ISSN: 2583-9888
DOI: doi.org/10.51219/JAIMLD/mohammed-mustafa-khan/255

URF PUBLISHERS
connect with research world

Journal of Artificial Intelligence, Machine Learning and Data Science

https://urfpublishers.com/journal/artificial-intelligence

Vol: 2 & Iss: 1

Research Article

Developing AI-Powered Intrusion Detection System for Cloud Infrastructure

Mohammed Mustafa Khan*

Citation: Khan MM. Developing AI-Powered Intrusion Detection System for Cloud Infrastructure. *J Artif Intell Mach Learn & Data Sci 2024*, 2(1), 1074-1080. DOI: doi.org/10.51219/JAIMLD/mohammed-mustafa-khan/255

Received: 02 February, 2024; **Accepted:** 18 February, 2024; **Published:** 20 February, 2024

***Corresponding author:** Mohammed Mustafa Khan, USA

ABSTRACT

Cloud computing has become an integral part of modern businesses and provides an avalanche of benefits such as flexibility, scalability, and cost-effectiveness. This has necessitated the adoption of cloud computing to support business structures, processes, procedures, and workflow operations. Nonetheless, the whooping reliance on cloud services to buffer business operations is crippled by cybersecurity threats since the traditional intrusion detection systems (IDS) scuffle with adapting to the dynamic and complex nature of cloud environments resulting in flaws and bugs in threat detection and response, therefore, calling for rapid establishment of the modern-day powerful security measures. Organizations want to secure their digital environment to avoid data breaches and cybercrime. Cyber-attacks have grown in sophistication and increased in number. It is a tremendous aspect to secure the cloud environment through detecting, protecting, and preventing cyber-attacks. This research paper trickles down into the development of an AI-powered Intrusion Detection System (IDS), which addresses the cloud infrastructure environment in relation to cybersecurity. The primary target focuses on using machine learning techniques to fortify cloud security. This can be done through an in-depth analysis of network traffic patterns. This helps identify unusual behaviors that indicate potential attacks on cloud servers and resources. The core approach to be utilized is the unsupervised learning methodology. The unsupervised learning methodologies focus on autoencoders, which helps boost anomaly detection accuracy within the cloud infrastructure. This research aims to demonstrate the efficacy of the proposed Intrusion and Detection System (IDS) in preventing and protecting cloud infrastructure against various cybercrimes. The integration of AI-driven anomaly detection mechanisms into the fabric of cloud security protocols can enhance and harden cloud security. By hardening the cloud infrastructure, it will remain resilient and reliable in case of any cybercrime. Furthermore, this paper acts as a roadmap on how machine learning can enhance cloud security in a cloud environment by providing insights and innovative ideas on mitigating all the threats and vulnerabilities of cybercrme in the digital economy.

Keywords: Machine learning, artificial intelligence, intrusion detection system, cloud infrastructure, network security, cybercrime

1. Introduction

The advent of cloud computing has streamlined the management and deployment of an organization's IT assets globally. Cloud computing technology has offered pervasive characteristics that provide scalability, resiliency, cost-effectiveness, and reliability of services. Despite the numerous benefits of cloud computing, it is hobbled by security challenges that must be addressed, and comprehensive measures ought to be established. Securing cloud infrastructure against cybercrime remains a challenge to many organizations. The existing traditional intrusion detection systems were primarily designed for on-premises platforms. Organizations are transitioning from on-premise solutions to cloud-based platforms to leverage the benefits of cloud services. The traditional IDS finds it challenging to adapt to the new technological aspects of cloud computing. Virtualization of networks and storage-defined networking makes it hard for traditional IDS to secure the cloud environment. The traditional IDS cannot effectively secure the cloud environment workloads. A comprehensive solution is needed to address these limitations and challenges. The solution

is to incorporate artificial intelligence into intrusion and detection systems that can enhance cloud security. The AI-powered IDS utilizes machine learning algorithms, which provide an effective solution to optimize cloud platform threat detection and response capacity[6].

