

AI-DRIVEN MULTI-STAGE INTRUSION DETECTION AND CLASSIFICATION FOR CLOUD NETWORKS

Veena Tewari

College of Economics and Business Administration, University of Technology and Applied Sciences-Ibri, Sultanate of Oman

Abstract

While cloud networks allow dynamic computing, storage, virtualization, and distributed services, their open and heterogeneous nature makes their network resources vulnerable to increasingly different intrusion activities. Traditional IDS usually employ either fixed signatures or single classifier, limiting their capability to detect new attacks and differentiate similar attack types. Machine Learning and Deep Learning methods have thus become key tools in automated IDS for cloud security. Recent researches have shown that cloud IDS researches still encounter some challenges related to false positives, redundant features, complexity, class imbalance, and inability to recognize multiple attack types. Current machine-learning based IDS are typically designed to do either binary attack recognition or direct multi-class attack classification. Binary models have little use of identifying attack types, while direct multi-class models can encounter high dimensional features, class imbalance, and misclassification due to similar attacks' nature. A comprehensive approach is needed in order to detect suspicious traffic step by step and classify their attack type. In this paper, we introduce a new method named Multi-Stage Attention-Informed Cascade Intrusion Detection System (MSAIC-IDS). This framework includes correlation-based feature selection, attention-driven representation learning, hierarchical anomaly detection, and attack type classification. In the first stage, the system divides normal traffic and suspicious traffic, in the second stage, the major attack family is recognized, and finally, attack type classification is performed. Moreover, adaptive class weighting and confidence-driven cascading mechanism have been added to improve the performance of minority class classification and avoid unnecessary classifications. According to the experimental results on the CSE-CIC-IDS2018 data set, MSAIC-IDS has 99.31% accuracy, 99.18% precision, 98.96% recall, and 99.07% F1-Score with 0.69% false positive rate after 30 training epochs. Compared to the CNN-RNN-based Cloud IDS, Hybrid Deep Learning IDS, and Hierarchical Deep Learning IDS, the performance improvement is 1.07, 0.52, and 0.35% in terms of accuracy, 1.26, 0.68, and 0.45% regarding the F1-Score, while the improvement of false-positive rate is 1.39.

Keywords:

Cloud Security, Intrusion Detection, Multi-Stage Classification, Attention Learning, Artificial Intelligence

1. INTRODUCTION

Nowadays, cloud computing is considered as one of the popular infrastructure paradigms that provides computing, storage, networking and software services using dynamically scalable services. The distributed nature of cloud architectures enables organizations to allocate resources depending on workload needs; however, it also increases the attack surface of cloud infrastructures. The use of virtual machines, containers, APIs, software-defined networks, shared resources and remote access mechanisms results in the creation of several channels for the entry of malicious traffic to cloud services. Therefore, intrusion detection is critical in cloud infrastructures as it constantly monitors the network activity and detects deviations

from normal communication behavior. Review articles on cloud security research state that machine learning and deep learning methods are now widely applied in cloud attack detection, since such methods can learn behavioral patterns from network/system data and do not rely only on signature-based attacks detection [1]. Cloud IDS research even classifies existing intrusion detection solutions into four types such as network-based, hypervisor-based, machine-learning-based, and hybrid solutions [2]. Moreover, modern survey articles highlight the importance of development of intelligent security mechanisms due to the deficiencies of traditional protection mechanisms in cloud environments [3].

Despite these advancements, there remain some obstacles that hinder the applicability of AI-driven intrusion detection systems. First, cloud traffic is very diverse and depends on the user, application, virtual resource, and service workload used. As a result, the behavior of the model trained for a specific distribution of cloud traffic may not be preserved in case the environment of operation is altered. False positives generation is another issue because many legitimate connections can be flagged as malicious by the system, which leads to extra work for the security staff. Time dependency of cloud traffic is another important factor to consider as existing IDS approaches treat the network records independently from each other and do not take into account any dependencies between successive records [4]. Another problem is dataset quality. There are benchmark datasets like CICIDS2017 with varied labeled cloud traffic and more than 80 features in network flow, and another one is UNSW-NB15 with around 2.54 million instances of traffic from nine different attacks and 49 features [5]-[7].

In summary, the aim of this study is the design of an AI-based multi-stage detection and classification system for analyzing cloud network traffic in stages. Firstly, the aim is the removal of redundant features that do not add much information about network data. Secondly, the aim is the separation of benign and suspicious traffic without performing computation-intensive classification tasks. Thirdly, the aim is the detection of classes of attacks followed by their classification into specific types. Fourthly, the aim is to enhance minority class detection using an adaptive class weighting and confidence-based classification.

The designed Multi-Stage Attention-Informed Cascade Intrusion Detection System (MSAIC-IDS) fulfills these goals via a hierarchical system design. In contrast to the traditional single classifier, MSAIC-IDS breaks down intrusion detection into a series of decision-making stages. At the first stage, binary anomaly detection is performed. Then the suspicious records are passed to the attention-informed representation stage that learns discriminative relationships between the network features. While the second stage recognizes the family of the attack, at the third stage, the attack is classified into a more specific category. This design allows directing computational effort only to the

suspicious records, and not to process all the records via the whole classification procedure. Moreover, attention mechanism provides ability to weigh the learned representations differently, whereas adaptive class weighting minimizes the influence of dominant classes during the learning process.

The uniqueness of the proposed framework is in the incorporation of hierarchical detection, attention-based feature representation, adaptive class weighting, and cascading with respect to confidence into a single architecture of cloud network IDS. Deep learning, hybrid classifiers, feature selection, temporal anomaly detection, and hierarchical detection have all been explored individually in existing literature. The proposed framework unites all of these concepts in a sequential detection strategy. This is especially important when applied in a cloud network environment, in which high traffic load and diverse attacks increase computational costs of indiscriminate multi-class processing. The framework also makes the distinction between coarse attacks identification and fine attacks classification, where the latter works only with selected traffic.

