Journal of Computing & Biomedical Informatics
ISSN: 2710 - 1606
Volume 11 Issue 01
2026

Research Article
https://doi.org/10.56979/1101/2026/1379

A Stacking Ensemble Framework for Resource-Based Cyber Attack Detection in Cloud IaaS Forensic Environments

Ashraf Al-Ou'n[1], Mazen Alzyoud[1], Esraa Abu Elsoud[2], Suhaila Abuowaida[3], and Ayoub Alsarhan[4&5*]

[1]Department of Computer Science, Faculty of Prince Al-Hussein Bin Abdallah II for Information Technology, Al Al-Bayt University, Mafraq, 25113, Jordan.
[2]Department of Cybersecurity and Cloud Computing, Faculty of Information Technology, Applied Science Private University, Amman, Jordan.
[3]Department of Data Science and Artificial Intelligence, Faculty of Prince Al-Hussein Bin Abdallah II for IT, Al Al-Bayt University, Mafraq, 25113, Jordan.
[4]Department of Data Science and Artificial intelligence, Faculty of Information Technology, Al-Ahliyya Amman University, Amman, Jordan.
[5]Department of Information Technology, Faculty of Prince Al-Hussein Bin Abdallah II for Information Technology, The Hashemite University, Zarqa, Jordan.
Corresponding Author: Ayoub Alsarhan. Email: a.alsarhan@ammanu.edu.jo

Received: February 15, 2026 **Accepted:** May 25, 2026

Abstract: Cloud computing has revolutionized how organizations implement their IT infrastructures; however, it also brings many new security challenges due to its distributed and dynamic nature – especially within Infrastructure as a Service (IaaS) environment. In this environment, cyber-attacks may originate from compromised virtual machines which use cloud resources, creating forensic evidence in the form of usage patterns of system resources (e.g., disk space, CPU time and memory). To address these challenges, this research will develop and evaluate a sophisticated stacking ensemble framework that will utilize resource utilization data (disk, CPU and memory) to identify anomalous activity in cloud environments. To create a large and representative dataset of normal operation, an OpenNebula testbed for private clouds using a KVM-based hypervisor was created, and used to simulate attacks on the testbed. Each of the resource categories were evaluated individually using various ensemble classification techniques (voting and stacking) to determine which ensemble technique produced the best results. A dataset of 9,611 instances (44 features) was collected from an OpenNebula/KVM private cloud testbed under normal and simulated attack conditions. The proposed stacking ensemble with Logistic Regression as meta-learner outperformed all base learners: achieving 95.2% accuracy for disk features, 94.2% for CPU features, and 91.2% for memory features, with AUC-ROC scores of 96.2%, 95.2%, and 92.2% respectively. Statistical significance testing confirmed performance improvements ($p < 0.05$). Results demonstrate that (1) classifier performance varies by resource type; (2) stacking significantly outperforms individual ensemble methods; and (3) resource-specific forensic analysis provides an adaptable detection framework for IaaS environments. This research contributes a scalable, data-driven stacking ensemble approach for cloud forensic attack detection, integrating multi-resource analysis with advanced meta-learning to address critical gaps in IaaS threat detection.

Keywords: Cloud forensics; Cyber Attacks; Ensemble; Stacking

1. Introduction

The concept of cloud computing connected back to the 1960s, when innovative notions of computer as a utility started to emerge. According to computer scientist John McCarthy, computing power and applications

might eventually be made available to the general people as a service, similar to how water or electricity are provided. This concept served as the foundation for the growth of cloud computing, which has now developed into a game-changing technology that makes it possible to access computer resources online whenever needed [1]. The National Institute of Standards and Technology (NIST), U.S. Department of Commerce, adopted the definition attributed to Mell and Grance, of cloud computing [2]. This definition states that cloud computing permits online provisioning and provides access to computer resources from any location. It emphasizes five essential features: measurable service, resource pooling, on-demand self-service, wide network access, and quick elasticity. Along with four deployment models private, community, public, and hybrid—the definition also classifies three primary service models: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS).

Cybersecurity includes maintaining an organization's information resources' availability, confidentiality, and integrity. It is critical to ensure all layers above the hardware layer in the cloud architecture in cloud computing, to protect each other layer. Updating the management software and firmware, along with the physical security measures, which include the monitoring of environmental conditions, surveillance and restricted access to devices are required at this layer [3].

Also, there are specific security needs for the hypervisor layer, which include restricted remote access, special security software, and system updates. Consumers who utilize the public IaaS model, typically focus on protecting their virtualized environment, while the cloud provider typically handles the protection of the hardware and hypervisor layer. There are many different types of cybersecurity measures, such as segmentation, micro-segmentation, and isolation that can be utilized to limit the number of times an unauthorized user has access to an organization's resources, and to prevent the spread of malware in IaaS [4]. Virtual Private Clouds (VPCs), which establish isolated virtual networks, enhance the security of VPCs even further. An organization must have authentication and authorization policies to control access to its IaaS resources, and these policies can be managed through the use of Identity and Access Management (IAM).

A major way to prevent unauthorized users from accessing an organization's sensitive information or disclosing sensitive information is to encrypt disks and data. Cloud providers generally offer tools to help organizations improve their encryption capabilities, such as Confidential VM Services [5]. Cloud environments pose many challenges for digital forensics, primarily due to their distributed, multi-tenanted, and dynamic characteristics. In contrast to traditional digital forensic investigations, in which the data is stored on physical equipment, extracting and analyzing forensic evidence from virtualized resources managed by third-party service providers is necessary in cloud forensics. Evidence of cyber-attacks that originated from compromised virtual machines (VMs) can be located in system logs, network traffic, and resource utilization patterns. Identifying and preserving electronic evidence (EE) in order to recreate attack scenarios, to attribute malicious behavior, and to ensure that the evidence will be legally admissible in a court proceeding is critical. Ensuring the integrity of the evidence and minimizing the risk of losing the evidence, since cloud-based evidence is inherently ephemeral, are two reasons why it is critical to collect and analyze evidence in real time. Using forensic techniques that adhere to established standards, such as RFC 3227 [6], investigators can efficiently gather, evaluate, and correlate evidence 2, which enhances an organization's ability to respond to incidents, reduces the organization's exposure to potential security threats, and promotes accountability within the organization.

