

Deep Learning Approaches for Threat Detection in Multi-Cloud Environments

Rohit Gorle

Senior Software Engineer

rohit.gorlee@gmail.com

Abstract—Multi-cloud computing has gained prominence as a popular IT architecture in recent years. Organizations use different cloud services provided by multiple companies to optimize operations. However, using multiple suppliers as a service increases security risks, which are opening a new surface area for attackers and making multi-cloud environments an attractive target. Malware detection capability is critical for multi-cloud infrastructure. Therefore, the research community is intuitively driven to use deep learning models for identifying the threat. Despite years of development and practical application, another systematic review of the detection of malware threats in multi-cloud environments is missing.

This work conducts such a review. Detection techniques based on deep learning methods designed for multi-cloud environments are analyzed in detail. Qualitative and quantitative analyses are presented in a structured overview. The research body covering malware detection supports future deep learning-based projects and serves as a concise foundation for widespread automated, state-of-the-art threat detection in multi-cloud infrastructures.

Keywords—Multi-Cloud Computing, Malware Detection, Deep Learning, Cloud Security, Threat Detection, Cybersecurity, Automated Detection, Security Analytics, Multi-Cloud Infrastructure, Malware Analysis.

I. INTRODUCTION

Multi-cloud environments allow organizations to retain their stability while achieving flexible deployment and limited vendor lock-in. Multi-cloud facilities cluster two or more cloud assets, built upon a cloud deployments and orchestrating resources offered by different cloud service providers. The components within a multi-cloud can reside on the same place or in different regions, and interact through the public Internet or dedicated private networks. However, these partnerships bring multiple challenges. Besides the advantages of the distributed architecture, which robs such environment of its characteristic single point of failure, threat detection can be lead throughout the multi-cloud environment at any domain with enough knowledge, resources, or time.

The threats category and their attack vectors are not very different than those present in a single cloud environment. Nevertheless, in a Multi Cloud Environment, threats can originate in any of the domains, leading to define a set of goals that they should accomplish. Threat Detection can be seen as a process composed of four high-level activities: Data Acquisition, Data Monitoring, Data Analysis and Incident Reporting, leading to the definition of a Risk Assessment Framework, which should assess Data Gravity, Data Latency and Data Sovereignty in Multi-clouds Inter-cloud. Organizations achieve the required synergies by deploying crucial workloads in the appropriate cloud environment.

II. BACKGROUND

A well-established taxonomy and actor-based framework describe a diverse threat landscape composed of threat actors with contrasting objectives and negotiation power. Recent cloud-related security incidents and evolving attacks exposures reveal common weaknesses vulnerable to threats affecting both privacy and the integrity of the overall data and service. While detection secrets run in separate clouds, in data-sovereign clouds with sensitive information, cross-cloud attacks may escape detection because the signals associated with them run through otherwise oblivious clouds. New methods for detection and prevention that cover the full attack lifecycle and span multiple clouds without jeopardising the detection of threats specific to each cloud are – and remain – an active goal of research activity.

The increasing complexity of the aforementioned landscape has favoured a growing division of labour among both the perpetrators and the defenders of cyber threats. Attackers focus on conceiving and exploiting new vulnerabilities, whereas defenders invest their capabilities in keeping systems properly configured and patched. Threat detection and prevention systems usually operate at large scale by continuously applying known rules of engagement, indicated by event correlation or analytical algorithms, to data streams collected from across the dynamic full attack lifecycle. Nevertheless, the dynamic nature of these systems exposes them to the risk of missing significantly altered exploits that do not match any of the alerting criteria in force. Advanced persistent threats may therefore successfully manoeuvre and perform their malicious activities unnoticed over an extended period.

A. Experimental Setup

The evaluation set contains $N = 100,000$ events with attack prevalence $\pi = 0.20$, in the spirit of the CICIDS2017-style traffic and the CloudSim-generated activity logs and telemetry (aligned to the Azure Monitor schema) described in the source review. Table 1 summarises the four architectures and Table 2 lists the dataset and deployment parameters used in the model.

In an Azure-based data-engineering deployment the four stages map naturally onto platform services: acquisition through Azure Monitor and Log Analytics exports (with AWS CloudTrail pulled through provider APIs) into a schema-less data lake, pre-processing in Spark or Data Factory flows, inference on GPU-backed Azure Machine Learning endpoints, and incident reporting through a SIEM such as Microsoft Sentinel. This mapping is an interpretation used to give the model parameters a concrete meaning; it is not described in the source review.

Table 1: Architecture Summary of the Evaluated Detection Models

Attribute	Model A	Model B	Model C	Model D
Architecture	CNN	LSTM	Transformer	CNN–LSTM
Source use case	Intrusion detection	Intrusion detection	Anomaly detection	Event classification
Parameters p _m (M)	2.4	3.1	11.8	5.2
Compute F _m (MFLOPs/event)	4.8	9.6	31.0	14.2
Recurrent steps T _m	0	32	0	16
Activation a _m (MB/event)	0.8	1.4	3.9	2.2
Pre-processing L _{pre} (ms)	8	10	12	11
CPU time t _{cpu} (μs/event)	150	190	260	230
Convergence τ _m (epochs)	8	14	22	12
Separability d'	3.6	3.8	4.1	4.3
Operating FPR	0.020	0.020	0.014	0.012

Model D is the hybrid CNN–LSTM; the other columns are single-architecture baselines. Parameter values are modelling assumptions.