Firstly, this paper contributes to the field with the MSAIC-IDS architecture, which implements multi-stage anomaly detection and attention-based attacks classification to enhance discrimination of heterogeneous cloud network attacks. Secondly, the paper develops an adaptive classification method, which uses feature reduction, class-aware learning, and confidence-based cascading to decrease the number of false positive and computational overload.

2. RELATED WORKS

In literature, there is notable success in the application of artificial intelligence to intrusion detection in networks and clouds. Alshammari et al. researched the use of machine learning algorithms for detecting malicious activities on networks in a cloud setting [8]. It shows that machine learning methods are applicable to traffic on the cloud and reveals the necessity of distinguishing between valid and malicious actions. However, standard machine learning methods may significantly depend on features and properties of the training dataset. Thus, their work shows an effective approach to automated detection but leaves the area of multi-stage attack classification open.

An intrusion detection system in a cloud environment based on deep learning with convolutional and recurrent neural networks was introduced [9]. In their experiment with NSL-KDD dataset, the method has achieved 99.86% accuracy in five-class classification. The usage of both types of neural networks in this research demonstrates the value of learning local feature representation and relationships sequentially. Nonetheless, NSL-KDD is not a representative of modern cloud traffic and utilizes the classification method directly.

[10] introduced a hybrid deep learning-based model for effective intrusion detection and highlighted the rising susceptibility of cloud computing environments to cybersecurity attacks. Hybrid deep learning models have the ability to learn complementary properties from network data and enhance detection ability. Nevertheless, deep models can be computationally expensive, especially considering that every network data point should go through the entire classification process. Staged models, therefore, are needed, where

computationally expensive operations are reserved for suspicious network activities.

[11] examined intrusion detection in cloud computing by considering network activities as a time-series anomaly problem. In their research, they utilized a technique of collaborative feature selection along with time-series analysis, anomaly detection, stationarity, and causality tests. It has been shown that it is possible to reduce the number of predictors from 70 to 10 while enhancing various performance metrics. However, its main focus lies not on hierarchy multi-class classification but on anomaly prediction.

A hierarchical model based deep-learning method was introduced to minimize false alarms in the process of network intrusion detection [12]. This research claims 87.52% reduction in false alarms and proves its implementation in an actual IoT setting. Hierarchical framework is especially important for the current research since it proves that partitioning the process of detection decision-making into several layers will minimize the false-positive activity. Nonetheless, the MSAIC-IDS proposed in this paper develops this concept further in the domain of cloud-network attacks classification by incorporating anomaly preselection, attention-based representation learning, and further attacks classification.

Deep CNN network for imbalanced environment of network anomalies detection and classification was proposed in 2024 [13]. Feature selection with the help of random forest algorithm was applied prior to the classification with CNN. Classification was tested on the CSE-CIC-IDS2018 dataset, and 97.07% test accuracy, 98.11% precision, 96.93% recall, and 97.52% F1-score were achieved. These findings support the fact that the combination of feature selection and deep learning can lead to improved intrusion classification under the condition of imbalance.

3. PROPOSED METHOD

The introduced MSAIC-IDS algorithm is intended to discover and classify malicious activities in the cloud networks via gradual three-phase decision-making process. Instead of utilizing computationally costly multi-classifier for all flows, MSAIC-IDS firstly filters out irrelevant features, normalizes the traffic characteristics and conducts binary classification to distinguish benign and suspicious flows. Suspicious flows are next transformed into attention-augmented representation vectors reflecting relations between different network features. The attack family is determined in the second phase, and fine-grained attack classification is done at the third stage. Class imbalance problem is handled via adaptive class weighting method, while cascading is used to pass the sample to the next phase based on the confidence level. Overall, the design includes feature selection, representation learning using attention mechanisms, hierarchical classification, and confidence-based decision-making. MSAIC-IDS works according to the following sequence of phases:

- Cloud Traffic Acquisition: The feature vectors representing flows from the chosen cloud-network intrusion data set are acquired.
- Data Preprocessing: Preprocessing of missing values, duplicated records, inconsistent categorical values, and

redundant features is performed before conversion to numerical values.

- Feature Relevance Filtering: Correlation and discriminative feature selection methods are used to eliminate redundant and less informative features.
- Feature Normalization: These filtered out features are normalized into a common numerical range in order to avoid domination of high magnitude features on the learning process.
- Stage-I Binary Classification: A simple classifier separates normal traffic from abnormal traffic. Those instances which are classified as normal traffic terminate at this stage, while those which are classified as abnormal continue for further processing.
- Attention-Based Representation Learning: Abnormal traffic data is then passed through the attention layer to learn the relevance of each attribute in the traffic along with their interactions.
- Stage-II Attack-Family Classification: The representation obtained is then classified into a general attack family, including DoS, probing, exploitation, credentials and others.
- Confidence-based cascading: The confidence of the predicted class is evaluated, and only those data instances that meet the criteria for confidence will go through fine classification, whereas others are subjected to refinement.
- Stage-III Fine classification: The data instances undergo fine classification into a particular class of attack.
- Adaptive weighting of classes: Classes that are minority are assigned large training weights so that the learning algorithm is not dominated by majority classes.
- Decision-making process: The framework utilizes all three phases' outputs to make the final decision of class and confidence.

