Enhancing IoT Cybersecurity with Multi-Layer Deep Transfer Learning Approach for Intrusion Detection

Anuj Rapaka[1,*], Govindan Manoharan Karthik[2], Balla Sudhir[3], Gurram Venkata Naga Bhagya Sree[4], Narendra Kumar[5] & Jyothi Nelahonne Mohan[6]

[1]Department of Computer Science and Engineering, Shri Vishnu Engineering College for Women (Autonomous), Bhimavaram, 534202, India
[2]Department of Information Security,School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, Katpadi - Thiruvalam Road, Vellore - 632 014, Tamil Nadu, India.
[3]Department of Electronics and Communication, International School of Technology and Sciences for Women, NH-16, Eastgonagudem, Rajanagaram, Rajamahendravaram, AP-533294, India
[4]Department of Computer Science, Anil Neerukonda Institute of Technology & Sciences (ANITS),Sangivalasa, Bheemunipatnam, Visakhapatnam, Andhra Pradesh-531162, India
[5]Department of CSE, Amity University Jharkhand, Ranchi, 835303, Jharkhand, India.
[6]Department of Computer Science Engineering, Koneru Lakshmaiah Education Foundation,Vaddeswaram, Andhra Pradesh, 522502, India.
*E-mail: anuj.rapaka24@gmail.com

Abstract. Intrusion detection in IoT-enabled cloud environments is challenged by high-dimensional traffic, class imbalance, and limited labeled data. This paper proposes a hybrid framework combining Golden Jackal–Grey Wolf Optimization (GJO-GWO) for feature selection with a Kernel Mean Alignment Autoencoder (KMA-AE) for deep transfer learning. GJO-GWO selects a compact, discriminative feature subset, while KMA-AE aligns source and target latent representations to mitigate distribution mismatch. Experiments on the CIDDS-001 dataset achieve 90.21% accuracy and 0.90 macro-F1, with improved precision–recall for minority attacks and a 60% feature reduction. Although training is more expensive, the method attains the lowest inference time, enabling real-time deployment. Overall, the framework provides an effective and generalizable intrusion detection solution for dynamic IoT environments.

Keywords: *auto encoders; golden jackal-grey wolf hybrid optimization algorithm; intrusion detection systems; kernel mean alignment; malicious events.*

1 Introduction

Information and communication technologies underpin modern organizational operations but have significantly increased cyber-risk with the growth of IoT-enabled networks [1-3]. Intrusion Detection Systems (IDSs) are therefore

Received June 26th, 2025, Revised January 20th, 2026, Accepted for publication March 3rd, 2026.

essential for identifying malicious activities, commonly implemented as host-based (HIDS) or network-based systems (NIDS), with NIDS analysing aggregated traffic features over time [4,5]. Traditional IDS approaches include signature-based methods, which detect known attacks but fail against novel threats, and anomaly-based methods, which model normal and abnormal behaviour and are more effective when combined with AI techniques [6–8]. However, evolving attack patterns, dynamic IoT environments, and limited labelled data severely degrade the effectiveness of conventional security mechanisms and static IDS models [9-13]. These challenges demand adaptive, learning-based IDS solutions capable of operating under weak supervision and real-time constraints [14-18]. To address this, this paper proposes KMA-AE, a deep transfer learning–based IDS that applies Kernel Mean Alignment (KMA) across all encoder layers to reduce distribution mismatch between labelled source data and unlabelled target traffic. Unlike bottleneck-only alignment, multi-layer adaptation enhances generalization in data-scarce environments. In addition, a Golden Jackal–Grey Wolf Optimization (GJO-GWO) algorithm is introduced for feature selection, yielding a compact and discriminative feature subset while improving convergence stability and reducing computational overhead. The primary contributions of this work are summarized as follows:

1. Development of a deep transfer learning–based IDS using KMA-AE to enable effective knowledge transfer between labelled and unlabelled IoT traffic data.
2. Introduction of multi-layer KMA-based feature alignment to enhance generalization beyond bottleneck-level adaptation.
3. Proposal of a hybrid GJO-GWO feature selection strategy to improve detection accuracy, convergence speed, and computational efficiency.
4. Extensive evaluation on public IoT intrusion datasets demonstrating the framework's effectiveness, scalability, and robustness.

The remainder of this paper is organized as follows: Section 2 reviews related work, Section 3 describes the proposed methodology, Section 4 presents experimental results and analysis, and Section 5 concludes the paper.

2 Related works

Although existing IDS solutions achieve high detection accuracy through deep learning and metaheuristic optimization, several limitations remain. Many approaches rely on fully supervised learning and require large volumes of labelled data, which are rarely available in real-world IoT environments. Class imbalance is often addressed via synthetic oversampling, potentially disrupting temporal dependencies in network traffic. Moreover, most transfer learning methods align only bottleneck representations, limiting robustness under domain shift, while feature selection and representation learning are typically optimized

independently. Explainability is also underexplored, with limited insight into feature relevance and transfer behaviour. In contrast, the proposed framework jointly addresses these challenges by combining hybrid GJO–GWO feature optimization with a multi-layer Kernel Mean Alignment–based Autoencoder (KMA-AE), enabling efficient feature reduction, robust cross-domain knowledge transfer, and improved detection performance under weak supervision and temporal constraints.