Table 2: Dataset and Deployment Parameters

Parameter	Symbol	Value
Test events	N	100,000
Attack prevalence	π	0.20
Window size	W	256 events
Clouds (Azure, AWS, private)	K	3
Arrival rate	λ	12,000 events/s
Inter-cloud RTT	RTT	28 ms
In-cloud access	L _{loc}	3 ms
Fan-out penalty	δ	0.15
Window payload / bandwidth	S / B	64 KB / 1 Gbps
Alert-correlation factor	ρ	0.5
Ingestion / reporting	L _{acq} / L _{rep}	18 / 6 ms
GPU peak / efficiency	P _{gpu} / η	8 TFLOPS / 0.25
Launch / step latency	L _o / τ_{seq}	4 / 0.6 ms
Max utilisation / vCPUs	u _{max} / V	0.8 / 4
Instance price	r _{gpu}	\$0.526/h
Egress price / event size	r _{egr} / s _e	\$0.087/GB / 1.2 KB
Egress share (cen. / dist.)	f _x	(K–1)/K / 0.05
Cross-cloud attack share	ϕ	0.30
Correlation recovery	κ	0.80
Sovereignty-restricted share	σ	0.25
Objective weights	w _f , w _l , w _c , w _s	0.40, 0.25, 0.15, 0.20
Latency SLA	L _{SLA}	100 ms

Prices are assumed pay-as-you-go figures for a T4-class GPU instance and inter-cloud egress.

B. Latency Comparison

Fig. 1 decomposes the end-to-end latency of Eq. 11 for the centralised design with K = 3 clouds. Cross-cloud transfer is the largest single stage, at 39.9 ms or 44% of Model D’s 90.3 ms total, so the placement of data matters more than the choice of network architecture. Inference differs strongly across models: 4.6 ms for the CNN against 24.4 ms for the LSTM, whose 32 sequential steps dominate its cost. Model B (98.3 ms) is the closest to the 100 ms SLA.

10.48047/jocaaa.2024.33.07.131

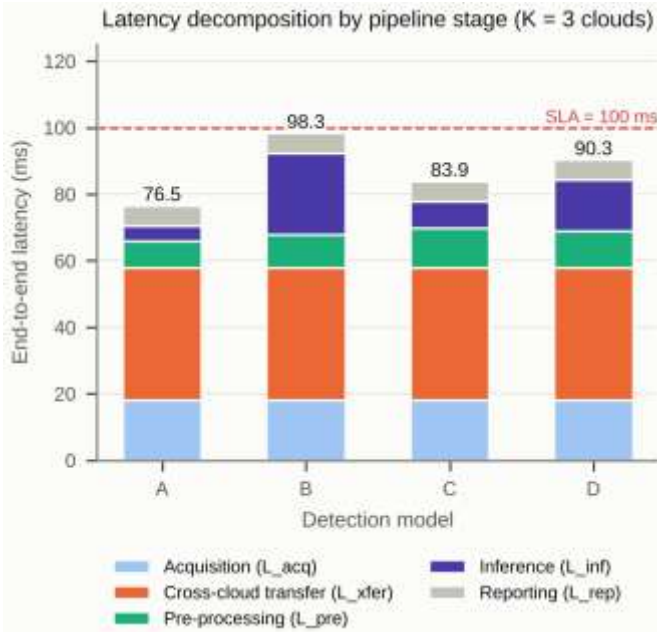


Fig. 1: **Latency comparison across detection models.** Stacked end-to-end latency by pipeline stage for centralised analysis over three clouds, computed from Eq. 11–15. The dashed line marks the 100 ms SLA.

C. Detection Performance

Fig. 2 shows F1-score convergence from Eq. 9. The CNN converges fastest ($\tau = 8$ epochs) but plateaus at $F1 = 0.9302$, whereas the hybrid model converges almost as quickly ($\tau = 12$) and reaches 0.9662. The Transformer converges slowest ($\tau = 22$).

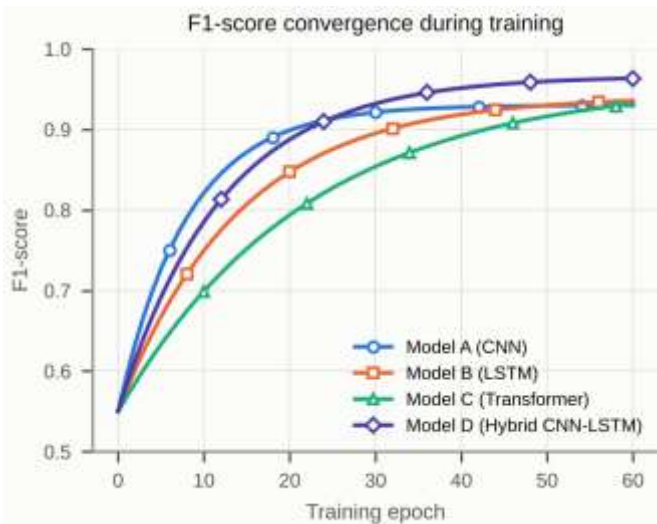


Fig. 2: **F1-score trend over training epochs.** Learning curves of Models A–D from Eq. 9; markers identify the models for grayscale printing.

The ROC curves in Fig. 3 follow Eq. 7. Model D attains the highest AUC (0.9988) and a recall of 0.979 at an FPR of 0.012, compared with 0.939 at 0.020 for Model A.

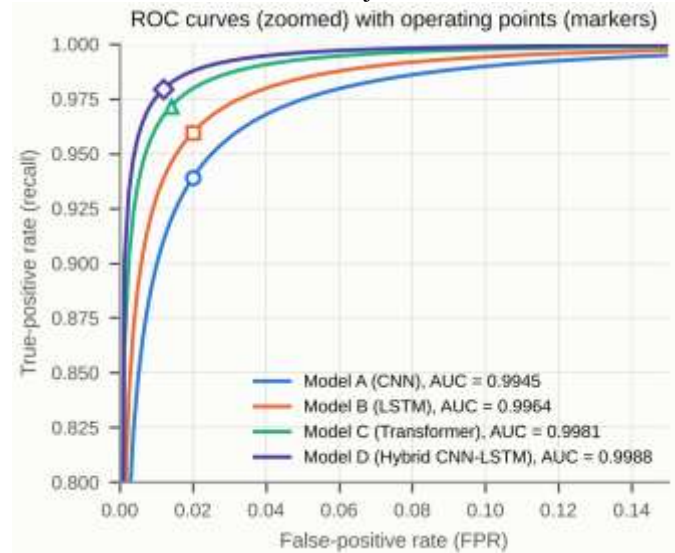


Fig. 3: **ROC curves and operating points.** Binormal ROC curves from Eq. 7, zoomed to $FPR \leq 0.15$. Markers show each model's operating point.