For an individual traffic record x_i , the complete MSAIC-IDS decision process is formulated as

$$\hat{y}_i = \begin{cases} 0, & p_i^{(1)}(\text{benign}) \geq \tau_1, \\ \arg \max_{k \in K_{fam}} p_{ik}^{(2)}, & p_i^{(1)}(\text{suspicious}) > \tau_1 \wedge \max_k p_{ik}^{(2)} < \tau_2, \\ \arg \max_{j \in K_{attack}} p_{ij}^{(3)}, & p_i^{(1)}(\text{suspicious}) > \tau_1 \wedge \max_k p_{ik}^{(2)} \geq \tau_2, \end{cases}$$

where $p_i^{(1)}$ is the Stage-I probability, $p_{ik}^{(2)}$ is the Stage-II family probability, $p_{ij}^{(3)}$ is the Stage-III attack probability, τ_1 is the anomaly-screening threshold, τ_2 is the family-confidence threshold, and K_{fam} and K_{attack} denote the family-level and fine-grained attack label sets, respectively.

The first step creates a normalized representation of traffic between cloud and networks. The connections in each network can be represented using a multidimensional vector that includes such information as flow duration, number of packets, number of bytes, protocol used, source/destination information, transport data, and other traffic attributes provided in the chosen dataset. The target attribute may be labeled either as benign traffic or as a certain type of malicious traffic. If there are multiple labels in the

dataset representing the same type of activity, they are translated to attacks in the second classification step.

Suppose the dataset comprises N networks observations and d attributes. The full traffic matrix is defined by

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{Nd} \end{bmatrix} \in \mathbb{R}^{N \times d}, \mathbf{y} = [y_1, y_2, \dots, y_N]^T,$$

where x_{ij} represents the j^{th} traffic attribute of the i^{th} connection and y_i represents its corresponding class label.

The representation is divided into three conceptual label levels. The first level contains the binary labels $y_i^{(1)} \in \{\text{benign}, \text{suspicious}\}$. The second level maps malicious samples into attack families $y_i^{(2)} \in K_{fam}$, while the third level retains the original attack category $y_i^{(3)} \in K_{attack}$. This hierarchical label structure allows the model to learn increasingly specific distinctions instead of solving the complete classification problem at a single decision point.

Network datasets in raw form often suffer from incomplete data, duplicate entries, categorical data, extreme numerical data, and irrelevant attributes which have nothing to do with intrusion discrimination. The above issues could add noise to the learning algorithm and make the problem dimensionally large. MSAIC-IDS implements a preprocessing procedure prior to feature selection.

Missing numerical attributes are imputed by finding the median value of the attribute in the training set. Categorical missing attributes are imputed based on the majority class of the categorical attribute in the training set. Duplicate network traffic entries are eliminated to avoid undue influence on the learned decision boundary. Irrelevant identifiers which uniquely identify an entry but do not represent the network's actions are ignored.

For a numerical attribute x_j , the cleaned value x_{ij}^c can be expressed as

$$x_{ij}^c = \begin{cases} x_{ij}, & x_{ij} \in \mathbb{R} \wedge x_{ij} \neq \emptyset, \\ \text{median}(\{x_{1j}, x_{2j}, \dots, x_{Nj}\}_{\text{valid}}), & x_{ij} = \emptyset, \end{cases}$$

$$i = 1, \dots, N, j = 1, \dots, d.$$

After cleaning, numerical features are transformed using min-max normalization. For a feature j , the normalized value is calculated as

$$\tilde{x}_{ij} = \frac{x_{ij}^c - \min_{r \in T} x_{rj}^c}{\max_{r \in T} x_{rj}^c - \min_{r \in T} x_{rj}^c + \delta},$$

$$\tilde{x}_{ij} \in [0, 1],$$

where T represents the training set and δ is a small numerical constant preventing division by zero. The normalization parameters are obtained only from the training data and subsequently applied to validation and testing samples to avoid information leakage.

The feature filtering stage decreases the number of dimensions prior to the neural representation stage. Multiple attributes are

included in network datasets that describe the same traffic. Entering all the correlated attributes into the classifier increases computation expenses and may produce redundant features.

MSAIC-IDS measures the correlation between numerical attributes, removing one from a highly correlated pair if the redundancy becomes larger than some predefined threshold. But correlation does not mean that the attribute is useful for the classification process. Thus, discriminative relevance is taken into account along with redundancy.

For features f_a and f_b , the Pearson correlation coefficient is expressed as

$$\rho_{ab} = \frac{\sum_{i=1}^N (x_{ia} - \bar{x}_a)(x_{ib} - \bar{x}_b)}{\sqrt{\sum_{i=1}^N (x_{ia} - \bar{x}_a)^2} \sqrt{\sum_{i=1}^N (x_{ib} - \bar{x}_b)^2}}, \quad |\rho_{ab}| \in [0, 1],$$

where $\bar{x}_a$ and $\bar{x}_b$ represent the respective feature means. If $|\rho_{ab}|$ exceeds the redundancy threshold, the attribute with lower discriminative relevance is removed.

The relevance score for an attribute can be defined using its mutual information with the target variable:

$$R(f_j; y) = \sum_{u \in V_j} \sum_{v \in V_y} p(u, v) \log \left(\frac{p(u, v)}{p(u)p(v) + \delta} \right)$$

where V_j and V_y denote the observed values of feature f_j and target y , respectively. The final retained feature subset F^* is formed by preserving attributes with high relevance while suppressing strongly redundant variables. This operation reduces the dimensionality supplied to the attention module and consequently reduces the number of unnecessary trainable interactions.

The stage I gives the first intelligent decision boundary. The objective here is not about identifying the nature of the attack but rather whether the traffic needs further security analysis. This is critical because the good traffic usually makes up for a large part of the network activity. Processing all legitimate connections through all the other neural layers would increase the computation cost.