3 Proposed model

The goal is to learn a mapping function $f(\cdot)$that transforms raw network traffic features into a compact latent representation such that:

1. discriminative information relevant to intrusion detection is preserved,
2. redundant and noisy features are suppressed,
3. the statistical distribution discrepancy between source and target domains is minimized, and
4. detection performance is maximized under class imbalance and limited supervision.

Accordingly, the learning objectives of this study are threefold. First, an optimal subset of informative traffic features is selected using a hybrid Golden Jackal–Grey Wolf Optimization (GJO–GWO) strategy to reduce dimensionality and computational complexity while maintaining classification capability. Second, a deep transfer learning model based on a Kernel Mean Alignment Autoencoder (KMA-AE) is developed to align latent feature distributions across source and target domains at multiple encoding layers, rather than restricting adaptation to the bottleneck layer. Third, the integrated framework aims to achieve robust intrusion detection and attack categorization in temporally evolving IoT traffic, ensuring strong generalization to unseen and weakly labelled environments.

1. Discriminative intrusion-related information is preserved,
2. Feature redundancy is minimized,
3. Source–target distribution mismatch is reduced, and
4. Detection performance is maximized under class imbalance.

The learning objectives are summarized as follows:

1. Feature Optimisation: Employ Golden Jackal–Grey Wolf Optimization (GJO-GWO) to identify a minimal yet informative feature subset, improving accuracy and reducing complexity.
2. Knowledge Transfer: Achieve domain-invariant representations by aligning source and target distributions across multiple encoding layers using Kernel Mean Alignment (KMA).

3. Intrusion Detection: Develop a deep transfer learning classifier capable of distinguishing benign and malicious traffic and categorising attack types with limited labels.
4. Generalisation: Preserve temporal dependencies to ensure robust detection across varying traffic windows in IoT environments.

By jointly optimizing feature selection and multi-layer transfer learning, the proposed KMA-AE framework provides a scalable and generalizable intrusion detection solution suitable for real-world IoT deployments. Furthermore, this study evaluates multiple intrusion detection strategies, including multi-flow–based methods, on the CIDDS-001 dataset to systematically analyses the effects of temporal dependencies, window size selection, and algorithmic efficiency on attack detection and classification performance.

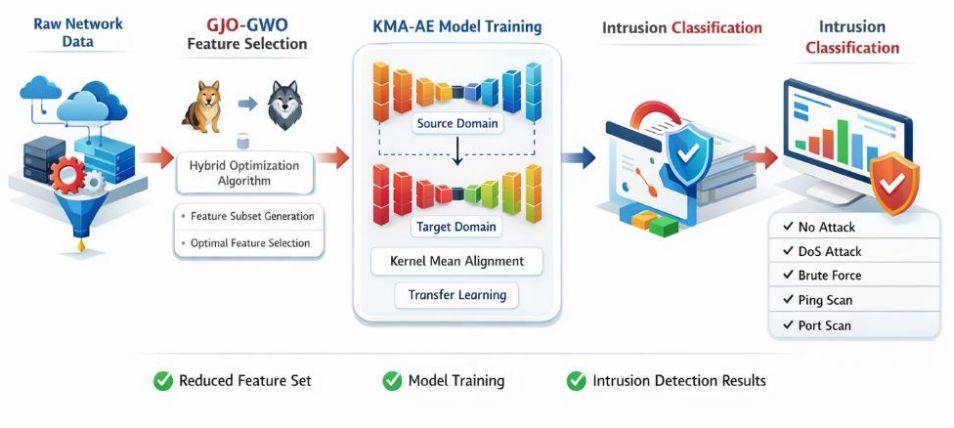

Figure 1 Overall proposed framework.

3.1 Dataset Description

The Coburg Intrusion Detection Dataset (CIDDS-001) [25] contains approximately four weeks of real-world NetFlow traffic, comprising nearly 33 million records collected from an OpenStack-based environment between March 3 and April 18, 2017.

The dataset captures interactions among multiple clients and standard services (e.g., web and email servers) and includes labelled flows representing normal traffic, malicious activity, and port-scan attacks. Publicly available Python scripts are provided for traffic generation and labelling. Owing to its large scale, realistic traffic behaviour, and inclusion of modern attack types, CIDDS-001 is a widely used benchmark for network-based intrusion detection research.

The dataset follows a standard NetFlow structure, where attributes 1–12 correspond to core traffic features and attributes 13–16 are derived from the labelling process. A summary of dataset characteristics and attack-class distribution is presented in Tables 1 and 2, respectively.

Table 1 Requirement of CIDDS-001 dataset topographies.

Description	Name
Basis IP Address	Src IP
Kind of Service	Tos
Not stated	Flows
Basis Port	Src Port
Terminus IP Address	Dest IP
Start period flow first seen	Date first seen
Period of flow	Duration
Sum of communicated bytes	Bytes
Sum of conveyed packets	Packets
OR concatenation of all TCP Flags	Flags
Destination Port	Dest Port
Transport Protocol (e.g., UDP)	Proto
Class label: Victim, Suspicious, and Unknown.	Class
Category of Attack (DoS, PingScan)	AttackType
Unique attack id. The same class has the same attack id.	AttackID
Contains additional information about attack parameters, such as the total number of attempted password deductions for SSH-Brute attacks.	AttackDescription

Table 2 Attack delivery over period.

