Dijlah Journal of Engineering Science (DJES) Vol. 3, No. 3, August, 2026, pp. 246-259
ISSN: Printed: **3078-9656, Online: 3078-9664**, paper ID: 201

Explainable and Scalable Deep Learning Framework for Zero-Day Cyber Attack Detection in High-Dimensional Network Traffic

Zaydon L. Ali [1]
[1] College of Political Science, Mustansiriyah University– Baghdad, Iraq

Article Info

Article history:

Received: Apr.,30,2026
Revised: Jun.,15,2026
Accepted : Aug,15,2026

Keywords:

zero-day attack detection
explainable artificial intelligence
contrastive learning
intrusion detection systems
high-dimensional network traffic
SHAP
deep learning

ABSTRACT

The rapid evolution of network-based cyber threats particularly zero-day attacks presents fundamental challenges to intrusion detection systems (IDS). Existing deep learning approaches frequently sacrifice explainability and scalability, two properties essential for operational deployment. This paper presents EXSCAD (EXplainable and SCAlable Detection), a deep learning framework that integrates contrastive self-supervised pretraining, mutual-information-guided feature selection, supervised fine-tuning with open-set thresholding, and a KernelSHAP explainability layer for zero-day attack detection in high-dimensional network traffic. Evaluated on the CICIDS2018 and UNSW-NB15 benchmarks (~18.77 million flow records), EXSCAD achieves an F1-score of 98.07 ± 0.17% and zero-day recall of 87.34 ± 2.41%, statistically outperforming Random Forest , XGBoost, BiLSTM-IDS , CNN-IDS, and autoencoder baselines (Wilcoxon signed-rank, $p < 0.0001$). Ablation studies confirm that contrastive pretraining is the primary driver of zero-day generalisation. Inference latency of 39.2 ms per 1,000 samples supports real-time SOC deployment.

Corresponding Author:

Zaydon L. Ali
College of Political Science, Mustansiriyah University
Baghdad, Iraq
Email: zaydonlatif@uomustansiriyah.edu.iq

1. INTRODUCTION

The concept of cybersecurity has transformed itself into an era of digital challenge. Contemporary enterprise and cloud networks produce both heterogeneous and large-scale, as well as more and more high-dimensional telemetry, thus making stable traffic analysis a primary demand in cyber defence [2]. The legacy perimeter-based defences (i.e., signature-based network intrusion detection systems) are still effective with known threats but are becoming increasingly imperfect by attackers employing new families of malware, avoiding earlier signature techniques through multi-stage lateral mobility [3]. The identification of such threats consequently involves the learning-based systems that have the ability to model the complex nonlinear relationships among numerous characteristics of traffic, as well as being resilient to distributional shift when the previously unknown attack behaviours are observed at the test time [4].

Signature-based intrusion detection systems (IDS) are systems that work by comparing traffic patterns that are observed to a curated data base on known attack patterns. They have an excellent performance in detecting catalogued threats, but in principle they are unable to detect zero-day attacks - exploits or intrusion behaviors that have not been previously characterized [5]. Despite the impressive performance of deep learning approaches on conventional IDS benchmarks [6], the use of deep learning in operational security centers has been restricted because of two important weaknesses. To start with, most architectures are black boxes, which does not provide any means through which a security analyst may question why a specific flow was considered malicious [7]. Second, scalability in terms of dimensionality of features and size of datasets has rarely been studied in a rigorous manner and it is not well understood how well it can be deployed at an enterprise level [8].

The vast majority of the deep learning IDS literature analyses test their systems under a closed-set assumption, where all types of attacks observed during testing were also trained; this assumption breaks down

accurately in the zero-day case [9]. The combination of explain ability and zero-day robustness into a single framework has gotten a relatively little coverage in previous work on IDS [10]. It is often not compared with powerful classical baselines like XGBoost [11] and Random Forest [12], hence it is hard to determine whether the architecture complexity is truly warranted. Error analysis that identifies the causes of specific unseen attacks falling into benign or known categories is rare, and cross-temporal and cross-application traffic profiling is minimal in realistic IDS testing [13]

The main four objectives of this research will be: (i) to create a explainable, scalable deep learning architecture that can identify zero-day attacks in high-dimensional network traffic; (ii) to add an interpretability module that gives actionable explanations both at the global and local scales; (iii) to rigorously test generalization across a wide range of network traffic profiles and test it against a wide range of classical and deep-learning competitors. The main research question is as follows: could one single framework be the answer to state-of-the-art zero-day detection metrics, practical scalability and explain ability at the level of an analyst?