Class imbalance has a strong effect (Fig. 4, Eq. 8). At $\pi = 0.20$ Model D reaches $F1 = 0.966$, but at $\pi = 0.05$ it falls to 0.887 and at $\pi = 0.01$ to 0.618, because false positives from the benign majority swamp the few true detections. This supports the stratified sampling and SMOTE-style balancing discussed in the source review.

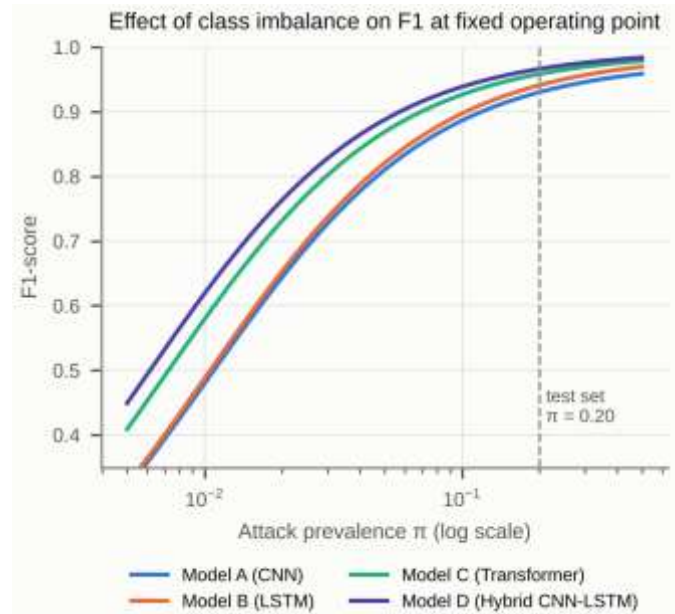


Fig. 4: **Effect of attack prevalence on F1-score.** F1-score of each model at its fixed operating point as attack prevalence varies (Eq. 8). The dashed line marks the test-set prevalence.

III. THREAT LANDSCAPE IN MULTI-CLOUD ENVIRONMENTS

The typical attack surfaces, underlying motivations, and modus operandi of actors aiming to compromise multi-cloud environments evolve in parallel with the development of new capabilities. Threat landscapes emerge from combinations of adversaries, targets, and incident types, serving as risk matrices that (ideally) help prioritize defenses. These threats

are particularly relevant for detection and prevention since they represent classes of incidents that automated mechanisms must address. Conversely, attacks may be mounted from external locations, including the Open Web. Several actors abuse poorly configured systems for storage, processing, or execution, perform malware and botnet management via exploited assets, or leverage such services to evade detection and attribution for malicious activities. Attackers seek to exploit the associated trust relations among clouds and between clouds and their tenants. For example, a vulnerability in the control plane of the cloud provider for a specific service may also affect the service customers.

Recent publications identify new vulnerabilities or exploits in multi-cloud ecosystems, assess the challenge of proper resource configuration by customers, highlight supply-chain-related risks in leveraging third-party services, and simulate the behavior of adaptive malware in such environments. Ritche, Shin, and Ibekwe argue that the multi-cloud strategy addresses specific organizational needs and challenges but creates its own set of concerns that future security solutions must tackle. For example, while layering multiple clouds provides a protective buffer against deployment breaches, that redundancy may welcome a deluge of supply-chain attacks targeting trusted third-party connections, rendered exploitable by automated and adaptive attack capabilities. These evolving threats require autonomous monitoring and response systems based on machine learning and artificial intelligence (AI) techniques.

A. Resource Utilisation

Fig. 5 reports utilisation of the provisioned fleet at $\lambda = 12,000$ events/s (Eq. 16, Eq. 17). Model D loads the GPU most (72.3%) because it needs only one instance, leaving the least headroom, while Model B needs 2 instances to stay under the 80% target. The Transformer is CPU- and memory-heavy (78.0% CPU, 2.25 GB) due to tokenisation and attention activations.

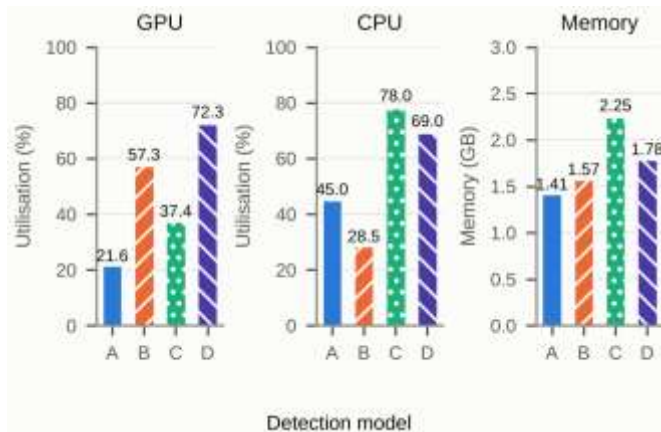


Fig. 5: **Resource utilisation of the detection models.** GPU utilisation, CPU utilisation and memory footprint of the provisioned fleet at 12,000 events/s, computed from Eq. 15–17.

B. Efficiency Metrics

Fig. 6 combines throughput, latency-normalised accuracy and cost-normalised accuracy (Eq. 21). The CNN has the highest throughput (55.5 k events/s) and F1 per 100 ms

(1.22), while Model D has the best F1 per USD (11.8), because it delivers the highest F1 at the same single-instance cost as Models A and C.