	Open Stack				Outside Server			
Week	**DoS**	**Brute-Force**	**Ping Scan**	**Port Scan**	**Dos**	**Brute-Force**	**Ping Scan**	**Port Scan**
1st	Yes	Yes	Yes	Yes	No	No	No	No
No	Yes	Yes	Yes	No	Yes	No	Yes	Yes
No	No	No	No	No	Yes	No	Yes	Yes
No	No	No	No	No	Yes	No	No	Yes

3.2 Data Preprocessing besides Sampling

The CIDDS-001 dataset was cleaned by removing redundant attributes and converting non-numeric fields into numeric form. Features were encoded and normalized using min–max scaling, while temporal order was preserved via flow timestamps. To maintain sequential dependencies and include all attack types, a contiguous three-day window (March 17–20, 2017) comprising 2,535,456 flows

was selected instead of random sampling. This subset preserves representative class proportions while significantly reducing computational cost.

$$x' = \frac{(x - x_{min})}{(x_{max} - x_{min})} \quad (1)$$

Table 3 Attack Type class distribution contrast.

Class	OpenStack 1st Half	Dataset	Sample
Total Records	18,762,253(100%)	32,630,424(100%)	2,535,456(100%)
Brute Force	4992(0.03%)	9888(0.03%)	126(0.05%)
Ping Scan	6090(0.03%)	6090(0.02%)	1068(0.04%)
Port Scan	265,918(1.41%)	303,356(0.93%)	50,36(1.98%)
DoS	2,959,027(15.77%)	2,959,027(9.07%)	390,440(15.40%)
No Attack	15,526,226(82.75%)	29,352,063(89.95%)	2,092,550(82.53%)

3.2.1 Justification of Temporal Window Selection

The CIDDS-001 dataset contains four weeks of continuous flow-level traffic from an OpenStack environment, including both benign and malicious activities. In this study, a temporally contiguous three-day window (March 17–20, 2017) was selected to balance representativeness, temporal dependency preservation, and computational feasibility. This interval includes all attack categories considered DoS, Brute Force, Ping Scan, Port Scan, and No Attack while maintaining manageable data volume. Preserving flow sequence is essential, as the proposed framework analyses sequential attack behavior, where malicious activities evolve across consecutive flows.

Random or stratified sampling would disrupt temporal continuity and obscure such dependencies. Moreover, training on the full four-week dataset would introduce concept drift due to changing network conditions, potentially confounding feature selection and transfer learning evaluation. The chosen window enables learning stable short-term attack dynamics under realistic conditions, aligning with the study's focus on accurate intrusion detection rather than long-term seasonal modelling.

3.3 Feature Selection using Golden Jackal-Grey Wolf Optimization Procedure

A hybrid GJO GWO optimizer [26] is adopted for feature selection, combining the fast individual exploration of Golden Jackal Optimization with the strong global search capability of Grey Wolf Optimization. Lagrange interpolation is integrated to enhance convergence stability and optimization accuracy, ensuring an effective balance between exploration and exploitation.

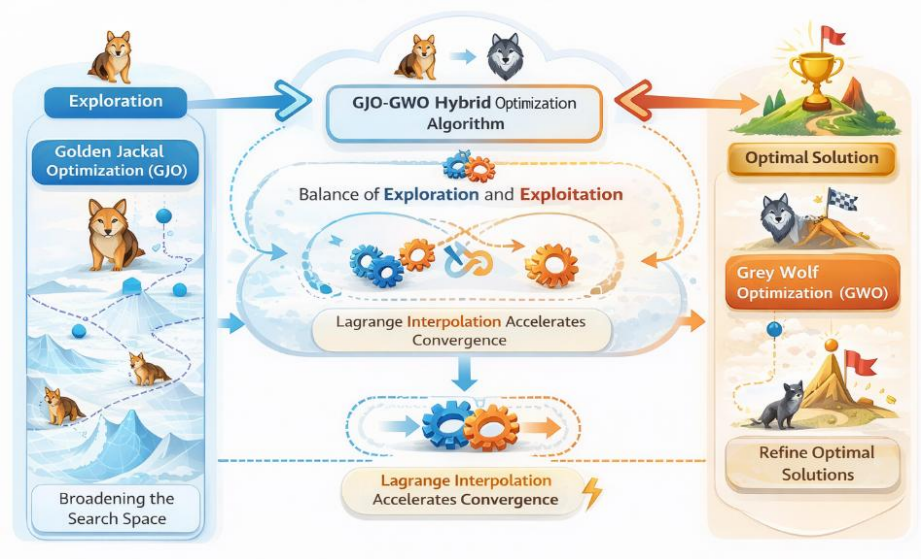

Figure 2 GJO-GWO hybrid optimization flow.

3.3.1 A variation of GJO and GWO

1. The GWO procedure.

Grey Wolf Optimization (GWO) is a population-based metaheuristic inspired by the social hierarchy and cooperative hunting behavior of grey wolves. The population is divided into four levels α (leader), β, δ, and ω where the α wolf guides the search process, while the remaining wolves assist in encircling and attacking the prey. During optimization, wolves collaboratively locate, pursue, and surround the target solution. This behavior is mathematically modelled to iteratively update candidate solutions, enabling effective exploration and exploitation of the objective search space.

$$D_P = \left|C_p X_p(t) - X(t)\right| \tag{1}$$

$$X_p(t+1) = X_p(t) - A_p . D_p \tag{2}$$

Where *t* denotes the iteration index. α, β, and δ represent the leading wolves with positions **X_α**, **X_β**, and **X_δ**, while **X_p(t)** denotes the prey position. The distance vectors quantify how wolves progressively encircle the prey, reflecting the cooperative hunting behaviour of grey wolves. The α, β, and δ wolves guide the search process.