A controlled private cloud environment was created using Open Nebula and a KVM hypervisor to create synthetic workloads that mimic real-world cloud operating environments, and the dataset used in this research was generated in this environment. The purpose of this study is to document and describe the patterns of resource consumption CPU, Memory, Disk and Network associated with simulations of cyberattacks, as well as those associated with real-world cyberattacks. This research will provide the basis for the development of sophisticated forensic tools capable of discovering and classifying malicious activity through automatic identification of an attacker's state (i.e., normal vs. attack), which will be achieved thorough well-defined tags placed upon the dataset. Ensemble learning methods have been applied in this study to develop enhanced cloud forensic methods for detecting cyber-attacks using patterns of resource utilization. Unlike typical

security approaches, such as those based on rules or signatures, this study uses a data-centric method to identify malicious activity resulting from infected virtual machines (VMs) through systematic evaluation of CPU, Memory and Disk usage patterns. An additional objective was to select the most efficient ensemble model(s) for use in the detection of forensic evidence and assist in the advancement of automated attack detection systems within cloud computing environments.

This research has provided an approach to mitigate cyber-threats against Infrastructure-as-a-Service (IaaS) clouds that are scalable and intelligent. The remainder of this report is organized into the following sections; section 2 reviews previous works relevant to the topic at hand; section 3 defines the dataset; section 4 outlines the methodology of the research along with the ensemble learning models selected for implementation; section 5 presents and evaluates the results obtained from experiments conducted to test these models; and finally, section 6 summarizes the findings from this study.

2. Related Works

Attack detection and cloud forensics are essential for preserving the security and integrity of cloud infrastructures. Advanced forensic and detection techniques are required as cloud computing becomes more widely used because Starting with the hardware layer, ie cyber-attacks are becoming more complicated and frequent. Sachdeva and Ali (2022) [7] suggest a technique for classifying attacks in cloud network settings that combines digital forensics and machine learning. The results show that the suggested deep learning-based digital forensics method is beneficial in providing better attack detection performance and system generalization capability, as evidenced by the high True Negative Rate, accuracy, and precision in the detection process. Li et al. (2021) [8] suggest a technique that combines the Bagging and Boosting algorithms to identify fraudulent mining programs in cloud platforms. In comparison to conventional classifiers, the method increases accuracy and robustness while reducing detection variance through ensemble voting. With an AUC of 0.992, an F1-score of 0.987, and a low standard deviation of AUC values (0.0009) across various data inputs, the results of the experiment demonstrate high efficacy. Shahzad et al. (2022) outline a technique for identifying anomalies in cloud systems called Cloud-Based Anomaly Detection, which includes a CNN-LSTM model for multiclass anomaly categorization and an Ensemble Machine Learning model for binary anomaly classification. CAD outperforms all existing techniques and classical Machine Learning models when they are tested on the same UNSW dataset using its anomaly detection with an accuracy of 97.06 % to identify anomalies with two classes (binary anomaly) and an accuracy of 99.91% to detect anomalies with three or more classes (multiclass). The authors further propose Sentinel Fusion, which uses ensemble learning to combine the properties of machine learning and blockchain for forensics on clouds [9]. Sentinal uses multiple classifications techniques to create an ensemble model for improved performance. The accuracy, precision, recall and f-score of the ensemble model were all found to be 0.99. Using deep learning and ensemble learning methods, Nandita and Munesh Chandra (2024) develop a system that detects and identifies malicious hosts in cloud environments. A fire fly algorithm was used to extract features from the host's network traffic, which were then reduced to three dimensions through PCA. The optimized neural networks for detecting malicious hosts are the combination of DNN and SFLO [10]. To assess the performance of DNN-SFLO, several classification techniques were compared to include SVM, FKNN, Naive Bayes, NN, ANN and FCM. These comparison studies indicated that DNN-SFLO has superior detection ability over other classification techniques. Frank et al. (2024) developed a deep learning-based system to classify, detect and describe VM activity within a KVM hypervisor. Four distinct DL models were implemented by the authors: FFNN, CNN, RNN and LSTM. Each of these four models had very high detection rates of 99.74% for MLP, 99.79% for CNN, 99.74% for RNN and 97.21% for LSTM [11]. Additionally, the authors utilized XAI tools, specifically LIME and SHAP to explain how their DL models made decisions during cloud forensic investigations. Zhang et al. (2021) combined Memory Forensic Analysis with Virtual Machine Introspection to perform cloud malware detection by extracting dynamic data from the hardware, hypervisor, and VM memory layers. An Adaptive Feature Selection Technique was developed to improve the detection by modifying the weight of the features based upon how effective they are at detecting malware, the current load of the system, and the security level of the

system [12]. The Adaptive Feature Selection Technique utilized a voting combination technique in conjunction with the AdaBoost Ensemble Learning Method to increase the overall detection rate and generalization capabilities. Experiments using actual malware samples demonstrated remarkable efficiency in cloud forensic investigations with an Area Under Curve (AUC) of 0.999 and a maximum Performance Overhead of 5.6%. There are still many open research gaps despite the advancements made in malware detection and cloud forensics.