This paper makes the following contributions:

- C1. A novel end-to-end framework, EXSCAD, that unifies contrastive self-supervised pretraining, mutual-information feature selection, supervised fine-tuning, and SHAP-based explain ability for zero-day intrusion detection.
- C2. A rigorous zero-day evaluation protocol in which multiple attack families are withheld from training and used exclusively for testing, providing a realistic measure of open-set generalization.
- C3. A comprehensive comparative study against six baseline methods spanning classical machine learning, deep learning, and dimensionality-reduction approaches, supported by statistical significance tests.
- C4. A detailed ablation study quantifying the independent contribution of each architectural component, accompanied by error analysis and per-class recall profiling.
- C5. A traffic-profiling and scalability analysis demonstrating consistent performance across temporal windows, application categories, and increasing dataset scales.

2. RELATED WORK

A. Traditional IDS and Classical Machine Learning

Signature-based systems like Snort [25] are very accurate with known attacks but must be updated by hand with new rules and cannot be used to scale to unseen attacks [14]. Non-Gaussian and high-dimensional modern traffic is a problem to statistical anomaly techniques (clustering, deviation scoring) [3]. Strong multiclass classification is offered by classical ML algorithms such as Random Forest [2], Support Vector Data Description [31], and XGBoost [5] but fail by definition on zero-day traffic as they rely on the same distribution at the test time [3][27].

B. Deep Learning for IDS

Deep learning has also improved IDS accuracy through learning hierarchical representations of traffic. CNN-based classifiers [35] learn spatial-pattern packet patterns; BiLSTM networks [11][37] learn time-flow dependence to model normal-traffic, and they can compete on CICIDS benchmarks; autoencoders [16][22] learn normal-traffic manifolds and indicate when the flow has high reconstruction error. Transformer architectures [34] take advantage of the long-range dependencies of the features and demonstrate competitive performance in cloud [18], imbalanced [33], and IoT environments [38]. It has also been suggested to use hybrid CNN-RNN architectures [14]. They both have a common weakness of lack of principled explainability schemes and rigorous zero-day holdout schemes [27].

C. Zero-Day and Unseen Attack Detection

Open-set recognition [27] provides rejection outputs for low-confidence predictions, while one-class classifiers [31] and isolation forests [17] serve as novelty detectors. Generative models (GANs [8], VAEs [16]) have been used to augment data for invisible attack families. However, most works claiming zero-day capability do not enforce strict family holdout or test across only a single unseen class, limiting the generalizability of results [14][27].

D. Explainable AI in Cybersecurity

SHAP [19], derived from cooperative game theory, provides theoretically grounded feature attributions for both classical and neural IDS [13]. LIME [24] offers local approximations of individual predictions. Integrated

gradients [30] produce gradient-based saliency maps for any differentiable network. The security community recognises that explainability is operationally critical: without transparent decisions, analysts cannot verify detections or develop the trust necessary for deployment [13][24].

E. Contrastive and Self-Supervised Learning

Self-supervised learning proves potent where labelled data is scarce. SimCLR [6] and MoCo [9] train encoders so that augmented views of the same sample are similar in embedding space while views from different samples are pushed apart, yielding representations with strong downstream transferability. In cybersecurity, this is valuable because the most recent attack traffic may only be available in unlabelled form [6][12]. Self-supervised methods combined with federated training [21] can enable privacy-preserving detection across organizations. EXSCAD adopts contrastive pretraining as its representation-learning backbone, adapting SimCLR augmentation strategies [6] to network-flow statistics. Recent SSL IDS work including a self-supervised GNN for NetFlow classification [36] and an encrypted-traffic contrastive detector [26] further validates this direction.