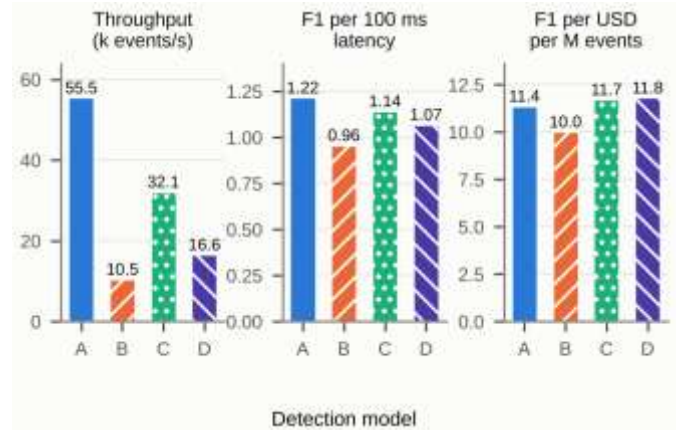


Fig. 6: **Efficiency metrics of the detection models.** Throughput, F1 per 100 ms of latency and F1 per USD per million events (Eq. 15 and Eq. 21).

C. Cost–Performance Trade-off

Fig. 7 compares centralised and distributed placement. Moving Model D from centralised to distributed analysis cuts its cost per million events by 49% and its latency by 25%, but lowers F1 by 3.2% (from 0.9662 to 0.9353), because cross-cloud attacks are only partly recovered by correlation (Eq. 10). Egress dominates the centralised bill (\$0.070 of \$0.082 per million events), a direct consequence of data gravity.

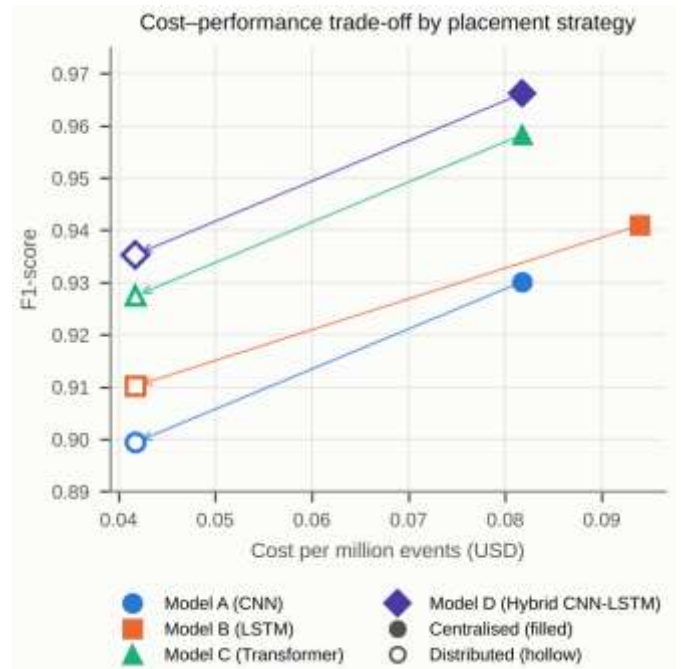


Fig. 7: **Cost versus performance by placement strategy.** F1-score against cost per million events for centralised (filled) and distributed (hollow) placement; arrows show the change when moving to distributed placement.

D. Scalability with the Number of Clouds

Fig. 8 varies the number of clouds K for Model D. Centralised latency grows with K through the fan-out term of

Eq. 12 and crosses the 100 ms SLA at $K = 6$, while distributed latency stays flat at 67.4 ms for $K \geq 2$. Centralised cost rises with K because a larger share of data crosses cloud boundaries.

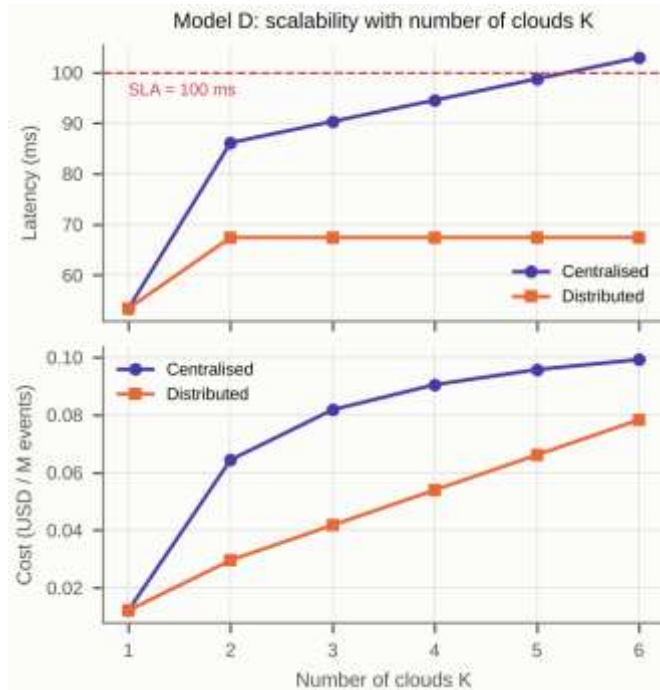


Fig. 8: Scalability of latency and cost with the number of clouds. Model D latency (top) and cost per million events (bottom) versus the number of clouds K for both placement strategies.

IV. DEEP LEARNING METHODOLOGIES FOR THREAT DETECTION

Deep learning techniques for security threat detection in multi-cloud environments are reviewed, answering the question: "What deep learning approaches have been applied to threat detection?" Grouped into supervised and non-supervised methods, a total of 17 articles show that (1) techniques based on convolutional and stacked autoencoder and restricted Boltzmann machine structures dominate, (2) evaluations usually exploit only part of the popular CICIDS2017 dataset, and (3) limited reproducibility, a shortage of publicly available code, and insufficient consideration of biases and ethical impacts suggest an urgent need for improvement.