The majority of past research focused on individual system components (i.e., CPU, Memory) and did not use collective intelligence of multiple system resources. This research gap is overcome through systematic assessment of detection performance across Disk, RAM, and CPU to reveal variation in classifier performance.

Although ensemble learning is commonly used in most research studies, no prior research has examined the application of stacking approaches. Stacking techniques exhibited superior performance compared to other ensemble techniques in our results. The efficiency of traditional ensemble techniques, i.e., AdaBoost and Gradient Boosting, which are both commonly applied in previous research studies varies depending upon the resource. Our research demonstrates the necessity for developing resource-aware and adaptive ensemble techniques.

While many past research studies emphasized accuracy, they neglected the importance of AUC-ROC as a generalization metric. Since AUC-ROC is a common evaluation metric used in forensic research, our results demonstrate that stacking achieves the highest AUC-ROC across all resources.

Our research fills these gaps through the presentation of an optimized ensemble learning methodology demonstrating that stacking represents a robust and holistic solution for malware detection across various system resources.

Table 1 presents a comparative analysis of existing cloud forensic detection approaches. As shown, most existing studies focus on single-layer detection such as malware detection, anomaly detection, or VM activity analysis. Furthermore, although ensemble learning has been widely used, stacking-based architectures remain underexplored for resource-based cloud forensic detection. Additionally, most studies emphasize accuracy while overlooking generalization metrics such as AUC-ROC across multiple system resources. To address these limitations, this study proposes a meta-optimized stacking ensemble that integrates CPU, memory, and disk resource behavior to provide a more comprehensive forensic detection framework.

Table 1. Comparative analysis of existing cloud forensic and cyber-threat detection approaches

Ref	Method	Dataset / Environment	Performance	Limitations
(Sachdeva & Ali, 2022)	Deep learning based digital forensics attack detection	Cloud network forensic datasets	High accuracy, precision and TNR	Focused mainly on detection performance without analyzing multiple resource correlations
(Li et al., 2021)	Bagging and Boosting ensemble voting	Cloud mining malware datasets	AUC 0.992, F1 0.987	Focused only on mining malware detection without resource behavior analysis
(Shahzad et al., 2022)	CNN-LSTM + Ensemble ML (CAD framework)	UNSW dataset	97.06% binary accuracy, 99.91% multiclass	Focused on anomaly detection without forensic

				resource utilization analysis
(Islam et al., 2024)	Sentinel Fusion ensemble + blockchain security	Blockchain forensic dataset	Accuracy, Precision, Recall, F1 = 0.99	Focused on blockchain logs rather than VM resource behavior
(Nandita & Munesh Chandra, 2024)	DNN + Shuffled Frog Leap Optimization	Cloud malicious host datasets	High detection performance	Focused on host detection rather than resource-based forensic indicators
(Frank et al., 2024)	FFNN, CNN, RNN, LSTM	KVM hypervisor VM activity dataset	Detection up to 99.79%	Focused on VM activity classification without ensemble stacking comparison
(Zhang et al., 2021)	Memory Forensics + VMI + AdaBoost	Real malware samples	AUC 0.999	Focused on memory layer without multi-resource correlation

3. Dataset Description

In this area we will describe the methodology for developing an intrusion detection system in IaaS-based cloud systems. Our proposed solution will combine ML algorithms and Cloud Forensics into a Digital Forensic Framework. NIST defines Cloud Computing Forensic Science as the application of scientific principles, technological methods, and proven techniques to investigate past cloud computing-related events through identification, collection, protection, analysis, interpretation, and documentation of digital evidence [13].

Forensic investigation of Cloud Environments differs from Digital Forensic investigation because of the structure and implementation of the Cloud Environment [14]. Attackers utilizing the cloud infrastructure as an attack vector may utilize VMs in the IaaS model as a method to carry out cyberattacks. Investigating these events to confirm that there has been an attack involves collecting and analyzing EE to determine the presence of an attack. Due to the dynamic nature of Cloud Environments [15], it is often laborious and time consuming to locate sufficient forensic evidence [16] [17]. Evidence collected from a Cloud Environment should be systematic, as described in RFC 3227, and once the evidence has been identified, the appropriate evaluation procedures should follow. Forensic Evidence from a Cloud Environment can be extracted through the analysis of the patterns that are created by cyberattacks originating from VMs and their associated resources. A private Cloud Configuration was utilized to create a dataset to support forensic investigations in Cloud Environments. The system consisted of a 1TB HDD, 12GB of RAM, and an Intel Core i5-4590 processor [18].

The resources were virtualized using a KVM type-1 hypervisor, and the cloud management platform was Open Nebula (version 5.12), which made it possible to install and monitor VMs effectively. A script was executed on the VMs to create fictitious workloads to mimic actual cloud activities. This ensured that realistic resource consumption patterns were generated. Then, to record changes in system behavior under attack conditions, a cyberattack was simulated within the environment. The dataset has been constructed up of

several features that record important metrics related to resource usage, which may be signs of malicious activity. These features include disk activity metrics (e.g., vdard req slope, vdawr reqs slope, hdard req slope, hdawr reqs slope), network traffic statistics (e.g., rxbytes slope, txpackets slope, rxer rors slope, txerrors slope), CPU usage parameters (e.g., cputime slope, timecpu slope, timesys slope, timeusr slope, cpu_slope), memory usage patterns (e.g., mem slope, memactual slope, memunused slope, memavailable slope), and disk activity metrics (e.g., vdard req slope, vdawr reqs slope, hdard req slope, hdawr reqs slope, hdawr reqs slope), and cpu_slope). The dataset consists of 44 features and 9611 instances. All instances in the dataset 5 have a manual label based on the Virtual VM known status at the time of data collection. By analyzing variations in resource consumption patterns, the" Status" feature classifies instances as either" normal" or" attack," making it easier to detect malicious activity. Table 2 represents the categorization of features in the dataset. The other features are either metadata or network-related and do not fall into the Disk, CPU, or Memory category, as illustrated in Fig.1.