By normalizing the machine learning models, the AI-powered IDS can effectively scrutinize network traffic and identify any malicious activities and operations in the network. The use of AI-driven IDS has revolutionized cloud security. The principal objective of using machine learning(ML) for anomaly detection is to establish models that correctly categorize network traffic as usual or malicious. It entails training models using labeled network traffic datasets and various features such as port numbers, protocol type, packet size, and source or destination IP addresses. These models can then be deployed in real-time to continuously monitor network traffic for all the end-point devices and generate notifications whenever anomalous behavior is detected. Organizations employing machine learning in network security will likely secure their IT assets from cyber threats to protect against emerging cyber threats before causing havoc. One of the benefits of this solution is that it can adapt to new cyber threats since it learns any new threat and alerts other systems for action to be taken. This research paper expounds on how machine learning techniques can promote the network security of cloud infrastructures. The paper focuses on how machine learning techniques can identify network traffic anomalies and allow threat detection.

2. Literature Review

Signature-based techniques were initially used to identify the known threats. Advancements in technology led to new security challenges, making it difficult for legacy intrusion detection methods to function appropriately, resulting in the adoption of AI-driven solutions to countermeasure the security challenges. Tashfeen & M. (2024) studied the signature-based intrusion detection system used in the identification of known threats and found limited efficacy in the modern cloud environment. The authors agreed that the traditional intrusion detection system for threat detection had some security issues, making it challenging to handle anomaly detection in networks for cloud platforms.

One of the applications of ML techniques is used to monitor or detect attacks and security challenges on the Cloud. According to Nassif, et al. (2021), ML techniques applied to intrusion detection systems for monitoring and detecting are practical solutions that address the security challenges in the cloud platform. The authors found unsupervised learning was the best ML method for network anomaly detection since it could detect unknown threats. The authors defined autoencoders as a type of neural network that uses unsupervised machine learning to support its operations. Autoencoders consist of two major parts: encoder and decoder. The encoder part accepts data as an input to perform compression, while the decoder part generates an output in the reconstruction process. The most tremendous aspect not to be forgotten is that the input data is compressed using a lower compressional representation, and the output is reconstructed. The reconstructed data will contain the errors that can be used to detect anomalies.

According to Zavrak, et al. (2020), the increase in network traffic is attributed to the fact that the security aspects of the network keep on improving day by day. Users do not have to worry about the security aspects of the network. This rise was pushed by the adoption of AI-powered intrusion detection systems to detect network anomalies. The authors based their study on anomaly-based methods that can find unknown attacks when integrated adequately with systems like cloud platforms. They performed a study to demonstrate the efficacy of autoencoders in intrusion detection. The authors found that employing the autoencoder-based intrusion detection system aids in detecting high levels of attacks. Their study became a game changer, leading to the utilization of autoencoders as one of the network intrusion detection tools. Autoencoders have proved to be effective in monitoring and detecting network attacks with low positive rates.

3. Methodology

3.1 Data Collection and Preprocessing

Autoencoders require data collection to function effectively. The data collection and preprocessing processes are the backbone of the analysis and model development. This section shows steps in acquiring and reforming network traffic data, thus ensuring it is suitable for subsequent analysis.

Data Collection

Network traffic consists of normal and unusual activities collected from cloud servers. This inclusion approach gathers all the information needed to understand network behavior and generate some patterns. The network routine operations are examined, which enables the identification of abnormal operations. The dataset entails diverse network patterns and anomalies by collecting data from the cloud server. Some problems that can arise from data collection are inaccurate data, data bias, missing data, and data imbalance. To curb these problems, web crawling and scraping techniques can be employed. These techniques utilize automation tools to clean data and pre-clean available data sets[2].