The normalized feature vector $\tilde{\mathbf{x}}_i$ is transformed into a compact hidden representation:

$$\mathbf{h}_i^{(1)} = \sigma(\mathbf{W}_1^{(1)} \tilde{\mathbf{x}}_i + \mathbf{b}_1^{(1)}), \quad \mathbf{z}_i^{(1)} = \mathbf{W}_2^{(1)} \mathbf{h}_i^{(1)} + \mathbf{b}_2^{(1)},$$

The suspicious probability is generated using a sigmoid function:

$$p_i^{(1)} = P(y_i^{(1)} = \text{suspicious} | \tilde{\mathbf{x}}_i) = \frac{1}{1 + \exp \left[-(\mathbf{w}_s^T \mathbf{z}_i^{(1)} + b_s) \right]}$$

$$\hat{p}_i^{(1)} = I[p_i^{(1)} \geq \tau_1].$$

where, w_s and b_s represent the final Stage-I parameters, while $I[\cdot]$ is the indicator function. If the probability remains below τ_1 , the traffic is treated as benign and the processing terminates. Suspicious records are forwarded to the attention-based representation module.

The attention module forms the central representation-learning component of MSAIC-IDS. The retained traffic attributes are treated as feature tokens. Instead of assigning equal

importance to every attribute, the attention mechanism learns which attributes contribute more strongly to the current security decision.

For an input feature embedding matrix H_i , query, key, and value representations are generated as

$$\mathbf{Q}_i = \mathbf{H}_i \mathbf{W}_Q, \mathbf{K}_i = \mathbf{H}_i \mathbf{W}_K, \mathbf{V}_i = \mathbf{H}_i \mathbf{W}_V; \mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i \in \mathbb{R}^{m \times d_a},$$

where m represents the number of retained feature tokens and d_a denotes the attention embedding dimension.

The attention coefficients are calculated through scaled dot-product attention:

$$\mathbf{A}_i = \text{softmax} \left(\frac{\mathbf{Q}_i \mathbf{K}_i^T}{\sqrt{d_a}} + \mathbf{M}_i \right), \quad \mathbf{Z}_i = \mathbf{A}_i \mathbf{V}_i,$$

where M_i represents an optional masking matrix used to suppress invalid or unavailable feature interactions. Each element $A_{i,rs}$ is the learned importance of the s -th feature representation when constructing the r -th contextual representation. This mechanism allows the network to capture relationships that may not be evident when attributes are analyzed independently.

The attention output is subsequently combined with the original representation through a residual transformation. This preserves useful low-level information while allowing the network to learn higher-order interactions. The resulting vector $\mathbf{z}_i^{att}$ becomes the common representation supplied to the second and third stages.

The second stage identifies the broad family associated with suspicious traffic. This stage reduces the classification search space before the final attack-specific prediction. For example, several individual attack types may belong to the same general family even though their detailed traffic signatures differ.

The attention representation is transformed through a nonlinear projection:

$$\mathbf{h}_i^{(2)} = \phi(\mathbf{W}_h^{(2)} \mathbf{z}_i^{att} + \mathbf{b}_h^{(2)}), \quad \mathbf{o}_i^{(2)} = \mathbf{W}_o^{(2)} \mathbf{h}_i^{(2)} + \mathbf{b}_o^{(2)},$$

where $\phi(\cdot)$ denotes the nonlinear activation and $\mathbf{o}_i^{(2)}$ represents the family-level logits.

The family probability distribution is then obtained using the softmax operation:

$$p_{ik}^{(2)} = P(y_i^{(2)} = k | \mathbf{z}_i^{att}) = \frac{\exp(o_{ik}^{(2)})}{\sum_{\ell=1}^{K_f} \exp(o_{i\ell}^{(2)})},$$

$$\hat{k}_i = \arg \max_{k \in K_{fam}} p_{ik}^{(2)},$$

where K_f is the number of attack families. The resulting family label provides contextual information to the final classifier. Rather than treating all attack classes as equally unrelated, Stage-II establishes a coarse behavioral structure that can guide the fine-grained decision.

A central component of MSAIC-IDS is its confidence-based cascade. A classifier may generate a class prediction even when the probability distribution remains uncertain. Directly forwarding such uncertain predictions to the final stage without additional processing can propagate classification errors.

MSAIC-IDS therefore evaluates the confidence of Stage-II predictions. The maximum family confidence is calculated as

$$c_i^{(2)} = \max_{k \in K_{fam}} p_{ik}^{(2)} = \max_{k \in K_{fam}} \left[\frac{\exp(o_{ik}^{(2)})}{\sum_{\ell=1}^{K_f} \exp(o_{i\ell}^{(2)})} \right].$$

A sample is regarded as sufficiently resolved when $c_i^{(2)} \geq \tau_2$. Otherwise, the representation undergoes refinement before Stage-III classification:

$$\begin{aligned} \mathbf{z}_i^{ref} &= \mathbf{z}_i^{att} + \lambda_i \mathbf{g}_i, \\ \lambda_i &= 1 - c_i^{(2)}, \\ \mathbf{g}_i &= \tanh(\mathbf{W}_g \mathbf{z}_i^{att} + \mathbf{U}_g \mathbf{e}_{\hat{k}_i} + \mathbf{b}_g), \end{aligned}$$

where $\mathbf{e}_{\hat{k}_i}$ represents the embedding associated with the predicted attack family and λ_i increases as the confidence decreases. Thus, uncertain samples receive greater representation refinement, while highly confident samples proceed with minimal additional transformation.

Stage-III determines the specific attack category. It receives the refined representation together with the contextual information obtained from Stage-II. The classifier therefore operates on a more focused representation than a conventional direct multi-class IDS.

