 2016.

[21] F. Smarandache, "Neutrosophic Overset, Neutrosophic Underset, and Neutrosophic Offset. Florentin Smarandache Similarly for Neutrosophic Over-/Under-/Off-Logic, Probability, and Statistics," 2017.

[22] F. Smarandache, "Operators on single-valued neutrosophic oversets, neutrosophic undersets, and neutrosophic offsets," *Collect. Pap.*, vol. 9, p. 112, 2022.

[23] F. Smarandache, *Operators for Uncertain Over/Under/Off-Sets/-Logics/-Probabilities/-Statistics*. Infinite Study, 2025.