Cloud computing provides multiple advantages, including low cost, high computational power, and scalability. Multi-cloud infrastructures utilize the services of several cloud service providers concurrently to tackle critical needs. However, this improved architectural flexibility comes at the expense of increased risk and a more complex threat landscape. Security incidents targeting multi-cloud deployments are already common, and it is widely accepted that traditional defense mechanisms can no longer protect complex information systems. Consequently, researchers have begun to explore the use of machine learning and deep learning for detecting anomalies and potential intrusions in multi-cloud environments.

Supervised classification forms the most common and applicable methodology for attack detection. Various architectures have been deployed in threat detection frameworks: Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) for network intrusion detection, Recurrent Neural Networks (RNN) for distributed denial-of-service attack detection, a transformer model for network anomaly detection, and a hybrid CNN-RNN approach for technique-based security event classification. As deep learning methodologies require large training datasets with ample examples for each class, synthetic datasets have been generated to facilitate model training for SSH brute-force, FTP brute-force, and denial-of-service attacks. Approaches are evaluated on different multiclass classification metrics, including confusion matrices, precision, recall, F1-score, and accuracy, and methods overcome class imbalance by using stratified sampling and the SMOTE algorithm. Model interpretability is an important research area, and attention-based models play a significant role in providing model explainability. The black-box nature of these deep networks is mitigated through explainable AI methods from domains such as computer vision. A few works enhance model robustness against adversarial attacks deployed at the data or model level using feature-squeezing, noise-injection, and adversarial-training techniques.

Deep learning methodologies can also be deployed to assist supervised classification, especially when labeled data is scarce or prohibitively expensive. Semi-supervised, few-shot, and zero-shot learning are gaining attention in attack detection frameworks, driven by the need to expand networks' operating envelopes where attack signatures are unknown. Data and feature augmentation methodologies have also been researched to address these issues, as have data-synthesis techniques that allow data generation across clouds while abiding by privacy requirements.

Table 3: Comparative Performance of Detection Models

Metric	Model A	Model B	Model C	Model D	Improv. (%)
Precision	0.9215	0.9230	0.9455	0.9533	+0.8
Recall	0.9390	0.9596	0.9715	0.9795	+0.8
F1-score	0.9302	0.9410	0.9583	0.9662	+0.8
Accuracy	0.9718	0.9759	0.9831	0.9863	+0.3
MCC	0.9125	0.9261	0.9478	0.9577	+1.0
AUC	0.9945	0.9964	0.9981	0.9988	+0.1
F1 at $\pi = 0.05$	0.8098	0.8203	0.8684	0.8874	+2.2

Computed from Eq. 1–8 at $N = 100,000$ and $\pi = 0.20$. Improv. = Model D versus the best of Models A–C.

Table 4: Error and Latency Metrics

Metric	Model A	Model B	Model C	Model D	Improv. (%)
FPR (%)	2.00	2.00	1.40	1.20	+14.3
FNR (%)	6.10	4.04	2.85	2.05	+28.1
Error rate (%)	2.82	2.41	1.69	1.37	+18.9
L _{inf} (ms)	4.6	24.4	8.0	15.4	-234.1
L _{total} , centralised (ms)	76.5	98.3	83.9	90.3	-18.0
L _{total} , distributed (ms)	53.6	75.4	61.0	67.4	-25.7
Throughput (k events/s)	55.5	10.5	32.1	16.6	-70.1

Errors from Eq. 4–5 and latency from Eq. 11–15 with $K = 3$. Lower is better except throughput.

V. DATA ACQUISITION, PREPROCESSING, AND FEATURE ENGINEERING

Interoperability creates a rich ecosystem with numerous data sources that relate to security incidents, including platform logs, network packets, telemetry data, threat intelligence feeds, alerts, and notifications from third-party security solutions. Consequently, a threat-detection pipeline can incorporate information from multiple clouds, enabling a more complete picture of an ongoing incident. However, the data housed in different clouds are inherently different. Therefore, the availability of high-quality labeled data for training remains one of the main bottlenecks preventing the adoption of deep-learning-based threat-detection systems in real-life scenarios. The labeling process is often cumbersome, and datasets for specific security problems may require substantial manual effort. Multiple clouds have varied data characteristics, including nomenclature for similar entities or activities and, at times, data that represent conflicting or contradictory information. In a multi-cloud setup, these challenges are amplified. A more practical approach to mitigating the risk of privacy leakage is through the use of active-learning techniques or advanced synthetic-data-generation methods. Therefore, external data sources, such as threat intelligence feeds, can also be valuable inputs for threat-detection models.

A large number of threat-detection tasks examine only a subset of the security signals available within a cloud or cloud network. Consequently, part of the information space relevant to a detection task is ignored, which may, at times, contain evidence that is critical for achieving reliable detection. An active learner can leverage the labels of only a small fraction of the training instances to achieve a smaller error rate than a random sampling strategy. However, the feasibility of generating informative labels relies on the skill of an expert for properly labeling only a few of the incoming instances. Furthermore, apart from generating spatial and temporal correlations, it is also essential to model the relationships between network entities and to have some contextual information. Several works have explored graph structures as a natural way of representing network data. Indeed, a network can be represented as a temporal graph in which the links appear and disappear over time, even though such representations can introduce more complexity into the learning process.