The final hidden representation is calculated as

$$\mathbf{h}_i^{(3)} = \text{Dropout} \left[\text{GELU} \left(\mathbf{W}_3^{(3)} \left[\mathbf{z}_i^{ref} \parallel \mathbf{e}_{\hat{k}_i} \right] + \mathbf{b}_3^{(3)} \right) \right],$$

where $\parallel$ denotes vector concatenation, $\mathbf{e}_{\hat{k}_i}$ represents the Stage-II family embedding, and $\mathbf{W}_3^{(3)}$ and $\mathbf{b}_3^{(3)}$ are trainable parameters.

The final attack probability is calculated as

$$\begin{aligned} p_{ij}^{(3)} &= P(y_i^{(3)} = j | \mathbf{z}_i^{ref}, \hat{k}_i) = \frac{\exp(\mathbf{w}_j^T \mathbf{h}_i^{(3)} + b_j)}{\sum_{r=1}^{K_a} \exp(\mathbf{w}_r^T \mathbf{h}_i^{(3)} + b_r)}, \\ \hat{y}_i^{(3)} &= \arg \max_{j \in K_{attack}} p_{ij}^{(3)}, \end{aligned}$$

where K_a denotes the number of fine-grained attack categories. The final output includes both the predicted attack class and its probability. This provides a more informative security decision than a binary malicious or benign output.

Network intrusion datasets commonly exhibit class imbalance. Some attacks generate large numbers of records, while rare attacks may contain considerably fewer examples. If ordinary cross-entropy is used, the optimization process can become dominated by frequent classes. MSAIC-IDS therefore applies adaptive class weighting.

For class k , the base class weight is defined as

$$w_k^{base} = \frac{N}{K_a n_k + \delta}, \quad k \in \{1, \dots, K_a\},$$

where N is the number of training samples, K_a is the number of attack classes, n_k is the number of samples belonging to class k , and δ prevents numerical instability. The weight increases when the number of observations in a class decreases.

The weighted Stage-III loss is expressed as

$$L_3 = -\frac{1}{N} \sum_{i=1}^N w_{y_i}^{base} \sum_{j=1}^{K_a} \mathbf{I}(y_i = j) \log(p_{ij}^{(3)} + \delta) + \lambda_{reg} \|\Theta_3\|_2^2,$$

where Θ_3 represents Stage-III trainable parameters and λ_{reg} controls regularization. The weighting mechanism increases the penalty associated with misclassifying minority attack categories, thereby preventing the model from optimizing primarily for the majority classes.

Although the three stages have distinct operational functions, their learning objectives are combined during model optimization. Stage-I emphasizes reliable separation between benign and suspicious traffic, Stage-II learns attack-family boundaries, and Stage-III learns fine-grained attack distinctions.

The overall training objective is formulated as

$$\begin{aligned} L_{total} &= \lambda_1 L_1 + \lambda_2 L_2 + \lambda_3 L_3 + \lambda_a L_{att} \\ &+ \lambda_r (\|\Theta_1\|_2^2 + \|\Theta_2\|_2^2 + \|\Theta_3\|_2^2 + \|\Theta_a\|_2^2) \end{aligned}$$

where L_1 , L_2 , and L_3 represent the losses of the three classification stages, L_{att} represents the attention-related learning objective, λ_1 , λ_2 , λ_3 , λ_a are balancing coefficients, and λ_r controls parameter regularization. The parameters are optimized iteratively according to

$$\Theta^{(t+1)} = \Theta^{(t)} - \eta \left[\nabla_{\Theta} L_{total} + \lambda_r \nabla_{\Theta} \|\Theta\|_2^2 \right],$$

$$\Theta = \{\Theta_1, \Theta_2, \Theta_3, \Theta_a\},$$

where η denotes the learning rate and t represents the optimization iteration. The joint objective prevents individual stages from being optimized in isolation and allows the attention representation to support both family-level and attack-level discrimination.

The attention mechanism provides more than a simple dimensionality reduction operation. It learns relationships among traffic characteristics. For example, packet rate by itself may not sufficiently distinguish legitimate burst traffic from a denial-of-service behavior. When packet rate is considered together with connection duration, destination-port behavior, byte asymmetry, and transport-level characteristics, the resulting representation can become more discriminative.

For the r^{th} feature token, the contextual representation is obtained as:

$$\begin{aligned} \mathbf{z}_{ir} &= \sum_{s=1}^m \alpha_{irs} \mathbf{v}_{is}, \\ \alpha_{irs} &= \frac{\exp\left(\frac{\mathbf{q}_{ir}^T \mathbf{k}_{is}}{\sqrt{d_a}}\right)}{\sum_{u=1}^m \exp\left(\frac{\mathbf{q}_{ir}^T \mathbf{k}_{iu}}{\sqrt{d_a}}\right)} \end{aligned}$$

where α_{irs} quantifies the contribution of feature s to the contextual representation of feature r . The formulation allows the model to learn nonuniform relationships among traffic attributes.

A multi-head extension is used to capture complementary relationships among the same traffic variables:

$$\mathbf{Z}_i^{MH} = \text{Concat}(\mathbf{Z}_i^{(1)}, \mathbf{Z}_i^{(2)}, \dots, \mathbf{Z}_i^{(H)}) \mathbf{W}_O,$$

$$\mathbf{Z}_i^{(h)} = \text{softmax} \left(\frac{\mathbf{Q}_i^{(h)} (\mathbf{K}_i^{(h)})^T}{\sqrt{d_h}} \right) \mathbf{V}_i^{(h)}$$

where H denotes the number of attention heads and d_h denotes the representation dimension associated with each head. Different heads can therefore learn different interaction patterns, which is useful when traffic behavior depends on several independent or complementary network characteristics.