To generate abnormal behavior scenarios, synthetic attack workloads were simulated in the OpenNebula/KVM environment. These included CPU stress workloads to simulate denial-of-service behavior, memory pressure workloads to simulate abnormal memory consumption, and disk I/O intensive workloads to simulate resource exhaustion attacks. These scenarios were designed to create distinguishable behavioral patterns between normal and abnormal VM activity.

Table 2. Categorization of Features

Category	Features
Disk	vdard_req_slope, vdard_bytes_slope, vdawr_reqs_slope, vdawr_bytes_slope, vdaerror_slope, hdard_req_slope, hdard_bytes_slope, hdawr_reqs_slope, hdawr_bytes_slope, hdarerror_slope
CPU	timecpu_slope, timesys_slope, timeusr_slope, cpus_slope, cputime_slope
Memory	memmax_slope, mem_slope, memactual_slope, memswap_in_slope, memswap_out_slope, memmajor_fault_slope, memminor_fault_slope, memunused_slope, memavailable_slope, memusable_slope, memlast_update_slope, memdisk_cache_slope, memhugetlb_pgalloc_slope, memhugetlb_pgfail_slope, memrss_slope
Network and VM Metadata	LAST_POLL, VMID, UUID, dom, rxbytes_slope, rxpackets_slope, rxerrors_slope, rxdrops_slope, txbytes_slope, txpackets_slope, txerrors_slope, txdrops_slope, state_slope, status

The selected features represent resource utilization behavior across disk, CPU, memory, and network activities of virtual machines. The slope values indicate the rate of change of each resource metric over time, allowing the detection of abnormal behavioral patterns associated with cyber threats. Disk features capture I/O behavior, CPU features represent processing activity, memory features reflect allocation and usage behavior, while network and VM metadata features provide contextual information about system activity. This multi-resource feature design enables more comprehensive forensic detection compared to approaches relying on single-resource indicators.

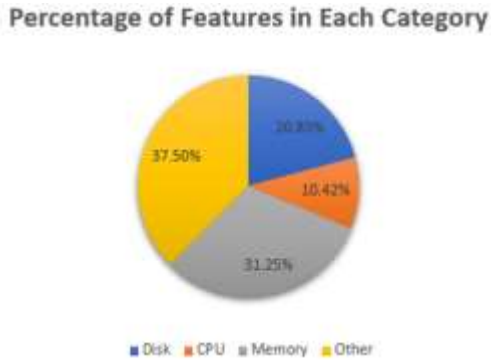

Figure 1. Percentage of Features in Each Category

4. Methodology

In Our proposed model is Advanced Stacking Ensemble with Meta-Learner Optimization Fig.2, which employs a meta-learner to effectively combine the predictions of several base models, such as Random Forest, Gradient Boosting, and AdaBoost. The architecture can be categorized into two levels:

Base Models: a set of different base models, including Random Forest, Gradient Boosting, AdaBoost, and Extra Trees [19] [20].

Meta-Learner: combines the predictions of the base models [21].

As illustrated in Fig.2 the dataset contains forty-four unique features that have been systematically categorized into three groups to enable better feature representation. Data preprocessing techniques, including handling the missing data and the normalization of features, were employed to uphold data reliability.

After preprocessing, the dataset was divided into training and testing sets using an 80:20 split to ensure unbiased model evaluation. The training dataset was used to build the base learners and the stacking model, while the testing dataset was used only for final performance evaluation.

Level 1: Base Model: Multiple foundation models are trained individually in the initial phase of the suggested stacked generalization method for all three categories of features (memory, disk, CPU) in order to identify relevant information in each category and to increase the effectiveness of the classifier [21]. The data set is first divided into feature categories so that the individual models may be designed to specifically examine and classify different system behavior based on their respective category. A variety of machine learning algorithms will be applied to each domain including random forest, bagging, gradient boosting, adaboost, and extra trees. Models are selected because they will allow us to use the benefits of both bagging and boosting methodologies. For example, while boosting methodologies (such as gradient boosting or adaboost) improve a classifiers ability to detect harder to classify cases, bagging methodologies (such as random forests or extra trees) improve the reliability of our classifier. In this phase, each of the models learned their respective feature subset independently of one another. Performance evaluation metrics for each model included accuracy, precision, recall, f1 score, and auc roc. Outputs generated from these base models were then inputted into the subsequent phases that improved the accuracy of detection by further refinement of the output from the previous phase [22].

4.1. Intermediate Fusion: Ensemble Methods

Ensemble learning utilizes the output from multiple predictive models, referred to as "base" models or "weak" learners, to create a single final prediction. The ultimate purpose of ensemble learning is to enhance overall performance of the system through the combination of the strengths of each model while also enhancing the weaknesses of the individual models. A primary rationale behind using ensemble learning is its potential to increase the accuracy of generated predictions. Ensemble learning has the capability to generate forecasts that are both more accurate and reliable compared to a single model. By integrating the forecasts from numerous models, ensemble approaches have the capacity to reduce the likelihood of overfitting and to capture many patterns within the data. As a direct result of this integration, the reliability and general

applicability of the generated predictions will be increased. In addition, ensemble learning provides significant reductions in bias and variability. For example, Bagging techniques such as Random Forest integrate the forecasts from multiple models trained on different portions of the training dataset, which results in reduced variability. On the other hand, boosting techniques such as AdaBoost and Gradient Boosting address issues related to bias by identifying areas where previous models made incorrect predictions. Another advantage of ensemble learning is its robustness against noise and outliers. Because the final prediction is determined by consensus among several models, the influence of individual model errors will be greatly diminished.