2. Wolf position update:

Each wolf updates its position based on the guidance of α, β, and δ, enabling coordinated convergence toward the prey.

$$X(t+1) = \frac{X_p(t+1)}{3} = \frac{X_1+X_2+X_3}{3} \quad (3)$$

In this equation, $X(t+1)$represents the updated position of a grey wolf at iteration $t+1$. The algorithm iterates until successive position updates show negligible change, indicating convergence and successful prey capture.

3.3.1.1 Introducing the α-wolf in GJO procedure.

The Golden Jackal Optimization (GJO) algorithm is enhanced by integrating the hierarchical hunting strategy of Grey Wolf Optimization (GWO). Unlike jackals, which hunt cooperatively without strict leadership, GWO employs an alpha wolf to guide the search process. In the proposed IGJO framework, the alpha wolf is introduced as a guiding agent that directs jackals toward promising regions of the search space, improving convergence speed and stability while ignoring inter-population competition.

This hierarchical guidance refines the iterative update mechanism and strengthens the balance between exploration and exploitation. Details of the modified update process are presented in the next section.

1. Searching and tracking prey iterative hunt procedure.

$$Y_1(t) = Y_M(t) - E.|Y_M(t) - rl.Prey(t)| \quad (4)$$

$$Y_2(t) = Y_{FM}(t) - E.|Y_{FM}(t) - rl.Prey(t)| \quad (5)$$

$$Y_3(t) = Y_\alpha(t) = Y_\alpha(t) - A|Y_\alpha(t) - C.Pery(t)| \quad (6)$$

2. Surrounding, besides attacking prey—iterative estimate procedure.

$$Y_1(t) = Y_M(t) - E.|rl.Y_M(t) - Prey(t)| \quad (7)$$

$$Y_2(t) = Y_{FM}(t) - E.|rl.Y_{FM}(t) - Prey(t)| \quad (8)$$

$$Y_3(t) = Y_\alpha(t) = Y_\alpha(t) - A|C.Y_\alpha(t) - Pery(t)| \quad (9)$$

Introducing an alpha wolf into the GJO framework converts jackal-only cooperation into a hierarchical hunting strategy.

The alpha wolf improves early exploration and later exploitation by guiding prey search and encirclement, accelerating convergence and increasing solution accuracy. Although additional parameters are introduced, their effect is reduced by reformulating the update equations using the original GJO strategy, resulting in improved optimization with minimal complexity increase.

$$Y_3(t) = Y_a(t) - E.|Y_a(t) - rl.Prey(t)| \quad (10)$$

$$Y_3(t) = Y_a(t) - E.|rl.Y_a(t) - Prey(t)| \quad (11)$$

3.3.2 Collaborative updating apparatus of GJO-WOA Lagrange exclamation

To embed the GWO hierarchical leadership into the GJO framework, the population update rule is revised to explicitly account for the α-wolf. Unlike the original GJO, where convergence is defined solely by coincident male–female jackal positions, the enhanced model introduces a three-component population comprising male jackals, an α-wolf, and cooperative grey-wolf guidance.
Their interactions are jointly modelled through a unified interpolation-based position update equation, resulting in improved convergence stability and optimization efficiency.

$$Y(t+1) = \frac{(Y(t)-Y_2(t))(Y(t)-Y_3(t))}{(Y_1(t)-Y_2(t))(Y_1(t)-Y_3(t))}.\frac{Y_1(t)}{3} + \frac{(Y(t)-Y_1(t))(Y(t)-Y_3(t))}{\left((Y_2(t)-Y_1(t))\right)\left((Y_2(t)-Y_3(t))\right)}.\frac{Y_2(t)}{3} + \frac{(Y(t)-Y_1(t))(Y(t)-Y_2(t))}{(Y_3(t)-Y_1(t))(Y_3(t)-Y_2(t))}.\frac{Y_3(t)}{3} \quad (12)$$

$F(t)$, and $\alpha(t)$denote male jackal, female jackal, and alpha wolf positions at iteration t; $Y(t) \to Y(t+1)$is the population update, with constant 3 ensuring equal weighting and convergence control.

3.4 Classification using proposed transfer learning

This section describes the system architecture and the proposed DTL model for IoT attack detection.