Multiclass evaluation in the reviewed work relies on confusion matrices, precision, recall, F1-score and accuracy. For a detector evaluated on N events with attack prevalence π , the confusion-matrix counts TP, FP, TN and FN yield the metrics below.

$$\text{Precision} = \frac{TP}{TP+FP} \quad (\text{Eq. 1})$$

where TP and FP are true and false positives. Precision is the share of raised alerts that correspond to real attacks.

$$\text{Recall} = \frac{TP}{TP+FN} \quad (\text{Eq. 2})$$

where FN counts missed attacks. Recall (the true-positive rate) is the share of attacks that are detected.

$$F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (\text{Eq. 3})$$

F1 is the harmonic mean of Eq. 1 and Eq. 2. It is the primary metric because the benign class dominates in practice, which the source review addresses with stratified sampling and SMOTE.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (\text{Eq. 4})$$

where TN counts correctly ignored benign events. Accuracy is reported for completeness, but it is dominated by the benign majority when π is small.

$$\text{FPR} = \frac{FP}{FP+TN}, \quad \text{FNR} = \frac{FN}{FN+TP} \quad (\text{Eq. 5})$$

FPR measures benign events wrongly flagged (analyst alert fatigue); FNR measures attacks that go undetected.

$$\text{MCC} = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \quad (\text{Eq. 6})$$

The Matthews correlation coefficient uses all four confusion-matrix cells and stays informative under class imbalance.

$$\text{TPR}(f) = \Phi(d' + \Phi^{-1}(f)), \quad \text{AUC} = \Phi\left(\frac{d'}{\sqrt{2}}\right) \quad (\text{Eq. 7})$$

where Φ is the standard normal CDF, f is the false-positive rate and d' is the separability index of a detection model. A larger d' gives a higher recall at the same FPR; the alert threshold selects the operating point on this curve.

$$P(\pi) = \frac{\pi \text{TPR}}{\pi \text{TPR} + (1-\pi) \text{FPR}} \quad (\text{Eq. 8})$$

where π is attack prevalence. Eq. 8 shows how precision, and therefore F1, falls as attacks become rarer, even with fixed TPR and FPR.

$$F_1(t) = F_\infty - (F_\infty - F_0) \exp\left(-\frac{t}{\tau_m}\right) \quad (\text{Eq. 9})$$

where t is the training epoch, F_∞ is the converged F1 from Eq. 3, F_0 is the initial F1 and τ_m is the convergence time constant of model m .

$$\text{TPR}_{\text{dist}} = \text{TPR} (1 - \varphi + \varphi \kappa) \quad (\text{Eq. 10})$$

where φ is the share of attacks that span several clouds and κ is the fraction of those recovered by alert correlation. When each cloud scores only its own telemetry, cross-cloud attacks can escape detection, a risk the source review highlights.

B. End-to-End Latency Model

Detection latency is decomposed along the four process activities. Logs and traces are ingested through provider

APIs, normalised from the schema-less data lake (telemetry aligned to the Azure Monitor schema), scored by the model and reported as incidents.

$$L_{total} = L_{acq} + L_{xfer} + L_{pre} + L_{inf} + L_{rep} \quad (\text{Eq. 11})$$

where L_{acq} is provider-API ingestion, L_{xfer} is cross-cloud transfer, L_{pre} is schema normalisation or tokenisation, L_{inf} is model inference and L_{rep} is incident reporting (all in ms).

$$L_{xfer}^{cen} = L_{loc} + \mathbb{1}_{K>1} \left[RTT(1 + \delta(K-1)) + \frac{S}{B} \right] \quad (\text{Eq. 12})$$

Centralised analysis pulls telemetry from the other $K-1$ clouds. L_{loc} is in-cloud access, RTT is the inter-cloud round-trip time, δ is the fan-out penalty per additional cloud, S is the window payload, B is link bandwidth and $\mathbb{1}_{[K>1]}$ is an indicator that removes the term for a single cloud.

$$L_{xfer}^{dist} = L_{loc} + \mathbb{1}_{K>1} \rho RTT \quad (\text{Eq. 13})$$

Distributed analysis keeps raw data in place and exchanges only alerts; ρ is the fraction of an RTT spent on alert correlation. This also keeps sensitive data inside its jurisdiction.

$$L_{inf} = \frac{W F_m}{\eta P_{gpu}} + L_0 + T_m \tau_{seq} \quad (\text{Eq. 14})$$

where W is the window size in events, F_m is the compute per event of model m , η is hardware efficiency, P_{gpu} is peak GPU throughput, L_0 is launch overhead, T_m is the number of sequential recurrent steps and τ_{seq} is the latency per step.

$$\theta_m = \frac{W}{L_{inf}}, \quad n_m = \left\lceil \frac{\lambda}{\theta_m u_{max}} \right\rceil \quad (\text{Eq. 15})$$

where θ_m is the throughput of one instance, λ is the event arrival rate, u_{max} is the target maximum utilisation and n_m is the number of instances required.

C. Resource and Cost Model

$$U_{gpu} = \frac{\lambda}{n_m \theta_m}, \quad U_{cpu} = \frac{\lambda t_{cpu}}{n_m V} \quad (\text{Eq. 16})$$

where V is the number of vCPUs per instance and t_{cpu} is CPU time per event; U_{gpu} and U_{cpu} are the resulting utilisations of the provisioned fleet.

$$M_m = M_0 + 4 p_m + W a_m \quad (\text{Eq. 17})$$

where M_0 is the runtime memory (GB), p_m is the parameter count (millions, stored as 4-byte floats) and a_m is activation memory per event (MB).

$$C_m = \frac{n_m r_{gpu}}{3600} \cdot \frac{10^6}{\lambda} + s_e f_x r_{egr} \quad (\text{Eq. 18})$$

where r_{gpu} is the instance price per hour, s_e is the raw event size in KB (so 10^6 events carry s_e GB), f_x is the share of data crossing a cloud boundary and r_{egr} is the egress price per GB. Compute is billed on provisioned capacity; egress reflects data gravity.