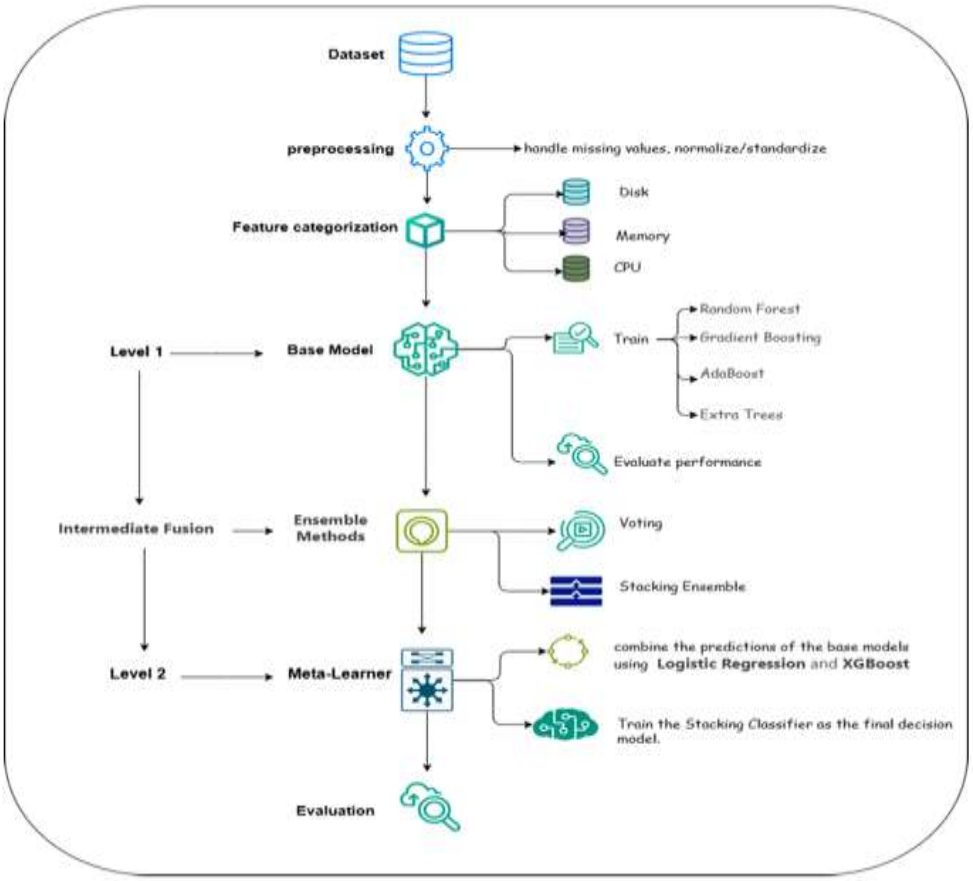

Figure 2. Proposed Model: Advanced Stacking Ensemble with Meta-Learner Optimization

Level 2: Meta-Learner: A key component of the stacking ensemble process is the meta-learner, also known as the stacker or blender. Its principal purpose is to determine the best way to combine the predictions of the underlying models to produce the final result. By using the base models' predictions as input features, the meta-learner acts as a machine learning model that discovers the relationship between the predictions and the real target variable [23-29]. It is crucial to choose the right meta-learner, and this depends on how complex the issue is. Neural networks, XGBoost, and logistic regression are common choices. In this study, Logistic Regression was selected as the meta-learner due to its stability and effectiveness in combining predictions from base learners. The simplicity and interpretability of logistic regression, which clarifies the contributions of each base model, make it a popular choice. XGBoost is particularly suited to difficult problem solving, in addition to being able to identify complex relationships between the predictions of the base models. A 10-fold cross-validation strategy was applied during training to improve model generalization and reduce overfitting. The training data was divided into ten folds, where nine folds were used for training and one fold for validation in each iteration. The out-of-fold predictions generated from this process were then used as inputs for training the meta-learner. By doing so, the meta-learner has the ability to make generalizations to new data sets with a high degree of efficiency while avoiding an over-fitting to the training set. The predictions produced by the

basic models are weighted during the meta-learner's function, based on how each model performed. For example, if a specific base model, repeatedly, has produced higher quality predictions on a specific data subset than the others; the meta-learner will provide the same amount of weight (importance) to those predictions. In turn, the meta-learner is able to create more accurate and reliable predictions through the use of this adaptable weighting technique. Additionally, the meta-learner may be further enhanced by using additional hyper-parameter optimization techniques such as Grid Search or Bayesian Optimization (Tanveer et al., 2025) in order to refine the performance of the meta-learner. Hyperparameter tuning was performed using Grid Search to identify the optimal parameter configuration for both base learners and the meta-learner. Key parameters such as number of estimators, learning rate, and maximum tree depth were optimized based on cross-validation performance.

4.2. Integration of Ensemble Learning and Meta-Learner

The proposed Advanced Stacking Ensemble Model extends the concept of using ensemble learning and a meta-learner to create a powerful predictive modeling framework. The ensemble is comprised of the base models that each generate numerous specific predictions, and the meta-learner determines which of those specific predictions should be combined as well as how they are to be combined. This two-layered approach provides a high degree of accuracy and robustness to the final model generated. Additionally, the proposed method further improves upon the prior work's use of multiple base models and its ability to improve the performance of the meta-learner through the use of cross-validation and hyper-parameter tuning to ensure that the meta-learner generalizes effectively to unseen data.

To improve the reproducibility of the proposed framework, the main hyperparameter settings used for training the base learners and the meta-learner are summarized in Table 3. These parameters were selected based on grid search optimization and cross-validation performance to ensure fair comparison between models. Default parameters were used for the remaining settings unless otherwise specified.