Data Preprocessing

This stage is crucial for fortifying data quality and its relevance. It consists of a triad process that entails cleaning, extraction, and normalization, as demonstrated in **Figure 1.**

a) **Cleaning:** This initial step entails cleansing the data to remove any discrepancies. The missing values, outliers, or corrupted data are identified and removed.

b) **Extraction:** This is the next step after data cleaning, which involves distilling salient characteristics from the raw data. The statistical measure of tendency, frequency domain analysis, and time series decomposition focuses on various aspects of network traffic[2].

c) **Normalization:** This marks the last step. The size of a dataset influences the processing and memory needed for iterations during training. Normalization lowers the size by lowering the data magnitude and order.

3.2 Model Selection: Autoencoders

Autoencoders is a neural network technique that uses unsupervised learning capabilities to access input data. There are different types of autoencoders: variational autoencoder, convolutional autoencoder, denoising autoencoder, and sparse autoencoder, to mention a few. This paper focuses on the general functioning of autoencoders and will not drill down into the specific type of autoencoder. An autoencoder entails two components, namely, the encoder and the decoder. These networks strive to compress input data into a lower dimensional latent space known as encoding and similarly reconstruct it back to the original format in a process known as decoding. The ability of autoencoders to learn various input data aligns well with its objective of anomaly detection[8]. An autoencoder is trained to copy its input and produce an output. The internal part of the autoencoder has a layer called h that defines a code that stands for input. As mentioned earlier, the network can be seen to contain two parts: the function of an encoder expressed as **h = f(x)** and a decoder that generates a reconstruction **r = g(h)**. In case an autoencoder manages to learn setting **g(f(x)) = x** everywhere, this signifies it is not of use. However, autoencoders are created in a way that they learn not to copy perfectly. Normally, they are limited in enabling them to copy approximately and are restricted in copying inputs that are similar to the training data. The model is designed in a way that prioritizes which factors of the input data must be copied and proceeds to learn the important characteristics of the data. The modern autoencoders have been designed with superiority in that it uses both deterministic functions and stochastic mappings known as $p_{encoder}$ **(h|x)** and $p_{decoder}$ **(x|h)**. The general structure of an autoencoder involves converting an input x into an output, known as the reconstruction **r**, via an internal representation or code **h**. This process consists of two main parts: the encoder **f**, which transforms **x** into **h**, and the decoder **g**, which links **h** back to **r**[7]. **Figure 2** shows the simple architecture of an autoencoder.

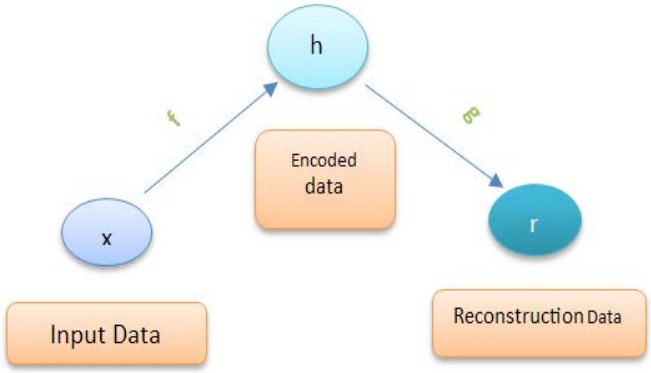

Figure 2: The General Architecture of an Autoencoder.

Experimental Evaluation

The success of anomaly detection methodologies depends on the value of training a model and the evaluation measures. This section shows the methodology approach that highlights the training and evaluation of the autoencoder. The autoencoder is trained to identify normal and unusual activities and how determine if it is doing its designated function.

Training the Model

Training an autoencoder involves adjusting its weights and biases through a process called backpropagation to minimize the difference between the input and the reconstructed output. Our autoencoder model's training phase entails optimizing model parameters to minimize the reconstruction fault between input data and their corresponding reconstructions. The autoencoder is fed with a wide range of network traffic. After learning and understanding the trends and patterns, the technique known as cross-validation is used to ensure the autoencoder has been trained successfully by memorizing the example and understanding every aspect[3].

Table 1: shows a general pseudocode for designing an autoencoder.