D. Comparative Metrics and Placement Optimisation

$$\Delta^\dagger = \frac{x_D - x^*}{x^*} \times 100 \quad (\text{Eq. 19})$$

For metrics where higher is better, x_D is the value of Model D and x^* is the best value among Models A–C.

$$10.48047/jocaaa.2024.33.07.131$$

$$\Delta^\dagger = \frac{x^* - x_D}{x^*} \times 100 \quad (\text{Eq. 20})$$

For metrics where lower is better (latency, error, cost). A positive Δ means Model D is better; a negative Δ means it is worse.

$$\eta_L = \frac{F_1}{L_{total}/100}, \quad \eta_C = \frac{F_1}{C_m} \quad (\text{Eq. 21})$$

η_L is F1 per 100 ms of latency and η_C is F1 per USD per million events, using Eq. 3, Eq. 11 and Eq. 18.

$$J_{m,g} = w_f (1 - \hat{F}_1) + w_l \hat{L}_{total} + w_c \hat{C}_m + w_s \hat{E}_s \quad (\text{Eq. 22})$$

where $g \in \{\text{centralised, distributed}\}$ is the placement strategy, $E_s = \sigma f_x$ is the exposure of sovereignty-restricted data (σ is its share of the data), w are weights and the hat denotes normalisation by the largest candidate value.

$$(m^*, g^*) = \text{argmin}_{m,g} J_{m,g} \quad (\text{Eq. 23})$$

The preferred model and placement minimise the weighted objective.

$$\text{s.t. } L_{total} \leq L_{SLA}, \quad E_s = \sigma f_x \quad (\text{Eq. 24})$$

The solution must meet the latency service-level agreement L_{SLA} .

VI. CONCLUSION

In accordance with the National Institute of Standards and Technology, Cloud Service Providers are required to assist users of multi-cloud infrastructure in detecting and responding to Attacks as a Service (AaaS) and other attacks. The ability to defend against evolved methods poses a challenge to managed security providers. Detection frameworks to respond to the evolving malware-as-a-service and AaaS threats against cloud and multi-cloud environments remain elusive.

The data sources from Amritapuri CloudLab, Microsoft Azure, and Amazon Web Services provide the infrastructure to implement deep learning-based detection frameworks. The approach enables transferring information for detection across service providers and facilitates protecting assets in hybrid-cloud, multi-cloud, and cloud-bursting scenarios. Existing deep learning-based methods employ commonly available datasets for cloud service models; however, these datasets have been created without real-time active threats originating from attackers providing AaaS through their cloud infrastructure. The detection framework responds to an AaaS threat injection scenario in which a cloud environment is breached with AaaS by an attacker on the other cloud using either malware-as-a-service or AaaS. The injected malware is then employed actively in a live scenario against the cloud or multi-cloud environment through Internet Control Message Protocol ping requests, L7 HTTP requests, or L4 socket layer L7 requests.

REFERENCES

- [1] Ahmad, Z., Khan, A. S., Shiang, C. W., Abdullah, J., & Ahmad, F. (2023). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, 32(1), e4150.
- [2] Loganathan, R. (2022). Converging Security Architecture and Compliance Management in Enterprise Data Center Ecosystems: A Unified Control Framework. *International Journal of Scientific Research and Modern Technology*, 1(12), 295-312.