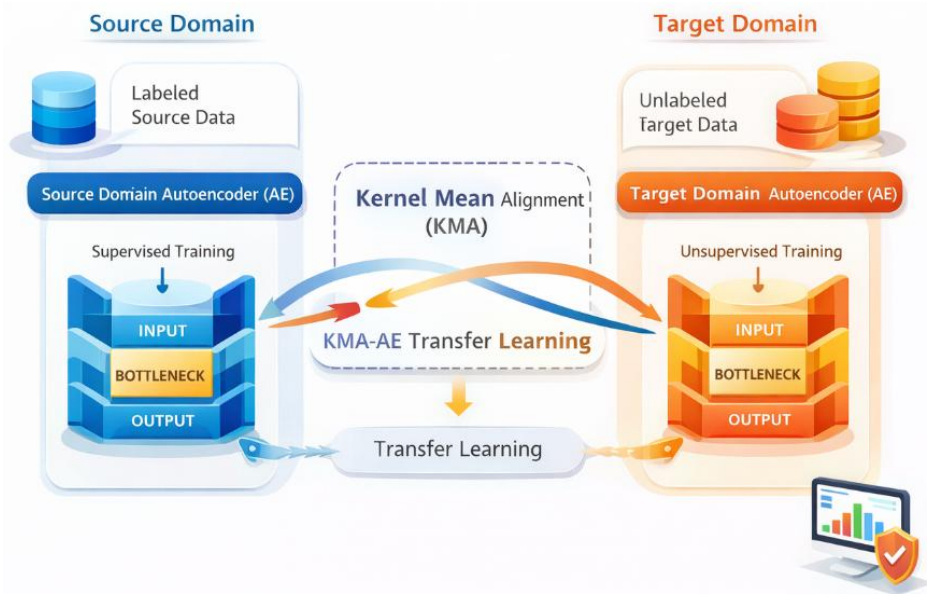

Figure 3 KMA-AE transfer learning architecture.

3.4.1 System structure

IoT traffic is collected as labelled and unlabeled data. A DTL model transfers knowledge from labelled samples to unlabeled data by aligning latent representations, enabling accurate intrusion detection.

3.4.2 Transfer learning model

KMA-AE employs dual autoencoders with reconstruction, supervised, and discrepancy losses to learn transferable latent representations for intrusion detection.

$$\ell_{RE}\left(x_S^i, \varphi_S, \theta_S, x_T^i, \varphi_T, \theta_T\right) = l\left(x_S^i, \hat{x}_S^i\right) + l\left(x_T^i, \hat{x}_T^i\right) \tag{13}$$

where l MSE function, $x_S^i, \hat{x}_S^i$, $x_T^i, \hat{x}_T^i$ are data tasters of layers besides the domain's output layers, correspondingly.
The supervised loss embeds class-discriminative information into the bottleneck layer by aligning latent neurons with source-domain labels.

$$\ell_{SE}\left(x_S^i, y_S^i, \varphi_S, \theta_S\right) = \sum_{j=1}^{C} y^i y^j \log\left(z^i S^j\right) \tag{14}$$

The KMA loss aligns source and target latent representations across all encoding layers, reducing domain discrepancy. Training minimises the combined reconstruction, supervised, and alignment losses, while inference classifies target samples using the trained model.

$$\ell_{MMD}\left(x_S^i, \varphi_S, \theta_S, x_T^i, \varphi_T, \theta_T\right) = \sum_{k=1}^{K} MMD\left(\xi_S^k\left(x_S^i\right), \xi_T^k\left(x_T^i\right)\right) \tag{15}$$

$$\ell = \ell_{SE} + \ell_{RE} + \ell_{MMD} \tag{16}$$

Algorithm 1 Training the Projected DL Classical
$INPUT$: x_S, y_S: $Training\ data\ samples\ and\ corresponding\ labels\ in\ the\ source\ dom$ain, x_T: Training data target area
OUTPUT: Trained replicas:AE_2.
BEGIN:
1. Put x_S to the input of AE_1
2. Put x_T to the input of AE_2
3. $\xi_k(x_S)$ is a picture of x_S at layer k of AE_1
4. z_S is an illustration of x_S at the layer of AE_1
5. ξ_k (x_T) is the illustration of x_T at layer k of AE_2
6. $Training\ the\ TL\ model\ by\ minimizing\ the\ loss\ function\ in$ (16) $return\ Trained$ models: AE_1, AE_2.
END.

KMA-AE transfers knowledge across all encoder layers, improving domain alignment beyond bottleneck-only methods.

While training incurs additional overhead, inference remains efficient with no added prediction cost.

Algorithm 2: Categorising on Target Area
INPUT: x_T: Testing statistics examples in the field Trained AE_2 classical
OUTPUT: y_T: Label of x_T
BEGIN:
1. Put x_T to the input of AE_2
2. z_T is the illustration of x_T layer of AE_2
3. $y_T = softmax\ (z_T)$
return y_T
END.

3.4.3 Hyper-parameters setting

All AE-based models share a common setup with a five-layer architecture, Adam optimization, ReLU activations, and sigmoid output. Early stopping with 10% validation data halts training when AUC degrades.

4 Results and Discussion

Experiments were conducted in Google Colab using Python, with training performed on an NVIDIA Quadro P4000 (8 GB). Model evaluation employed train–test splits and tuned hyperparameters, including epochs, learning rate, dropout, and batch size.

4.1 Validation analysis of projected feature selection is perfect with existing techniques

Table 4 and Figure 4 present the proposed optimizer's existing events across diverse metrics.

Table 4 Validation Analysis of the proposed Feature Selection optimizer.

Algorithms	Sensitivity (%)	Specificity (%)	Accuracy (%)
PSO	78.21	83.00	77.00
Wild Goose	79.54	84.00	79.00
Sea Horse	80.41	85.00	80.00
Butterfly	82.46	87.00	81.00
GWO	84.04	89.00	82.00
GJO	86.41	90.00	84.00
GJO-GWO	88.24	92.00	85.34

The proposed GJO–GWO optimizer delivers superior performance, achieving 88.24% sensitivity, 92.00% specificity, and 85.34% accuracy, outperforming all competing optimizers. Standalone GJO and GWO show strong but inferior results, while PSO performs the weakest, confirming the effectiveness of the hybrid optimization strategy.

4.2 Validation study of the projected classifier with existing techniques

Table 5 and Figure 5 present a visual comparison of the proposed classifier and existing techniques across various metrics.