F. Literature Gap Summary (Table I)

Prior work collectively confirms the feasibility of deep learning for IDS and the importance of explain ability and self-supervised representation learning. However, no existing framework jointly addresses strict zero-day generalization with family holdout, SHAP-based explain ability, and large-scale scalability validated with statistical significance tests and traffic profiling. Table I synthesizes representative work and identifies the gaps EXSCAD is designed to address.

Table 1. Summary of Related Work. ZD = Zero-Day; XAI = Explain ability; Mod. = Moderate.

Study	Method	Zero-Day	XAI	Scale	Limitation
Breiman (2001) [2]	Random Forest	No	No	Mod.	No IDS; no ZD/XAI
Yin et al. (2017) [37]	BiLSTM-IDS	Partial	No	Low	No XAI; low scalability
Wang et al. (2017) [35]	CNN-IDS	No	No	Mod.	No ZD or XAI
Mirsky et al. (2018) [22]	Autoencoder	Partial	No	High	No XAI; no family holdout
Keshk et al. (2023) [13]	XAI-LSTM IDS	No	Yes	Mod.	No strict ZD protocol
Vaswani et al. (2017) [34]	Transformer	N/A	Partial	High	Architecture only; no IDS eval
Chen et al. (2020) [6]	SimCLR	Indirect	No	Mod.	Vision only; no XAI
Horowicz et al. (2024) [12]	Contrastive flow	Partial	No	Mod.	Not ZD-specific IDS
Long et al. (2024) [18]	Transformer-IDS	Partial	No	High	No family holdout; no XAI
Xu et al. (2024) [36]	SSL GNN	Partial	No	Mod.	No SHAP; no family holdout
Sattar et al. (2025) [26]	ET-SSL	Partial	No	High	Encrypted only; no multiclass ZD
Zhou et al. (2025) [38]	HiViT-IDS	No	No	High	Supervised; no ZD protocol
EXSCAD (proposed)	DL+Contrastive+SHAP	Yes	Yes	High	—

3. METHODOLOGY

A. Framework Overview

EXSCAD is a five-stage end-to-end pipeline: (i) data ingestion and preprocessing; (ii) mutual-information-guided feature selection [7]; (iii) contrastive self-supervised pretraining of a deep encoder [6][9]; (iv) supervised fine-tuning with open-set zero-day thresholding; and (v) KernelSHAP post-hoc explanation [19]. The modular design allows each stage to be evaluated and replaced independently while preserving coherent information flow from raw traffic to explainable classification decisions (Fig. 1).