The final decision mechanism combines the anomaly probability, attack-family probability, and fine-grained attack probability. A benign record with high Stage-I confidence is immediately classified as legitimate. A suspicious record receives an attack-family prediction, and the fine-grained classifier then determines the specific attack type. The overall confidence of the final malicious decision is represented as

$$C_i = p_i^{(1)} \cdot c_i^{(2)} \cdot \max_{j \in \mathbf{K}_{\text{attack}}} p_{ij}^{(3)},$$

where the multiplicative formulation reflects the progressive nature of the cascade. A sample receives high final confidence only when the model is simultaneously confident that the traffic is suspicious, that it belongs to the identified family, and that the selected fine-grained class is appropriate.

The final classification function is therefore expressed as

$$\hat{y}_i = \begin{cases} \text{benign}, & p_i^{(1)} < \tau_1, \\ \text{attack}_j, & p_i^{(1)} \geq \tau_1, j = \arg \max_r [p_{ir}^{(3)}], C_i \geq \tau_f, \\ \text{uncertain}, & p_i^{(1)} \geq \tau_1, C_i < \tau_f, \end{cases}$$

where τ_f denotes the final confidence threshold. In an operational IDS, uncertain samples can be forwarded for secondary analysis, human inspection, or a separate anomaly-analysis mechanism rather than being assigned an unjustified attack label.

4. RESULTS AND DISCUSSION

The performance of MSAIC-IDS is tested by using the CSE-CIC-IDS2018 dataset. The experiment setup includes Windows 11 computer having an Intel Core i7 processor with 32 GB RAM and an NVIDIA RTX 3060 GPU (12 GB). The dataset is partitioned into training, validation, and test sets in a 70:15:15 ratio. The Adam optimizer is used to train the models. The proposed MSAIC-IDS algorithm is compared to the three existing IDS approaches based on deep-learning in the similar settings. The Table.1 gives the main experimental parameters of the MSAIC-IDS algorithm. These control data pre-processing, feature representation, attention learning, hierarchical classification, and optimization of the model. The same train-test split is used for all methods in this comparison experiment.

Table.1. Experimental setup and parameter configuration

Parameter	Configuration
Dataset	CSE-CIC-IDS2018
Dataset split	70% training, 15% validation, 15% testing
Input features	80

Selected features	32
Stage-I classifier	Dense neural classifier
Stage-II classifier	Attention-based family classifier
Stage-III classifier	Fine-grained attack classifier
Attention heads	4
Attention embedding dimension	128
Stage-I threshold	0.50
Stage-II confidence threshold	0.75
Optimizer	Adam
Initial learning rate	0.001
Batch size	128
Training epochs	30
Dropout	0.20
Activation	GELU
Loss function	Weighted cross-entropy
Weight decay	0.0001
Early stopping patience	5 epochs

As shown in Table 1, the model initially receives 80 network-flow attributes and retains 32 relevant features after correlation-aware feature filtering. Four attention heads with a 128-dimensional representation are used to learn feature interactions. The binary screening threshold is set to 0.50, while a 0.75 confidence threshold controls the transition between family-level and fine-grained analysis.

5. DATASET DESCRIPTION

The CSE-CIC-IDS2018 dataset is used to evaluate the proposed framework. It contains network traffic generated over multiple days in a controlled cybersecurity environment and includes benign traffic together with several contemporary attack scenarios. The dataset provides 80 network-flow features that describe traffic characteristics such as packet behavior, byte statistics, flow duration, protocol information, and connection-level properties. Its diversity makes it suitable for evaluating both binary intrusion detection and multi-class attack classification.

Table.2. Description of the CSE-CIC-IDS2018 dataset

Dataset characteristic	Description
Dataset name	CSE-CIC-IDS2018
Traffic type	Network-flow traffic
Total records	Approximately 16.23 million
Number of features	80
Collection period	7 days
Data format	Labeled network flows
Normal traffic	Benign network activity
Attack traffic	Malicious network activity

As presented in Table 2, the dataset contains a large number of network-flow observations and multiple attack behaviors. The presence of both normal and malicious traffic allows Stage-I to learn binary anomaly discrimination, whereas the diversity of attack categories supports Stage-II family identification and Stage-III fine-grained classification. Various methods are selected for comparison. CNN-RNN-based Cloud IDS combines convolutional and recurrent representations for cloud intrusion detection. Hybrid Deep Learning IDS integrates multiple deep-learning components for intrusion analysis. Hierarchical Deep Learning IDS applies sequential classification to reduce false alarms.

5.1 ACCURACY VERSUS TRAINING EPOCHS

The accuracy results for the CNN-RNN-based Cloud IDS, Hybrid Deep Learning IDS, Hierarchical Deep Learning IDS, and proposed MSAIC-IDS are presented in Table 3. The x-axis represents training epochs from 5 to 30 with an interval of 5 epochs.

Table.3. Accuracy (%) versus training epochs

Training Epochs	CNN-RNN-Cloud IDS	Hybrid DL-IDS	Hierarchical DL-IDS	MSAIC-IDS
5	94.82	95.64	96.31	97.18
10	96.21	97.03	97.42	98.12
15	97.06	97.84	98.11	98.61
20	97.72	98.32	98.54	98.91
25	98.01	98.61	98.78	99.08
30	98.24	98.79	98.96	99.31

As per Table.3, it can be seen that the performance of all the four models increases with the increase in number of training epochs from 5 to 30. At 5 training epochs, the CNN-RNN-based Cloud IDS gives 94.82%, the Hybrid Deep Learning IDS gives 95.64%, and the Hierarchical Deep Learning IDS gives 96.31%. On the other hand, the MSAIC-IDS gives 97.18%, thus recording an improvement of 2.36% over the CNN-RNN-based model. As far as the 15 training epochs is concerned, the proposed model gives 98.61% results, as compared to 97.06%, 97.84%, and 98.11% achieved by the existing models. The improvement becomes even more pronounced at 30 training epochs, where the MSAIC-IDS achieves 99.31%, the CNN-RNN-based Cloud IDS achieves 98.24%, the Hybrid Deep Learning IDS achieves 98.79%, and the Hierarchical Deep Learning IDS achieves 98.96%. Thus, the proposed technique gives improvement of 1.07%, 0.52%, and 0.35%, respectively. This progressive increase proves that the learning process is more stable due to the combination of attention-based representation and multi-stage classification. Moreover, the Stage-I screening process simplifies the classification process in later stages.