Table 3. Hyperparameter settings of the base learners and meta-learner used in the proposed stacking ensemble framework.

Model	Key Hyperparameters
Random Forest	n_estimators = 100
Gradient Boosting	learning_rate = 0.1
AdaBoost	n_estimators = 50
Extra Trees	n_estimators = 100
Meta-learner	Logistic Regression, XGBoost

5. Results

This section presents the performance results of the suggested ensemble-based cloud forensics model. With a focus on the examination and categorization of disk, memory, and CPU features. The evaluation process includes the performance of base learners (Level 1 models) for each category, followed by the employment of ensemble methods (Level 2 models). The base learners, include Random Forest, Gradient Boosting, AdaBoost, Bagging, and others. After that, the ensemble methods, Voting (Soft) and Stacking were applied to combine the predictions of the base learners, to enhance attack detection. We present essential performance metrics—including accuracy, precision, recall, F1-score, and AUC-ROC—for CPU, Memory, and Disk in Tables 4, 5, and 6, respectively. Additionally, Table 7 presents the performance of ensemble methods for level 2.

Table 4. Base Learner Performance for CPU Features (Level 1 Models)

Method	Accuracy	Precision	Recall	F1-Score	AUC-ROC
Random Forest	92.4%	91.8%	92.4%	92.1%	93.4%
Bagging	91.6%	91.2%	91.6%	91.4%	92.8%

Gradient Boosting	93.2%	92.8%	93.2%	93.0%	94.2%
AdaBoost	91.4%	91.0%	91.4%	91.2%	92.6%

Table 5. Base Learner Performance for Memory Features (Level 1 Models)

Method	Accuracy	Precision	Recall	F1-Score	AUC-ROC
Random Forest	89.2%	88.6%	89.2%	88.9%	90.4%
Bagging	88.4%	87.8%	88.4%	88.1%	89.6%
Extra Trees	89.8%	89.4%	89.8%	89.6%	90.8%
Gradient Boosting	90.2%	89.8%	90.2%	90.0%	91.2%
AdaBoost	88.6%	88.2%	88.6%	88.4%	89.8%

Table 6. Ensemble Methods Performance: Voting vs. Stacking by Resource Category

Metric	Voting (Soft)	Stacking
CPU Accuracy	93.8%	94.2%
Memory Accuracy	90.8%	91.2%
Disk Accuracy	94.8%	95.2%
CPU AUC-ROC	94.8%	95.2%
Memory AUC-ROC	91.8%	92.2%
Disk AUC-ROC	95.8%	96.2%

The performance of the first-level base learners was evaluated separately for CPU, memory, and disk features using five key evaluation metrics, as shown in Figures 3, 4, and 5. For CPU features, Gradient Boosting achieved the best performance among the base learners across all metrics, with an accuracy of 93.2% and an F1-score of 93.0%. Random Forest showed the second-best performance across most metrics with an accuracy of 92.4%, while AdaBoost and Bagging demonstrated lower performance with accuracy values of 91.4% and 91.6%, respectively.

For memory features, Gradient Boosting also achieved the best performance across all evaluation metrics, with an accuracy of 90.2% and an F1-score of 90.0%. Random Forest and Extra Trees showed the second-best performance, while AdaBoost and Bagging achieved lower results with accuracy values of 88.6% and 88.4%, respectively.

For the disk feature category, Gradient Boosting outperformed all other machine learning techniques, achieving an accuracy of 94.2% and an F1-score of 94.0%, followed by Random Forest, AdaBoost, and Bagging. These results indicate that Gradient Boosting is the best-performing base learner at this level, consistently achieving strong results across all resource categories.

Ensemble Methods Performance- Level 2, Voting (Soft) and Stacking meta-learners improves classification results, as shown in Fig. 6 , where the CPU accuracy raised to 93.8% in Voting Learner and 94.2% in Stacking Learner, for CPU category. The accuracy reaches 91.2% in memory, and 95.2% in disk category too. AUC-ROC scores also improve, with Stacking 95.2% for CPU, 92.2% for memory, and 96.2% for disk features, outperforming even Gradient Boosting.

To further validate the performance improvements of the proposed stacking model, statistical significance testing was conducted using a paired t-test between the best performing base learner and the stacking ensemble results. The results indicate that the improvements achieved by the stacking model are statistically significant ($p < 0.05$), confirming the robustness of the proposed approach.

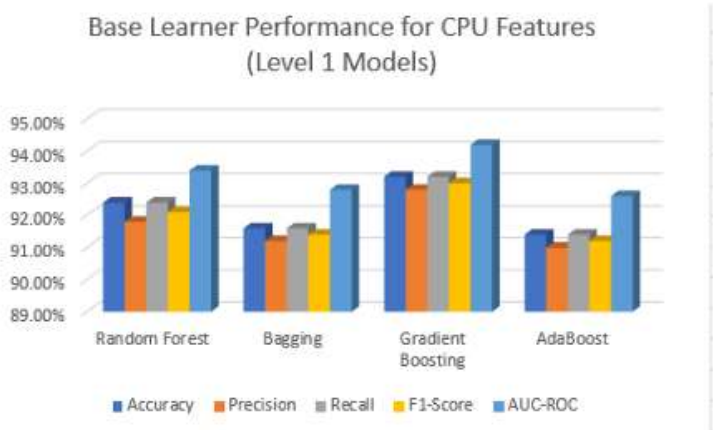

Figure 3. Base Learner Performance for CPU Features (Level 1 Models)