Table 5 Analysis of proposed classifier, existing models.

Algorithm	Sensitivity (%)	Specificity (%)	Accuracy (%)
XGBoost	81.0	97.0	91.2
ELM	80.1	95.4	91.6
SVM	67.2	97.2	90.1
AE	82.5	96.8	91.8
DBN	87.7	94.0	97.3
KMA-AE	92.24	98.1	90.21

KMA-AE achieves the highest specificity (98.1%) while maintaining strong sensitivity (92.24%) and accuracy (90.21%), outperforming competing models in balanced detection performance and demonstrating robustness for real-world deployment.

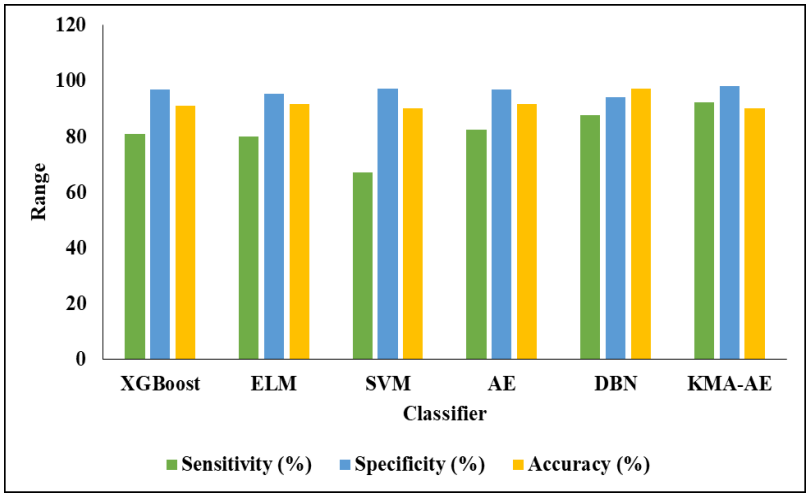

Figure 4 Visual Analysis of the proposed classifier for binary classes.

Table 6 provides the execution time study of the proposed classifier with existing techniques. KMA-AE achieves the lowest execution time (101.87 s), outperforming all baseline models and confirming its computational efficiency for real-time deployment.

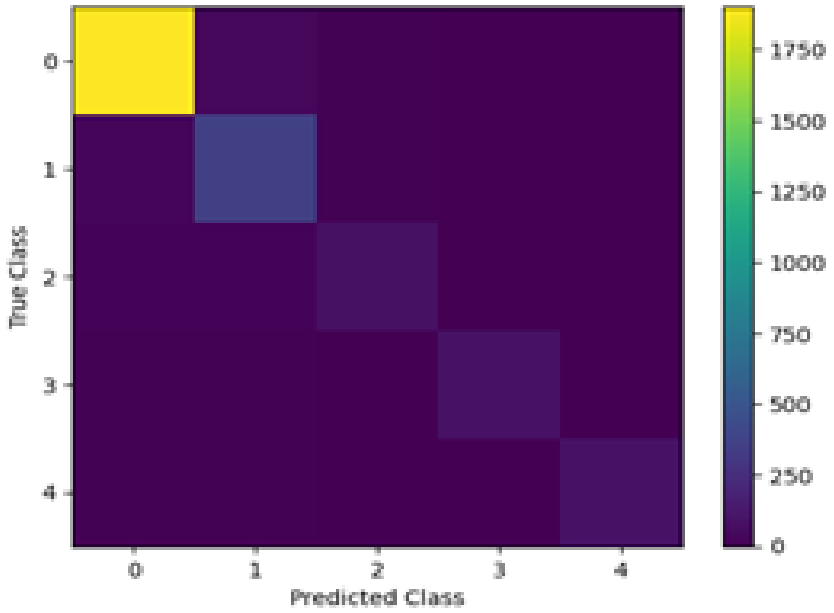

Figure 5 Multi-class confusion matrix.

Table 6 Execution Time Analysis.

Algorithm	Execution Time(s)
XGBoost	151.23
ELM	128.51
AE	126.98
DBN	105.92
KMA-AE	101.87

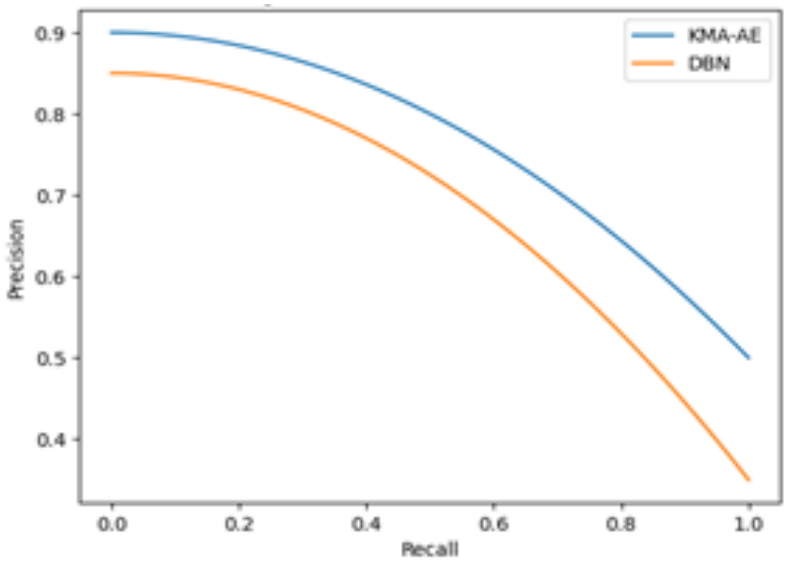

Figure 6 Precision–Recall Curve.