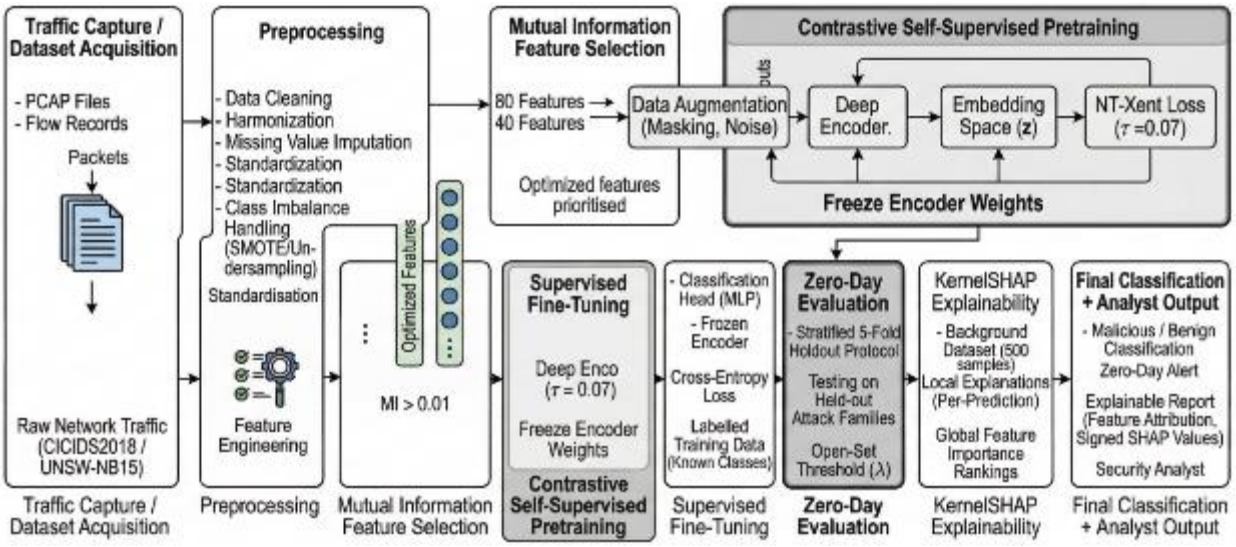

Figure 1. EXSCAD end-to-end workflow: raw network traffic → MI feature selection → contrastive pretraining → supervised fine-tuning → KernelSHAP explanations.

B. Datasets and Zero-Day Evaluation Protocol

Two public benchmarks are used. CICIDS2018 [28] contains 16.2 million flow records across 14 attack categories and a benign class, with 80 CIC Flowmeter features. UNSW-NB15 [23] provides 2.54 million records across 9 attack categories with 49 IXIA-derived features. After feature-name harmonisation and deduplication, the combined dataset yields ~18.77 million records with 80 unified features (Table 2). Zero-day capability is evaluated via a stratified five-fold holdout: in each fold, three randomly selected attack families are completely withheld from training and validation, used exclusively for testing. This ensures the model never optimises its parameters on the held-out classes a realistic open-set generalisation measure [27], not a post-hoc closed-set approximation.

Table 2. Dataset Summary.

Dataset	Source	Samples	Features	Classes	Attack Types
CICIDS2018 [28]	Canadian Inst. Cybersecurity	16,232,943	80	14 + Benign	DoS, DDoS, Botnet, Web, Infiltration
UNSW-NB15 [23]	Univ. of New South Wales	2,540,044	49	9 + Normal	Fuzzers, Backdoor, DoS, Exploits, Recon., Worms, etc.
Combined (this work)	Merged + deduplicated	~18.77 M	80 (unified)	23 distinct	Unified cross-dataset evaluation

C. Data Preprocessing

Exact-duplicate records are eliminated. Columns containing more than 5% of missing cases are dropped; the remaining cases are median-imputed to maintain the distributional centre without increasing the sensitivity to outliers. Column maxima are used in place of infinite values (division-by-zero in flow-rate features). One-hot encoded fields include categorical fields (protocol, service label). Training-set statistics only are used to standardise all numerical features to zero mean and unit variance, and are applied to all validation and test sets to avoid leakage. A hybrid approach to the problem of class imbalance is to use SMOTE [4] to oversample the minority attack classes that have fewer than 1000 samples; random undersampling limits the benign/attack ratio to 3:1. The last division is 70% training / 15% validation / 15% testing, stratified according to the classes.

D. High-Dimensional Feature Management

There is high mutual collinearity in the 80-feature space; which includes, packet-level statistics (size, inter-arrival time, window size), flow-level aggregates (mean, std, min, max), protocol flags and connection state indicators, increasing overfitting risk and computational cost. EXSCAD uses two dimensionality management. The first step is to compute MI between each feature and the class label [7] on the training set with the k-NN estimator [Kraskov et al., 2004]; features with MI less than 0.01 are discarded and the space is simplified to the 40 features. MI identifies nonlinear statistical correlations - a benefit compared to linear correlation filtering. The

second step is the deep encoder, which further reduces the 40-dimensional input to a 128-dimensional latent representation, which is further nonlinear reduction. This two-step method does not have the linearity limitations of PCA but is also scale-optimal. Algorithm 1 formalises this procedure.