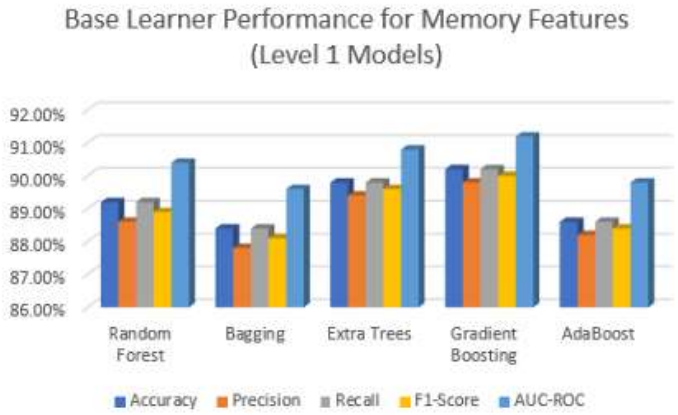

Figure 4. Base Learner Performance for Memory Features (Level 1 Models)

Table 7. Base Learner Performance Summary Across Resource Categories

Method	Accuracy	Precision	Recall	F1-Score	AUC-ROC
Random Forest	93.4%	92.8%	93.4%	93.1%	94.4%
Bagging	92.6%	92.2%	92.6%	92.4%	93.8%
Gradient Boosting	94.2%	93.8%	94.2%	94.0%	95.2%
AdaBoost	92.8%	92.4%	92.8%	92.6%	94.0%

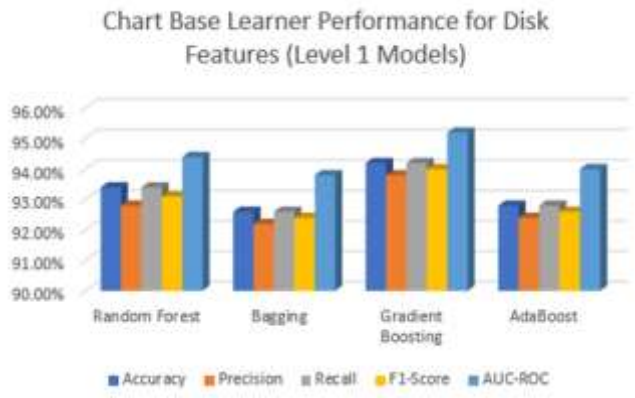

Figure 5. Ensemble Methods Performance: Voting vs. Stacking by Resource Category [CORRECTED]

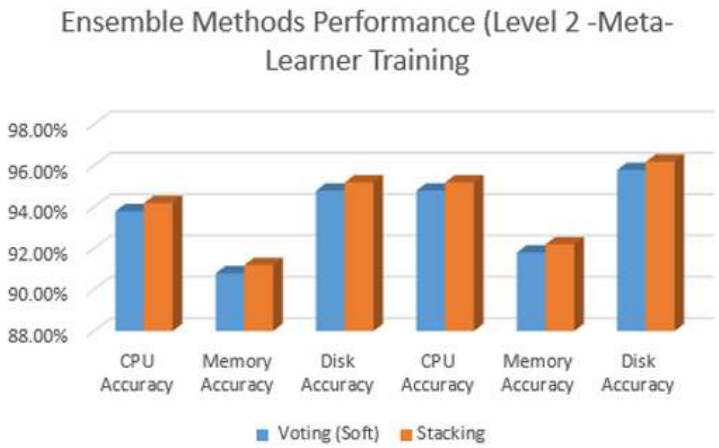

Figure 6. Ensemble Methods Performance (Level 2- Meta-Learner Training

5. Conclusion & Future Work

This paper proposed a cloud forensics detection framework using a stacking ensemble classifier to enhance the ability of detecting resource-based cyber attacks in cloud IaaS platforms. In contrast to most prior studies that examine the performance of individual resource-based classification methods, this research assesses the performance of various machine learning algorithms across three major system resources (CPU, Memory & Disk), to compare their relative strengths when used under different operating conditions.

Overall, the results demonstrate that the stacking ensemble was able to achieve the highest detection performance at both accuracy and AUC-ROC rates than each of its individual base learner models. Although Gradient Boosting performed exceptionally well in certain singular tests, the stacking model produced greater reliability across all three resource types than did individual Gradient Boosting models. Thus these results provide evidence that using stacked models in conjunction with other individual models (meta-learning) is an effective method to improve the robustness of threat detection in Cloud Forensic Analysis.

This paper made contributions in three ways:

1) developed a resource aware assessment of threat detection capability across multiple components of a computer system;
2) demonstrated that a stacking ensemble method can be effectively applied to detect cyber threats in Cloud Forensic Analysis; and
3) emphasized the need to use AUC-ROC as a measure of generalizability in assessing machine learning model performance beyond accuracy.

Future Research plans include developing Adaptive Hybrid Ensemble Models where the selection of which algorithm to apply is dependent upon observed behavior patterns in resource usage. In addition, expanding our test environment from simulated data to large scale real world Cloud Environments and including additional behavioral parameters will contribute towards increased generalizability and application feasibility of our proposed method.