General Pseudocode showing how to Design an autoencoder for monitoring networks
Import necessary libraries import numpy as np import tensorflow as tf from tensorflow.keras.models import Model from tensorflow.keras.layers import Input, Dense **# Define the encoder model** def build_encoder(input_dim, encoding_dim): input_layer = Input(shape=(input_dim,)) encoded = Dense(encoding_dim, activation='relu')(input_layer) encoder = Model(input_layer, encoded) return encoder **# Define the decoder model** def build_decoder(encoding_dim, output_dim): encoded_input = Input(shape=(encoding_dim,)) decoded = Dense(output_dim, activation='sigmoid')(encoded_input) decoder = Model(encoded_input, decoded) return decoder **# Define the autoencoder model** def build_autoencoder(input_dim, encoding_dim): input_layer = Input(shape=(input_dim,)) encoder = build_encoder(input_dim, encoding_dim)

```
decoder = build_decoder(encoding_dim, input_dim)
 encoded = encoder(input_layer)
 decoded = decoder(encoded)
 autoencoder = Model(input_layer, decoded)
 return autoencoder

# Load and preprocess data
def load_data():
 # Load your dataset here (e.g., MNIST, CIFAR-10, etc.)
 # For demonstration, we'll use random data
 data = np.random.rand(1000, 784)
 return data
# Train the autoencoder
def train_autoencoder(autoencoder, data, epochs=50, batch_size=256):
 autoencoder.compile(optimizer='adam', loss='mean_squared_error')
 autoencoder.fit(data, data, epochs=epochs, batch_size=batch_size, shuffle=True, validation_split=0.2)
# Main function to execute the autoencoder training
def main():
 input_dim = 784 # Dimension of the input data
 encoding_dim = 32 # Dimension of the encoding layer
 # Build the autoencoder
 autoencoder = build_autoencoder(input_dim, encoding_dim)
 # Load the data
 data = load_data()
 # Train the autoencoder
 train_autoencoder(autoencoder, data)
if __name__ == "__main__":
 main()
```

Source of the pseudocode: https://www.tensorflow.org/tutorials/generative/autoencoder

Checking Performance

Some measurements, such as precision, recall, and F-scores, can be used to review the performance of the autoencoder. Precision demonstrates the program's accuracy in identifying network anomalies. The recall shows the true anomalies that the model detected successfully. The F1 score combines the two metrics to give an overview of how well the program functions. Training the autoencoder programs with lots of datasets and regularly validating their performance allows the program to function correctly as per the business requirements[14].

4. Results

Anomaly detection Paradigm

Network anomalies are the abnormal aspects within the network that deviate from normal behavior. Network anomalies are known to disrupt the normal operations of the network. Anomaly detection can be determined by identifying network traffic that does not conform to the existing patterns. There are three types of network anomalies, namely collective, point, and contextual [9]. Each anomaly has its state of occurrence, as discussed below:

Point anomaly is brought about by a single data point showing features different from the other data group. For example, a rapid increase in purchase transactions.

The contextual anomaly occurs when the information usually acts within a specific situation. This anomaly is connected to data that constantly changes over time. For instance, an increase in network traffic over a single night.

Collective anomaly happens when the data cluster shows an abnormal pattern compared to the rest of the dataset. In this type, any abnormality that is being conducted by an individual data point is not considered. For instance, a computer system with network congestion displays video content. Anomalies can be further grouped into two subgroups: security-related anomalies and performance-related anomalies. We focus on the security-related anomalies that happen because of malicious malware actions that are supposed to flood the network with unnecessary traffic, hijack the amount of data, and make the system accessible[9].

How Anomaly Detection Works

Anomalies can be detected by calculating if the fixed threshold is less than the reconstruction loss. Generally, the autoencoders are trained to minimize reconstruction errors. The graph in Figure 4 shows how reconstruction loss can be calculated using a simplified version of datasets of Electrocardiograms [7]. Conversely, you can apply the same principle used in anomaly detection to find out the reconstruction loss of networks in a cloud environment. It is possible to determine the mean average error for standard examples from the training set. Future examples will be classified as anomalous if their reconstruction error exceeds one standard deviation from the errors observed in the training set.