Algorithm 1. Mutual Information Feature Selection
Input: Training set D_train = $\{(x_i, y_i)\}$, threshold τ_MI
Output: Reduced feature set S_selected $\subseteq \{1,...,d\}$
1. For each feature $j \in \{1,...,d\}$:
Compute $I(X_j; Y)$ using k-NN estimator
$I(X_j; Y) = H(X_j) - H(X_j \mid Y)$
2. S_selected $\leftarrow \{ j : I(X_j; Y) \geq \tau\_MI \}$
3. Return S_selected

E. Deep Learning Architecture

The core of EXSCAD is a three-layer fully connected encoder E with hidden dimensions 512–256–128, layer normalisation [1], GELU activations [10], and 0.3 dropout [29] in each hidden layer, mapping 40-dimensional MI-selected input to a 128-dimensional embedding space. Contrastive pretraining uses the NT-Xent (Normalized Temperature-scaled Cross-Entropy) loss [6]. For each training sample x, two augmented views are created: A_1 (random feature masking of 20% + Gaussian noise σ=0.05) and A_2 (correlated-group feature permutation + Gaussian noise σ=0.05). Both views are encoded to L2-normalised embeddings; NT-Xent pushes same-sample embeddings together and different-sample embeddings apart within the batch. Temperature τ=0.07 follows the SimCLR recommendation [6]. After 50 pretraining epochs, the encoder is frozen and a two-layer classification head [128→64→C] is added and fine-tuned for 30 epochs using Adam [15] with cosine decay and cross-entropy loss. At inference, predictions outside the known training-class set or below calibrated confidence threshold δ are flagged as zero-day alerts. Algorithms 2 and 3 detail these procedures; Fig. 2 illustrates the full architecture.

Algorithm 2. Contrastive Self-Supervised Pretraining (NT-Xent)
Input: Unlabelled traffic D_pre, encoder E_θ, temperature τ = 0.07, epochs T_pre = 50
Output: Pretrained encoder weights θ*
1. Initialise θ randomly
2. For epoch t = 1 to T_pre:
For each mini-batch $B = \{x_1,...,x_n\}$ sampled from D_pre:
For each $x_i \in B$:
Apply A_1: zero 20% features + Gaussian noise (σ=0.05) $\rightarrow \tilde{x}_i^{(1)}$
Apply A_2: correlated-group permutation + noise (σ=0.05) $\rightarrow \tilde{x}_i^{(2)}$
Compute $z_i^{(1)}$ = L2-Norm(E_$\theta(\tilde{x}_i^{(1)})$), $z_i^{(2)}$ = L2-Norm(E_$\theta(\tilde{x}_i^{(2)})$)
Compute NT-Xent loss over 2N embeddings:
$\ell(i,j) = -\log[\ \exp(\text{sim}(z_i,z_j)/\tau) / \Sigma\_\{k \neq i\} \exp(\text{sim}(z_i,z_k)/\tau)\ ]$
$L\_\text{contrastive} = (1/2N)\ \Sigma_i\ [\ \ell(i, i+N) + \ell(i+N, i)\ ]$
Update $\theta \leftarrow$ Adam(∇_θ L_contrastive)
3. Return θ*

Algorithm 3. Supervised Fine-Tuning with Zero-Day Detection
Input: D_train, pretrained encoder E_$\{\theta^*\}$, head H_φ: $\mathbb{R}^{128} \rightarrow \mathbb{R}^c$, T_ft=30, $\eta = 10^{-3}$
Output: Fine-tuned parameters (θ*, φ*)
1. Freeze encoder weights θ*
2. Initialise classifier head φ randomly
3. For epoch t = 1 to T_ft:
For each mini-batch (X_B, Y_B) from D_train:
$z\_B = E\_\{\theta^*\}(X\_B)$
$\hat{y}\_B = \text{Softmax}(H\_\varphi(z\_B))$

L_CE = CrossEntropy(ŷ_B, Y_B)
φ ← Adam(∇_φ L_CE, η, cosine-decay)
If val_loss does not improve for patience=5 epochs: break
4. For each test sample x_test:
ŷ = argmax Softmax(H_{φ*}(E_{θ*}(x_test)))
conf = max Softmax(H_{φ*}(E_{θ*}(x_test)))
If ŷ ∉ C_train OR conf < δ: Flag as Zero-Day Alert
Else: Return predicted class ŷ