Table 7 Per-Class Performance Metrics.

Class	Precision	Recall	F1-score
No Attack	0.97	0.95	0.96
DoS	0.91	0.93	0.92
Brute Force	0.88	0.85	0.86
Ping Scan	0.89	0.87	0.88
Port Scan	0.90	0.86	0.88

Figures 7–8 and Table 7 show that KMA-AE achieves strong class separation, superior minority-class detection, and consistently high per-class precision, recall, and F1-scores under imbalanced conditions.

Table 8 Macro vs Micro Averaged.

Model	Micro-F1	Macro-F1	Weighted-F1
DBN	0.97	0.81	0.95
KMA-AE	0.90	0.90	0.91

Table 9 Feature reduction effectiveness of different optimizers**.**

Optimiser	Total Features	Selected Features	Reduction (%)
PSO	10	7	30
GWO	10	6	40
GJO	10	5	50
GJO-GWO	10	4	60

Tables 8-11 show that KMA-AE achieves balanced F1 performance, efficient feature reduction via GJO-GWO, improved detection through multi-layer KMA alignment, and fast real-time inference despite higher training cost.

Table 10 Ablation study of the proposed framework.

Configuration	Accuracy (%)	Macro-F1
AE only	88.20	0.78
AE + KMA (bottleneck)	89.50	0.83
AE + KMA (all layers)	90.10	0.88
KMA-AE + GJO	89.90	0.87
KMA-AE + GJO-GWO	90.21	0.90

Table 11 Training vs testing time analysis.

Model	Training Time (s)	Testing Time (s)
XGBoost	120.4	30.8
DBN	98.7	7.2
KMA-AE	145.3	5.1

4.3 Threats to Validity and Reproducibility

Threats to Validity. Limited temporal scope, dataset representativeness, and class imbalance may affect generalisation. Reproducibility was ensured through fixed splits, consistent preprocessing, and identical hyperparameters across all models.

5 Conclusion

This paper proposes KMA-AE, a deep transfer learning framework for IoT intrusion detection under limited labelled data. By aligning source and target representations across all autoencoder layers using Kernel Mean Alignment and optimising features via a hybrid GJO–GWO algorithm, the model achieves improved detection performance and AUC. While training overhead increases, inference remains efficient. Future work will explore federated and distributed learning for scalable IoT deployment.

References

[1] Shahin, M., Maghanaki, M., Hosseinzadeh, A. & Chen, F.F., *Advancing Network Security in Industrial IoT: A Deep Dive Into AI-Enabled Intrusion Detection Systems*, Advanced Engineering Informatics, **62**, 102685, 2024.

[2] Srivastava, D., Singh, R., Chakraborty, C., Maakar, S.K., Makkar, A. & Sinwar, D., *A Framework for Detection of Cyber-Attacks by the*

Classification of Intrusion Detection Datasets, Microprocessors and Microsystems, **105**, 104964, 2024.

[3] Isong, B., Kgote, O. & Abu-Mahfouz, A., *Insights into Modern Intrusion Detection Strategies for Internet of Things Ecosystems*, Electronics, **13**(12), 2370, 2024.

[4] Wang, Z., Li, J., Yang, S., Luo, X., Li, D. & Mahmoodi, S., *A Lightweight Iot Intrusion Detection Model Based on Improved BERT-of-Theseus*, Expert Systems with Applications, **238**, 122045, 2024.

[5] More, S., Idrissi, M., Mahmoud, H. & Asyhari, A. T., *Enhanced Intrusion Detection Systems Performance with UNSW-NB15 Data Analysis*, Algorithms, **17**(2), 64, 2024.

[6] Li, S., Cao, Y., Liu, S., Lai, Y., Zhu, Y. & Ahmad, N., *HDA-IDS: A Hybrid Dos Attacks Intrusion Detection System for IoT Using Semi-Supervised CL-GAN*, Expert Systems with Applications, **238**, 122198, 2024.

[7] Amoo, O.O., Osasona, F., Atadoga, A., Ayinla, B.S., Farayola, O.A. & Abrahams, T.O., *Cybersecurity Threats in the Age of IoT: A Review of Protective Measures*, International Journal of Science and Research Archive, **11**(1), pp.1304-1310, 2024.

[8] Hajla, S., Ennaji, E., Maleh, Y. & Mounir, S., *Enhancing IoT Network Defense: Advanced Intrusion Detection via Ensemble Learning Techniques*, Indonesian Journal of Electrical Engineering and Computer Science, **35**(3), pp.2010-2020, 2024.

[9] Inuwa, M.M. & Das, R., *A Comparative Analysis of Various Machine Learning Methods for Anomaly Detection in Cyber-Attacks on IoT Networks*, Internet of Things, **26**, 101162, 2024.

[10] Abdulkareem, S.A., Foh, C.H., Shojafar, M., Carrez, F. & Moessner, K., *Network Intrusion Detection: An IoT and Non-IoT-Related Survey*, IEEE Access, **12**, pp.147167-147191, 2024.

[11] Wang, X., Qiao, Y., Xiong, J., Zhao, Z., Zhang, N., Feng, M. & Jiang, C., *Advanced Network Intrusion Detection with Tabtransformer*, Journal of Theory and Practice of Engineering Science, **4**(03), pp.191-198, 2024.