Why Autoencoders?

Unsupervised learning. They operate in an unsupervised learning paradigm, thus eliminating data labeling and extensive training. This makes it flexible and scalable in anomaly detection, thus allowing the reliable identification of emerging threats on training datasets.

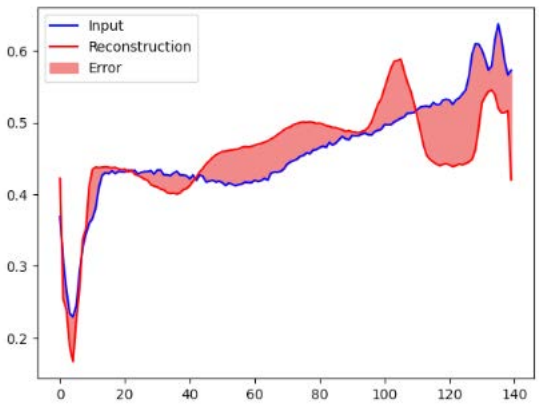

Figure 4: Shows the Anomaly Detection Graph. Source: https://www.tensorflow.org/tutorials/generative/autoencoder

Powerful to noise. They display resistance characteristics to noise; thus, they cannot be affected by noise attenuation. This makes them suitable for the analysis of real-world network traffic[9].

Nonlinearity and complexity. Owing to the nonlinear architecture of autoencoders, which has the capacity to capture important links within input data, it accelerates the notification of minute and unusual things that the cloud server might miss.

Dimensional reduction. Input data is compressed into a lower dimensional latent space; thus, autoencoders facilitate dimensional reduction, thus boosting computational efficiency[9].

Due to the aforementioned benefits, autoencoders remain to be a better option for abnormal network detection. It can also help in screening unusual patterns, protect against cyber-attacks, and strengthen network security, preventing it from cyber-attacks.

Properties of Network Anomaly Intrusion Detection System.

Autoencoders enhance Network Anomaly Intrusion Detection Systems by learning normal behavior, detecting anomalies via high reconstruction error, and reducing dimensionality. The network intrusion detection system (NAIDS) acts as the main purpose of processing network data by monitoring network traffic and identifying patterns in order to classify them as normal or malware. NAIDS screens network traffic in real time and examines packet headers and payloads to expose any malware issues. NAIDS has the ability to inspect network protocols for the identification of abnormal network conditions. NAIDS analyses network traffic in a granular manner in order to identify and classify all the network traffic. Additionally, NAIDS is scalable in nature, thus enabling it to support both medium- and large-sized workloads. The alert and reporting capabilities provide NAIDS to demonstrate its a powerful role. NAIDS can integrate with the Security Information and Event Management (SIEM) systems, thus enhancing its capabilities[13].

Challenges of NAIDS.

There are some challenges that affect the operation of NAIDs, which are outlined below:

- False positives. Generation of false positives by NAIDS is possible, thus issuing out false notifications through the false identification of legitimate network activity as malware. False positives can disturb security teams with unwanted notifications, causing boredom among the security teams. This can make them ignore other genuine security issues that need real attention.
- Encrypted traffic. Encryption protocols such as HTTPS and SSL have been widely adopted to protect data in motion. NAIDS finds it hard to decrypt this traffic in order to inspect the type of traffic that is being conveyed, thus making it difficult[4].
- Insider threats. This is the type of attack that is affecting most organizations. Most NAIDS focus on external attacks without considering that insider threats are the common types of attacks that are scaling out. Detecting insider threats requires an advanced security measure for monitoring network traffic[94].
- Consumption of resources. NAIDS utilizes a lot of computational resources, thus hindering network traffic, which affects network performance.