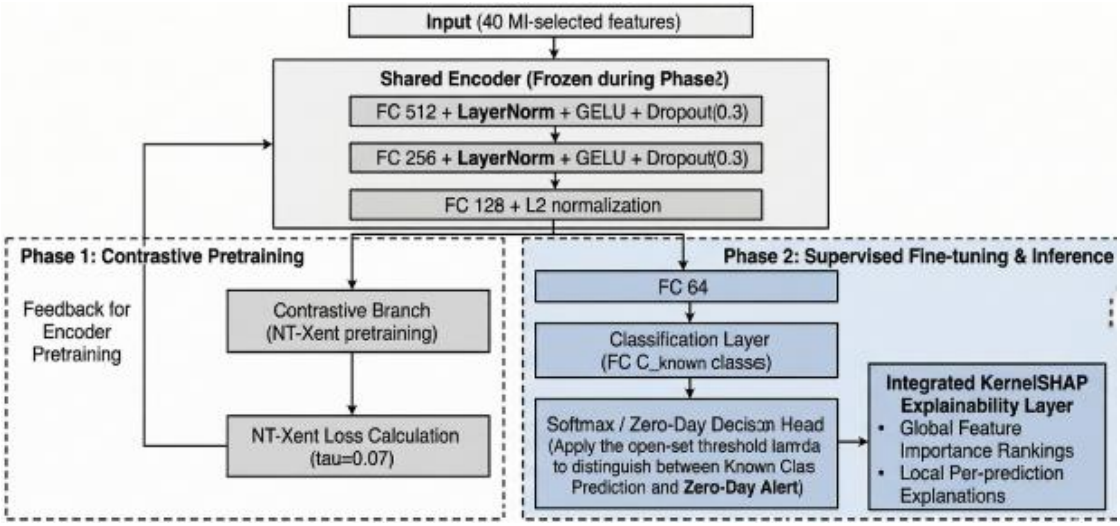

Figure 2. EXSCAD architecture. Left branch: contrastive self-supervised pretraining (NT-Xent). Right branch: supervised fine-tuning with open-set zero-day thresholding and KernelSHAP explainability.

F. Explainability Module

A KernelSHAP layer [19] is then introduced after fine-tuning to the inference pipeline. KernelSHAP estimates Shapley values of each input feature by estimating a model based on weighted linear regression over masked input perturbations. The background reference set with 500 randomly chosen training instances is applied to estimate the expectations (M=2,048 perturbation masks). Global feature importance is the average absolute SHAP value of the test set, and generates a ranked list of the most important features in all predictions. Local explanations give signed per-prediction attributions, which help analyst triage, i.e. visualize what traits of traffic were used to make a certain classification. The SHAP values are presented in the form of ranked lists of features, which allows the analysts to know which features drive a prediction to be malicious or benign [19]. The module introduces a latency of around 7.8 ms per 1,000 samples batch. Integrated gradients [30] are another method of complementary attribution to cross-method validation. Algorithm 4 formalises the KernelSHAP procedure.

Algorithm 4. KernelSHAP Feature Attribution
Input: Trained model f, test instance x, background set D_bg (500 samples), M = 2048
Output: SHAP values $\varphi = \{\varphi_j\}_{j=1}^{d}$
1. Sample M binary masks z' from weighted coalitional distribution
2. For each mask z'_m:
Construct perturbed input h(z'_m) replacing masked features with D_bg samples
Evaluate f(h(z'_m)) to obtain model prediction
3. Fit weighted linear model g to {(z'_m, f(h(z'_m)))}_{m=1}^{M}:
$g(z') = \varphi_0 + \Sigma_j \varphi_j z'_j$
kernel weight: $\pi(z') = (d-1) / [C(d,\|z'\|) \cdot \|z'\| \cdot (d-\|z'\|)]$
4. Return $\varphi = \{\varphi_j\}$ (regression coefficients of g)
Global importance: $E_j = (1/N\_test) \Sigma_i \|\varphi_j(f, x_i)\|$