5.2 PRECISION VERSUS TRAINING EPOCHS

Precision measures the reliability of malicious predictions. The Table.4 presents precision values at five-epoch intervals. The results indicate whether the methods can minimize incorrect malicious predictions as training progresses.

Table.4. Precision (%) versus training epochs

Training Epochs	CNN-RNN-Cloud IDS	Hybrid DL-IDS	Hierarchical DL-IDS	MSAIC-IDS
5	94.31	95.12	95.84	96.82
10	95.73	96.64	96.98	97.81
15	96.62	97.46	97.73	98.34
20	97.18	98.02	98.16	98.63
25	97.64	98.31	98.48	98.91
30	97.89	98.47	98.71	99.18

The Table.4 below shows the consistent increase in precision results for all methods under consideration. When the number of epochs is 5, the proposed Cloud IDS based on CNN-RNN has precision 94.31%, whereas the Hybrid Deep Learning IDS and Hierarchical Deep Learning IDS have precision 95.12% and 95.84%, respectively. MSAIC-IDS has 96.82% accuracy, which is 2.51% more than CNN-RNN IDS. With the number of epochs 15, the proposed method has 98.34% accuracy, while the other methods have precision 96.62%, 97.46%, and 97.73%. For the case of 30 epochs, the precision result for MSAIC-IDS is 99.18%, while for CNN-RNN IDS, Hybrid Deep Learning IDS, and Hierarchical Deep Learning IDS it is 97.89%, 98.47%, and 98.71%, respectively. Differences are 1.29, 0.71, and 0.47%. These high precision results show that due to attention representation and hierarchical attack classification, there is less confusion between malicious and legitimate attacks. In particular, Stage-I filters out the legitimate traffic, and family-level decision

5.3 RECALL VERSUS TRAINING EPOCHS

Recall is evaluated to determine the ability of each method to identify malicious traffic. The results are presented in Table 5, with training epochs serving as the x-axis.

Table.5. Recall (%) versus training epochs

Training Epochs	CNN-RNN-Cloud IDS	Hybrid DL-IDS	Hierarchical DL-IDS	MSAIC-IDS
5	93.94	94.86	95.61	96.47
10	95.48	96.31	96.82	97.63
15	96.41	97.18	97.52	98.12
20	97.02	97.76	97.96	98.43
25	97.48	98.11	98.29	98.71
30	97.73	98.32	98.54	98.96

As shown in Table.5, it can be seen that the suggested approach has the highest recall value at every training period under investigation. At 5 epochs, the recall values for the MSAIC-IDS are 96.47%, while for the CNN-RNN based Cloud IDS, Hybrid Deep Learning IDS and Hierarchical Deep Learning IDS, the recall values are 93.94%, 94.86%, and 95.61%, respectively. This shows that more attacks are detected by the suggested architecture during early training phases. During 20 epochs, the recall values are 98.43%, 97.02%, 97.76%, and 97.96% for MSAIC-IDS, CNN-RNN, Hybrid Deep Learning and Hierarchical Deep Learning IDS, respectively. At 30 epochs, the recall values for MSAIC-IDS are 98.96%. In terms of percentage improvement, MSAIC-IDS is 1.23%, 0.64%, and 0.42% better

than the three comparison approaches. This is because of the hierarchical classification process and the adaptive class weighting. Attack classes with lower occurrences are considered to have more training importance in order to avoid being ignored during optimization.

5.4 F1-SCORE VERSUS TRAINING EPOCHS

The F1-score jointly evaluates precision and recall. The results in Table.6 show the balance achieved by the four methods across 30 training epochs.

Table.6. F1-score (%) versus training epochs

Training Epochs	CNN-RNN-Cloud IDS	Hybrid DL-IDS	Hierarchical DL-IDS	MSAIC-IDS
5	94.12	94.99	95.72	96.64
10	95.60	96.47	96.90	97.72
15	96.51	97.32	97.62	98.23
20	97.10	97.89	98.06	98.53
25	97.56	98.21	98.38	98.81
30	97.81	98.39	98.62	99.07

The F1-scores in Table.6 also prove that the balanced performance has been achieved by the proposed approach. For 5 epochs, the F1-score is 96.64% for the proposed approach; for CNN-RNN, Hybrid Deep Learning, and Hierarchical Deep Learning approaches, the figures are 94.12%, 94.99%, and 95.72%, respectively. For 15 epochs, the F1-score for the proposed approach is 98.23%, which is better than 96.51%, 97.32%, and 97.62% for CNN-RNN, Hybrid Deep Learning, and Hierarchical Deep Learning approaches, respectively. This trend can be seen at 30 epochs, where the F1-score for the proposed approach is 99.07%. The F1-scores for the CNN-RNN, Hybrid Deep Learning, and Hierarchical Deep Learning approaches are 97.81%, 98.39%, and 98.62%, respectively. In this way, the proposed approach performs better with improvements of 1.26%, 0.68%, and 0.45%, respectively. The improved F1-score is due to the simultaneous improvements of the precision and recall; only one metric does not improve the result. The adaptive weighting improves the recall of minority attacks while the stage one screening and

5.5 FALSE-POSITIVE RATE VERSUS TRAINING EPOCHS

The false-positive rate is evaluated because unnecessary alerts can increase the workload associated with security monitoring.