Network Attacks

Refers to the illegitimate attempt to obtain unauthorized access to a computer network and system. Attacks can be categorized into two groups, namely internal and external. Internal attackers are the inside people, such as employees, who have direct access to the computer systems and networks but lack the privilege to access unauthorized resources. In contrast, external attackers are people outside the organization who want to gain illegitimate access to computer systems and computer networks[5]. By understanding the existing attacks, you can establish a machine-learning technique that addresses every type of attack. Autoencoders, which are neural networks that utilize the unsupervised learning technique, can be used to identify and determine some of these various categories of attacks and mitigate the lateral spread of the attacks.

Table 2: Shows the various categories of attacks with their corresponding definitions.

Category of Attack	Definition
Malware	Malicious software is created to damage or gain unauthorized access to computer systems.
Phishing	Deceptive techniques to trick individuals into passwords or financial data via email or fraudulent websites.
DDoS	Distributed Denial of Service attacks floods networks or servers with more traffic, rendering them unusable to legitimate users.
Man-in-the-Middle	This refers to an attack where a malicious actor gains and modifies communication between two parties without their notice.
SQL Injection	Injection of malicious SQL code into applications' input fields, leading to unauthorized access to databases or data leaks.
Supply Chain Attacks	Attacks that target the supply chain of an organization to compromise its products or services, often through third-party vendors.
Ransomware	Malware that encrypts files or systems and Demands payment for decryption.

Discussion

In this section, we examine the wide implications of the research findings, acknowledge the setbacks of the approach used, and make some proposals for future improvement.

Implications of the Research Findings.

This research paper has greatly contributed to the field

of network security and cloud security through the various techniques that have showcased the feasibility and effectiveness of AI-powered Intrusion detection systems in the cloud infrastructure. The identification of anomalous patterns has elevated the security architecture of the autoencoder[1]. The cloud security for cloud infrastructure has been enhanced. Organizations can run their business processes, procedures, and operations smoothly without worrying about security. The security team can easily identify and detect any cyber threat. The risk management team can effectively conduct risk assessments and mitigate any possible risks. Organizations that adopt the use of machine learning techniques for AI-powered intrusion detection systems are likely to secure all their digital assets.

Limitations

The model has displayed all the benefits of adopting it to support the cloud security environment. However, it has some major drawbacks. The persistence of false positives remains to be a major concern. The false positive that notifies the security team remains a major threat since it alerts the security team of potentially malicious activity, but in reality, it is not. This issue may be fixed by adjusting how sensitive the model can be used to spot unusual activities. Additionally, understanding why the model predicts abnormal results is difficult to validate[12]. Drilling down to the root cause remains challenging. Moreover, the disparity of anomalies in datasets leads to imbalanced data, which compels us to adopt strategies that address the imbalance.

Areas for Improvement

The model can be refined and modified even better by trying out some new techniques. One innovative idea is to utilize the ensemble methods such as stacking or bagging, which are used in machine learning. Bagging entails training various instances of similar learning algorithms of a diversified subset of the data to be trained, whereas stacking combines predictions from various distinct types of models known as base models. This can help in obtaining more anomalies and boost the accuracy of the detective systems. Another aspect that can be done to improve the model is by adjusting the model setting. The learning rate and the size of each layer can be refined to enable the model to work effectively and efficiently[15].

Furthermore, the model can be redesigned smarter by specifying some details pertaining to the cloud environment. For instance, specifying how frequently some parts of the system are used or the type of tasks that should be performed. This improves the intelligence of the model since it will be able to understand what is normal and what is not normal[11]. This makes it sharp in spotting anomalies.

Future Research Directions

Going forward, there are several unexhausted aspects for future and further research in the field of an intrusion detection system that aligns with cloud infrastructure.

Adaptive learning. Future studies could drill down to techniques of adaptive learning. Adaptive learning enables models to automatically adjust to changes in the cloud environment and emerging threats. Learning these adaptive models makes it easy to adapt dynamically to any new challenges[10].