10.48047/jocaaa.2024.33.07.131

- [3] Mondeddula, S. R. (2023). Natural Language Processing for Intelligent Automation of Financial Documents and Banking Operations. *International Journal of Intelligent Systems and Applications in Engineering*, 11, 605-612.
- [4] Aitha, A. R. (2022). Deep Neural Networks for Property Risk Prediction Leveraging Aerial and Satellite Imaging. *International Journal of Communication Networks and Information Security (IJCNIS)*, 14(3), 1308-1318.
- [5] Ferrag, M. A., Maglaras, L., Moschogiannis, S., & Janicke, H. (2023). Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. *Journal of Information Security and Applications*, 50, 102419.
- [6] Yandamuri, U. S. (2022). Cloud-Based Data Integration Architectures for Scalable Enterprise Analytics. *International Journal of Intelligent Systems and Applications in Engineering*, 10, 472-483.
- [7] Penaka, B. (2023). AI-Driven Predictive Analytics for Early Detection of High-Risk Patients in Cloud-Based Healthcare Systems. *International Journal of Medical Toxicology and Legal Medicine*, 26(3), 118-124.
- [8] Khraisat, A., Gondal, I., Vamplew, P., & Kamruzzaman, J. (2023). Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity*, 2, 20.
- [9] Mattaparthi, R. (2023). Connected Fleet Intelligence: Edge-Centric Analytics and Computer Vision for Predictive Manufacturing and Asset Resilience. *International Journal of Advanced Research in Computer Science & Technology (IJARCS)*, 6(5), 9077-9088.
- [10] Mahdaviifar, S., & Ghorbani, A. A. (2023). Application of deep learning to cybersecurity: A survey. *Neurocomputing*, 347, 149-176.
- [11] Mangalampalli, B. M. Intelligent Data Profiling for Healthcare Data Lakes Using AI-Enhanced Analytics.
- [12] Maseer, Z. K., Yusof, R., Bahaman, N., Mostafa, S. A., & Foozy, C. F. M. (2023). A survey on recent intrusion detection systems in cloud environment. 2023 IEEE 6th International Conference on Cloud Computing and Artificial Intelligence: Technologies and Applications, 1-6.
- [13] Reddy, V. A. R. (2023). API-First Design As A Strategy For Healthcare System Interoperability. *South Eastern European Journal of Public Health*, 224-247. <https://doi.org/10.70135/seejph.vi.7128>
- [14] Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2023). Deep learning approach for intelligent intrusion detection system. *IEEE Access*, 7, 41525-41550.
- [15] Mukesh, A., & Aitha, A. R. (2021). Insurance Risk Assessment Using Predictive Modeling Techniques. *International Journal of Emerging Research in Engineering and Technology*, 2(4), 68-79.
- [16] Aldweesh, A., Derhab, A., & Emam, A. Z. (2022). Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. *Knowledge-Based Systems*, 189, 105124.
- [17] Mattaparthi, R. (2021). Unified Data Lineage and Quality Governance Framework for Multi-Source Sensor Streams in Heavy-Duty Powertrain Manufacturing. *Online Journal of Mechanical Engineering*, 1(1), 1-15.
- [18] Ferrag, M. A., Shu, L., Djallel, H., & Choo, K.-K. R. (2022). Deep learning-based intrusion detection for distributed denial of service attack in agriculture 4.0. *Electronics*, 10(11), 1257.
- [19] Potturi, P. (2023). Cloud-Driven Business Intelligence Modernization for Scalable Enterprise Data Integration and Decision Support. *International Journal of Intelligent Systems and Applications in Engineering*, 11, 387.
- [20] Kolla, S. H., & Loganathan, R. (2023). Cloud-Native Deep Learning Architectures For Secure Generative AI Deployment In Enterprise Workflow Platforms. *Journal of International Crisis and Risk Communication Research*, 603-618.
- [21] Mangalampalli, B. M. Generative AI Applications In Healthcare Data Mart Design And Optimization.
- [22] Reddy, V. A. R. (2023). Orchestrating the Future Autonomous Healthcare Data Pipeline Management through Agentic AI Architectures. *International Journal of Advanced Research in Computer Science & Technology (IJARCS)*, 6(2), 7979-7992.
- [23] Hodo, E., Bellekens, X., Hamilton, A., Dubouilh, P.-L., Iorkyase, E., Tachtatzis, C., & Atkinson, R. (2022). Threat analysis of IoT networks using artificial neural network intrusion detection system. 2016 International Symposium on Networks, Computers and Communications, 1-6.
- [24] Sasidhar Reddy Mondeddula. (2022). AI-Driven Fraud Detection Using Ensemble Machine Learning for Secure Digital Banking Transactions. *International Journal of Intelligent Systems and Applications in Engineering*, 10(3s), 597-605. Retrieved from <https://ijisae.org/index.php/IJISAE/article/view/8515>
- [25] Liu, H., & Lang, B. (2022). Machine learning and deep learning methods for intrusion detection systems: A survey. *Applied Sciences*, 9(20), 4396.
- [26] Mishra, P., Varadharajan, V., Tupakula, U., & Pilli, E. S. (2022). A detailed investigation and analysis of using machine learning techniques for intrusion detection. *IEEE Communications Surveys & Tutorials*, 21(1), 686-728.
- [27] Otoum, S., Kantarci, B., & Mouftah, H. T. (2022). On the feasibility of deep learning in sensor network intrusion detection. *IEEE Networking Letters*, 1(2), 68-71.
- [28] Dixit, P., & Silakari, S. (2021). Deep learning algorithms for cybersecurity applications: A technological and status review. *Computer Science Review*, 39, 100317.
- [29] Mattaparthi, R. (2022). Engineering Predictive Industrial Systems Through IoT-Driven Asset Monitoring and Machine Learning Prognostics. *International Journal of Future Innovative Science and Technology (IJFIST)*, 5(1), 7790.
- [30] Gümüşbaş, D., Yıldırım, T., Genovese, A., & Scotti, F. (2021). A comprehensive survey of databases and deep learning methods for cybersecurity and intrusion detection systems. *IEEE Systems Journal*, 15(2), 1717-1731.
- [31] Mondeddula, S. R. (2022). Machine Learning-Based Credit Risk Assessment For Intelligent Lending and Default Prediction in Financial Services. *Journal of International Crisis and Risk Communication Research*, 5(1), 113-121. <https://doi.org/10.63278/jicrcr.v5i1.3820>
- [32] Hindy, H., Brosset, D., Bayne, E., Seeam, A. K., Tachtatzis, C., Atkinson, R., & Bellekens, X. (2021). A taxonomy of network threats and the effect of current datasets on intrusion detection systems. *IEEE Access*, 8, 104650-104675.
- [33] Kasongo, S. M., & Sun, Y. (2021). Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset. *Journal of Big Data*, 7, 105.
- [34] Yandamuri, U. S. (2021). A Comparative Study of Traditional Reporting Systems versus Real-Time Analytics Dashboards in Enterprise Operations. *Universal Journal of Business and Management*, 1(1), 1-13.
- [35] Kim, A., Park, M., & Lee, D. H. (2021). AI-IDS: Application of deep learning to real-time web intrusion detection. *IEEE Access*, 8, 70245-70261.
- [36] Ring, M., Wunderlich, S., Gründ, D., Landes, D., & Hotho, A. (2021). Flow-based benchmark data sets for intrusion detection. *Proceedings of the 16th European Conference on Cyber Warfare and Security*, 361-369.
- [37] Mangalampalli, B. M. (2023). AI-Driven Anomaly Detection in Healthcare Claims Data: A Business Intelligence Perspective. *Journal of Rare Cardiovascular Diseases*.
- [38] Rajesh Reddy Ellenki. (2022). Predictive Demand Forecasting Framework for Adaptive Retail Inventory Planning . *International Journal of Intelligent Systems and Applications in Engineering*, 10(2s), 412-418. Retrieved from <https://ijisae.org/index.php/IJISAE/article/view/8543>
- [39] Reddy, V. A. R. (2023). Predictive Healthcare Administration Using Advanced Payer Analytics and Population Health Data Engineering. *International Journal of Advanced Research in Computer Science & Technology (IJARCS)*, 6(2), 7967-7978.
- [40] Loganathan, R. (2021). Integrated Risk and Compliance Frameworks for Global Data Center Operations: A Governance-Centric Approach. *Universal Journal of Computer Sciences and Communications*, 1(1), 1-26.
- [41] Aleesa, A. M., Zaidan, B. B., Zaidan, A. A., & Sahar, N. M. (2020). Review of intrusion detection systems based on deep learning techniques: Coherent taxonomy, challenges, motivations, recommendations, substantial analysis and future directions. *Neural Computing and Applications*, 32, 9827-9858.