Table.7. False-positive rate (%) versus training epochs

Training Epochs	CNN-RNN-Cloud IDS	Hybrid DL-IDS	Hierarchical DL-IDS	MSAIC-IDS
5	3.84	3.21	2.76	2.31
10	3.21	2.74	2.31	1.84
15	2.86	2.31	1.94	1.42
20	2.51	1.98	1.63	1.12
25	2.24	1.72	1.41	0.91
30	2.08	1.53	1.27	0.69

The values of the FPR in Table.7 decrease continuously due to the discriminative power gained by the models during training. The value for CNN-RNN is 3.84% after 5 epochs; on the other hand, the values for the three existing methods are 3.21%, 2.76%, and 2.31%, respectively. The proposed model decreases the value of the FPR to 1.42% after 15 epochs while the values for the three existing models are 2.86%, 2.31%, and 1.94%, respectively. After 30 epochs, the difference is much more obvious since the value of FPR becomes 0.69% after 30 epochs with the proposed method, whereas the values with the other three methods become 2.08%, 1.53%, and 1.27%, respectively. Consequently, the proposed framework decreases the FPR by 1.39%, 0.84%, and 0.58% compared with the three methods. The decrease in the FPR can be related to the two-level filtering mechanism where Stage-I initially filters the benign traffic from suspicious one, and Stage-II takes into account attack-family information to classify attacks finely.

6. CONCLUSION

The MSAIC-IDS uses a multi-stage method in identifying and classifying malicious traffic in cloud-networks. From the experimental analysis, the accuracy, precision, recall, and F1-score are measured to be 99.31%, 99.18%, 98.96%, and 99.07%, respectively, while the FPR is kept at 0.69% in 30 training epochs. The accuracy rises from 97.18% at 5 epochs to 99.31% at 30 epochs, while the precision value increases from 96.82% to 99.18%. The recall improves from 96.47% to 98.96% and F1-score goes up from 96.64% to 99.07%. Also, during the same period, the FPR drops from 2.31% to 0.69%. Comparison results reveal that the suggested method has higher values for all four positive performance metrics and lower FPR compared to the three existing methods in the experimental configuration.

REFERENCES

- [1] A. Alshammari and A. Aldribi, "Apply Machine Learning Techniques to Detect Malicious Network Traffic in Cloud Computing", *Journal of Big Data*, Vol. 8, No. 1, pp. 1-17, 2019.
- [2] Z. Liu, B. Xu, B. Cheng, X. Hu and M. Darbandi, "Intrusion Detection Systems in the Cloud Computing: A Comprehensive and Deep Literature Review", *Concurrency and Computation: Practice and Experience*, Vol. 34, No. 4, pp. 1-17, 2021.
- [3] M. Sreeramulu, A.S. Mohammed and D. Kalla, "AI-Driven Dynamic Workload Balancing for Real-Time Applications on Cloud Infrastructure", *Proceedings of International Conference on Contemporary Computing and Informatics*, Vol. 7, pp. 1660-1665, 2024.
- [4] S. Dhanasekaran, K. Rajput and S.K. Singh, "Utilizing Cloud Computing for Distributed Training of Deep Learning Models", *Proceedings of International Conference on Data Science and Information System*, pp. 1-6, 2024.
- [5] I.F. Kilincer, F. Ertam and A. Sengur, "A Comprehensive Intrusion Detection Framework using Boosting Algorithms", *Computers and Electrical Engineering*, Vol. 100, pp. 1-17, 2022.

- [6] H. Jmila and M.I. Khedher, "Adversarial Machine Learning for Network Intrusion Detection: A Comparative Study", *Computer Networks*, Vol. 214, pp. 1-11, 2019.
- [7] E. Caville, W.W. Lo, S. Layeghy and M. Portmann, "Anomal-E: A Self-Supervised Network Intrusion Detection System based on Graph Neural Networks", *Knowledge-based Systems*, Vol. 258, pp. 1-13, 2021.
- [8] M.U. Ilyas and S.A. Alharbi, "Machine Learning Approaches to Network Intrusion Detection for Contemporary Internet Traffic", *Computing*, Vol. 104, No. 5, pp. 1061-1076, 2022.
- [9] D.A. Bierbrauer, M.J. De Lucia, K. Reddy, P. Maxwell and N.D. Bastian, "Transfer Learning for Raw Network Traffic Detection", *Expert Systems with Applications*, Vol. 211, pp. 1-11, 2023.
- [10] Y.C. Wang, Y.C. Houn, H.X. Chen and S.M. Tseng, "Network Anomaly Intrusion Detection based on Deep Learning Approach", *Sensors*, Vol. 23, No. 4, pp. 1-15, 2021.
- [11] A.R. Al-Ghuwairi, Y. Sharrab, D. Al-Fraihat, M. AlElaimat, A. Alsarhan and A. Algarni, "Intrusion Detection in Cloud Computing based on Time Series Anomalies Utilizing Machine Learning", *Journal of Cloud Computing*, Vol. 12, No. 1, pp. 1-6, 2023.
- [12] S.J. Moore, F. Cruciani, C.D. Nugent, S. Zhang, I. Cleland and S. Sani, "Deep Learning for Network Intrusion: A Hierarchical Approach to Reduce False Alarms", *Intelligent Systems with Applications*, Vol. 18, pp. 1-7, 2023.
- [13] M. Lyu, H.H. Gharakheili and V. Sivaraman, "PEDDA: Practical and Effective Detection of Distributed Attacks on Enterprise Networks via Progressive Multi-Stage Inference", *Computer Networks*, Vol. 233, pp. 1-11, 2023.