Behavioral profiling. The development of user and resource behavior profiles is another field of interest that can be exploited in future research. Behavioral profiling techniques allow the Intrusion Detection System to comprehend typical behavioral patterns in the cloud environment. Therefore, this can promote anomaly detection by allowing the system to accurately distinguish the normal and abnormal activity in the network[10].

Real-time deployment. Digging deeper into the real-time deployment of intrusion detection systems in a production cloud environment indicates a crucial point of research direction through the deployment of an Intrusion Detection System in real-time. Organizations can identify and detect threats in a timely manner. This is critical in risk management and mitigation[10].

6. Conclusion

Security concerns can result in losses in the cloud computing platform and accelerate users' loss of trust in cloud computing, which will have devastating impacts on the healthy and sustainable evolution of cloud computing. Creating AI-powered intrusion detection systems is one of the solutions for protecting cloud services from malicious attacks. Autoencoders emerge as a promising solution for fortifying security measures within the cloud environment. However, as cyber threats evolve and become increasingly sophisticated, we must prioritize continuous research and adaptation to secure cloud resources effectively. By remaining vigilant and proactive in our efforts to innovate and enhance security measures, we can better protect cloud infrastructure from emerging threats and ensure the integrity and confidentiality of data stored within the cloud environment.

7. References

1. Ashraf J, Bakhshi AD, Moustafa N, et al. Novel deep learning-enabled LSTM autoencoder architecture for discovering anomalous events from intelligent transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, 2020; *22*: 4507-4518.
2. Song Y, Hyun S, Cheong YG. Analysis of autoencoders for network intrusion detection. Sensors, 2021; 21: 4294.
3. Nassif AB, Talib MA, Nasir Q, et al. Machine learning for cloud security: a systematic review. *IEEE Access*, 2021; *9:* 20717-20735.
4. Sarvari S, Sani NFM, Hanapi ZM, et al. An efficient anomaly intrusion detection method with feature selection and evolutionary neural network. *IEEE Access*, 2020; *8*: 70651-70663.
5. Tabrizchi H, Kuchaki Rafsanjani M. A survey on security challenges in cloud computing: issues, threats, and solutions. *The journal of supercomputing*, 2020; *76:* 9493-9532.
6. Tashfeen MTA. Intrusion Detection System Using AI and Machine Learning Algorithm. In *Cyber Security for Next-Generation Computing Technologies* 2024: 120-140. CRC Press.
7. Intro to Autoencoders | TensorFlow Core. (n.d.). TensorFlow.
8. Wang W, Du X, Shan D, et al. Cloud intrusion detection method based on stacked contractive auto-encoder and support vector machine. *IEEE transactions on cloud computing*, 2020; *10:* 1634-1646.
9. Zavrak S, İskefiyeli M. Anomaly-based intrusion detection from network flow features using variational autoencoder. *IEEE Access*, 2020; *8:* 108346-108358.
10. Thakkar A, Lohiya R. A survey on intrusion detection system: feature selection, model, performance measures, application perspective, challenges, and future research directions. *Artificial Intelligence Review*, 2022; *55:* 453-563.
11. Thakkar A, Lohiya, R. A review on machine learning and deep learning perspectives of IDS for IoT: recent updates, security issues, and challenges. *Archives of Computational Methods in Engineering*, 2021; *28:* 3211-3243.

12. Kocher G, Kumar G. Machine learning and deep learning methods for intrusion detection systems: recent developments and challenges. *Soft Computing*, 2021; *25*: 9731-9763.

13. Hajj S, El Sibai R, Bou Abdo J. et al. Anomaly-based intrusion detection systems: The requirements, methods, measurements, and datasets. *Transactions on Emerging Telecommunications Technologies*, 2021; *32*: e4240.

14. Imrana Y, Xiang Y, Ali L, et al. A bidirectional LSTM deep learning approach for intrusion detection. *Expert Systems with Applications*, 2021; *185*: 115524.

15. Johnson Singh K, Maisnam D, Chanu US. Intrusion Detection System with SVM and Ensemble Learning Algorithms. *SN Computer Science*, 2023;*4:* 517.