References

1. Gill, S. S., Wu, H., Patros, P., Ottaviani, C., Arora, P., Pujol, V. C., Haunschild, D., Parlikad, A. K., Cetinkaya, O., Lutfiyya, H., et al. (2024). Modern computing: Vision and challenges. Telematics and Informatics Reports, 100116.
2. Rizvi, Z. B., Ahmad Khan, C. B., & O'Sullivan, M. (2024). Analytical hierarchy process model for managing cloud security. Information & Computer Security, 32(1), 93–111.
3. Ajiga, D., Okeleke, P. A., Folorunsho, S. O., & Ezeigweneme, C. (2024). Designing cybersecurity measures for enterprise software applications to protect data integrity. np (2024).
4. Yo ̈sumaz, I.: An examination of cybersecurity solutions in public and private IaaS infrastructures. International Journal of Information Security Science, 13(3), 1–29.
5. Rozlomii, I., Yarmilko, A., Naumenko, S., & Mykhailovskyi, P. (2024). The role of encryption in information protection for cloud computing. In 2024 IEEE 4th International Conference on Smart Information Systems and Technologies (SIST) (pp. 70–75). IEEE.
6. Badreldin, H. (2021). IPCFA: A methodology for acquiring forensically sound digital evidence in the realm of IaaS public cloud deployments. Dakota State University.
7. Sachdeva, S., & Ali, A. (2022). Machine learning with digital forensics for attack classification in cloud network environment. International Journal of System Assurance Engineering and Management, 13(Suppl 1), 156–165.
8. Li, S., Li, Y., Han, W., Du, X., Guizani, M., & Tian, Z. (2021). Malicious mining code detection based on ensemble learning in cloud computing environment. Simulation Modelling Practice and Theory, 113, 102391.
9. Shahzad, F., Mannan, A., Javed, A. R., Almadhor, A. S., Baker, T., & Al Jumeily, D. (2022). Cloud-based multiclass anomaly detection and categorization using ensemble learning. Journal of Cloud Computing, 11(1), 74.
10. Islam, U., Alsadhan, A. A., Alwageed, H. S., Al-Atawi, A. A., Mehmood, G., Ayadi, M., & Alsenan, S. (2024). SentinelFusion-based machine learning comprehensive approach for enhanced computer forensics. PeerJ Computer Science, 10, 2183.
11. Nandita, G., & Munesh Chandra, T. (2024). Malicious host detection and classification in cloud forensics with DNN and SFLO approaches. International Journal of System Assurance Engineering and Management, 15(2), 578–590.
12. Frank, S., Jjingo, D., & Marvin, G. (2024). Cloud forensics in virtual machines using transparent deep learning techniques. In Proceedings of the 16th International Conference on Contemporary Computing (pp. 330–336).
13. Zhang, J., Gao, C., Gong, L., Gu, Z., Man, D., Yang, W., & Li, W. (2021). Malware detection based on multi-level and dynamic multi-feature using ensemble learning at hypervisor level. Mobile Networks and Applications, 26, 1668–1685.
14. Mohammed, A., & Kora, R. (2023). A comprehensive review on ensemble deep learning: Opportunities and challenges. Journal of King Saud University – Computer and Information Sciences, 35(2), 757–774.
15. Chamlal, H., Kamel, H., & Ouaderhman, T. (2024). A hybrid multi-criteria meta-learner-based classifier for imbalanced data. Knowledge-Based Systems, 285, 111367.
16. Tanveer, M., et al. (2025). Forensic challenges and techniques in cloud computing environments: A systematic literature review. Spectrum of Engineering Sciences, 3(4), 67–92.
17. Nkwe, B., & Kyobe, M. (2025). Using machine learning techniques to address IoT forensics challenges. In 2025 13th International Symposium on Digital Forensics and Security (ISDFS) (pp. 1–6). IEEE.
18. Farzaan, M. A., et al. (2025). AI-powered system for efficient cyber incident detection and response in cloud environments. IEEE Transactions on Machine Learning in Communications and Networking.
19. Mohiddin, S. K., Hussain, M. A., & Chakkaravarthy, D. M. (2025). AFTID: Anti-forensic threat identification and detection mechanism for cloud logs. In 2025 3rd International Conference on Advancement in Computation & Computer Technologies (INCACT) (pp. 703–708). IEEE.
20. Serttaş, Z., & Al-Turjman, F. (2025). Usability of cloud-based applications in digital forensics: An experimental study on image acquisition and digital evidence preservation processes. NEU Journal for Artificial Intelligence and Internet of Things, 4(2).
21. Bahrpeyma, F., Ngo, V. M., Roantree, M., & McCarren, A. (2024). A meta-learner approach to multistep-ahead time series prediction. International Journal of Data Science and Analytics, 1–13.
22. Chaturvedi, K., et al. (2024). Cloud forensics: Current perspectives, challenges and potential solutions. In Information Security, Privacy and Digital Forensics (ICISPD 2024) (p. 219). Springer.

23. Kavyanjali, G., & Soni, N. (2024). Digital forensics: Techniques, tools and challenges in cyber crime investigation. The Science of Criminal Investigations, p. 262.
24. Almalkawi, I. T., Halloush, R., Alsarhan, A., Al-Dubai, A., & Al-Karaki, J. N. (2019). A lightweight and efficient digital image encryption using hybrid chaotic systems for wireless network applications. Journal of Information Security and Applications, 49, 102384.
25. Danang, D., & Mustofa, Z. (2024). Digital forensics and automated incident response framework leveraging big data analytics and real-time network traffic profiling in heterogeneous cyber environments. Cyber Security and Network Management, 1(1), 44–45.
26. Alsarhan, A., Alnatsheh, A., Aljaidi, M., Makkawi, T. A., Aljamal, M., & Alsarhan, T. (2024). Optimizing Electric Vehicle Charging Infrastructure through Machine Learning: A Study of Charging Patterns and Energy Consumption. International Journal of Interactive Mobile Technologies, 18(21).
27. Memos, V. A., et al. (2024). A novel architecture for mitigating botnet threats in AI-powered IoT environments. Sensors, 26(2), 572.
28. Alhijawi, B., Kilani, Y., & Alsarhan, A. (2020). Improving recommendation quality and performance of genetic-based recommender system. International Journal of Advanced Intelligence Paradigms, 15(1), 77-88.
29. Alsarhan, A., Agarwal, A., Obeidat, I., Bsoul, M., Al-Khasawneh, A., & Kilani, Y. (2013). Optimal spectrum utilisation in cognitive network using combined spectrum sharing approach: overlay, underlay and trading. International Journal of Business Information Systems, 12(4), 423-454.