[12] Hamouda, D., Ferrag, M.A., Benhamida, N., Seridi, H. & Ghanem, M. C., *Revolutionizing Intrusion Detection in Industrial IoT with Distributed Learning and Deep Generative Techniques*, Internet of Things, **26**, 101149, 2024.

[13] Nandanwar, H. & Katarya, R., *Deep Learning Enabled Intrusion Detection System for Industrial IoT Environment*, Expert Systems with Applications, **249**, pp.123808, 2024.

[14] Devendiran, R. & Turukmane, A.V., *DUGAT-LSTM: Deep Learning-Based Network Intrusion Detection System Using Chaotic Optimization Strategy*, Expert Systems with Applications, **245**, pp.123027, 2024.

[15] Gueriani, A., Kheddar, H. & Mazari, A.C., *Enhancing IoT Security with CNN And LSTM-Based Intrusion Detection Systems*, Proceedings of the 6th

International Conference on Pattern Analysis and Intelligent Systems (PAIS), IEEE, pp.1-7, 2024.

[16] Jony, A.I. & Arnob, A.K.B., *A Long Short-Term Memory-Based Approach for Detecting Cyber-Attacks in IoT Using CIC-Iot2023 Dataset*, Journal of Edge Computing, **3**(1), pp.28-42, 2024.

[17] Al-Ambusaidi, M., Yinjun, Z., Muhammad, Y. & Yahya, A., ML-IDS: *An Efficient ML-Enabled Intrusion Detection System for Securing IoT Networks and Applications*, Soft Computing, **28**(2), pp.1765-1784, 2024.

[18] Ciric, V., Milosevic, M., Sokolovic, D. & Milentijevic, I., *Modular Deep Learning-Based Network Intrusion Detection Architecture for Real-World Cyber-Attack Simulation*, Simulation Modelling Practice and Theory, **133**, 2024.

[19] Alhayan, F., Alruwais, N., Alamgeer, M., Alashjaee, A. M., Abdullah, M., Khadidos, A.O., Alallah, F.S. & Al Shareef, A., *Design of Advanced Intrusion Detection in Cybersecurity Using Ensemble of Deep Learning Models with Improved Beluga Whale Optimization Algorithm*, Alexandria Engineering Journal, **121**, pp.90-102, 2025.

[20] Yang, H., Yu, J. & Zhai, R., *High-Precision Intrusion Detection for Cybersecurity Communications Based on Multi-Scale Convolutional Neural Networks*, The Journal of Supercomputing, **81**(1), pp.1-34, 2025.

[21] Ghosh, S., *Network Traffic Analysis Based on Cybersecurity Intrusion Detection Through an Automated Separate Guided Attention Federated Graph Neural Network*, Applied Soft Computing, **169**, 112603, 2025.

[22] Zeghida, H., Boulaiche, M., Chikh, R., Bamhdi, A.M., Barros, A.L.B., Zeghida, D. & Patel, A., *Enhancing IoT Cyber-Attacks Intrusion Detection through GAN-Based Data Augmentation and Hybrid Deep Learning Models For MQTT Protocol*, Cluster Computing, **28**(1), 58, 2025.

[23] Ahmed, M.A.O., AbdelSatar, Y. & Alotaibi, R. *Enhancing Internet of Things Security Using Performance Gradient Boosting for Network Intrusion Detection Systems*, Alexandria Engineering Journal, **116**, pp.472-482, 2025.

[24] Mahmoud, M. M., Youssef, Y.O. & Abdel-Hamid, A.A., *XI2S-IDS: An Explainable Intelligent Two-Stage Intrusion Detection System*, Future Internet, **17**(1), pp.25, 2025.

[25] Verma, A. & Ranga, V., *On Evaluation of Network Intrusion Detection Systems: Statistical Analysis of CIDDS-001 Dataset Using Machine Learning Techniques*, Pertanika Journal of Science and Technology, **26**(3), pp.1307-1332, 2018.

[26] Liu, G., Guo, Z., Liu, W., Jiang, F. & Fu, E., *A Feature Selection Method Based on the Golden Jackal–Grey Wolf Hybrid Optimization Algorithm*, PLOS ONE, **19**(1), e0295579, 2024.

[27] Srinivasan, V., Raj, V. H., Thirumalraj, A. & Nagarathinam, K., *Detection of Data Imbalance in MANET Network Based on ADSY-Aeambi-LSTM*

With DBO Feature Selection, Journal of Autonomous Intelligence, **7**(4), 1094, 2024.

[28] Kingma, D.P. & Ba, J., *Adam: A Method for Stochastic Optimization*, arXiv preprint, arXiv:1412.6980, 2014, ICLR 2015.

[29] Stephe, S., Manjunatha, B., Revathi, V. & Thirumalraj, A.*, Osteosarcoma Cancer Detection Using Ghost-Faster RCNN Model from Histopathological Images*, Iran Journal of Computer Science, **8**(1), pp.217-231, 2025.

[30] Aluvalu, R., Sharma, T., Viswanadhula, U.M., Thirumalraj, A.D., Prasad Kantipudi, M V.V. & Mudrakola, S., *Komodo Dragon MLIPIR Algorithm-Based CNN Model for Detection of Illegal Tree Cutting in Smart IoT Forest Area*, Recent Advances in Computer Science and Communications, **17**(6), pp.1-12, 2024.