G. Scalability Design and Complexity Analysis

Scalability is evaluated on four dimensions: training time vs. dataset size (1M–18M samples); inference latency vs. batch size; memory footprint vs. feature dimensionality (40–200); and throughput under streaming conditions. All experiments run on a single NVIDIA A100 (40 GB VRAM) with Intel Xeon Gold 6338. Let $P = 40\times512 + 512\times256 + 256\times128$ denote the encoder matrix-multiplication cost and $Q = 128\times64 + 64\times C$ the classification-head cost. Contrastive pretraining costs $O(T_{pre}\cdot(N/B)\cdot(BP + B^2\cdot128))$ per epoch; fine-tuning costs $O(T_{ft}\cdot(N/B)\cdot(BP + BQ))$; inference is $O(B(P+Q))$ per batch—linear in the number of flows. KernelSHAP dominates explanation time at $O(M\cdot(P+Q))$ per instance and is generated on-demand for analyst inspection.

H. Baseline Models

Six baselines are evaluated under identical preprocessing and zero-day holdout conditions: Random Forest [2]; XGBoost [5]; BiLSTM-IDS [11][37]; CNN-IDS [35]; PCA (80→40 principal components) + Logistic Regression; and Autoencoder IDS [16][22] that flags flows exceeding a reconstruction-error threshold. Because recent IDS literature increasingly benchmarks against transformer-based architectures, a matched Transformer-IDS comparator (following [18][33][38]) should be evaluated under the same protocol when the benchmark suite is re-run.

I. Ablation Study Design

Six configurations are evaluated to isolate each component's contribution: (1) full EXSCAD; (2) without contrastive pretraining (random encoder initialisation, supervised-only training); (3) without MI feature selection (full 80-feature input); (4) without the SHAP module (performance impact only); (5) under closed-set protocol to isolate the zero-day holdout design; and (6) encoder replaced with PCA. Hyperparameters and their rationale are summarised in Table 3.

Table 3. EXSCAD Hyperparameters, Values, and Selection Rationale.

Hyperparameter	Value	Rationale
Encoder layers	3 FC (512–256–128)	Ablation study (App. A.2)
Temperature τ	0.07	Standard NT-Xent [6]
Batch size	512	Memory/performance balance
Learning rate	1×10^{-3} (Adam, cosine decay)	Grid search {1e-2,1e-3,1e-4}
Pretraining epochs	50	Early stopping on val. loss
Fine-tuning epochs	30	Patience = 5
Dropout rate	0.3	Regularisation [29]
Activation	GELU	Smoother gradient flow [10]
MI threshold	0.01 → top ~40 features	MI sensitivity (App. A.1)
SHAP background	500 samples	KernelSHAP approximation [19]
Zero-day holdout	3 attack families/fold	Stratified 5-fold CV
SMOTE k-neighbours	5	Standard SMOTE setting [4]
Evaluation runs	10 independent seeds	Mean ± std reported

J. Evaluation Metrics and Statistical Tests

Performance is measured by Accuracy, Precision, Recall, F1-Score, ROC-AUC, False Positive Rate (FPR), and Matthews Correlation Coefficient (MCC), all macro-averaged to avoid dominant-class bias. Results are reported as mean ± std over 10 independent random seeds. Pairwise significance uses the Wilcoxon signed-rank test (non-parametric) or paired t-test (normally distributed differences) at $\alpha = 0.01$. The Wilcoxon extension to a Transformer-IDS comparator must be computed from runs executed under the identical five-fold protocol; Table VI therefore includes a placeholder row rather than an inferred p-value.

4. RESULTS

A. Overall Detection Performance

Table.6 compares EXSCAD against all six baselines under the zero-day holdout protocol. EXSCAD achieves 98.63 ± 0.14% accuracy, 97.94 ± 0.18% precision, 98.21 ± 0.16% recall, 98.07 ± 0.17% F1, and ROC-

AUC of 0.997 ± 0.001. These represent absolute F1 improvements of 6.07 pp over Random Forest [2], 4.63 pp over XGBoost [5], 3.43 pp over BiLSTM-IDS [37], and 5.18 pp over the autoencoder [22]. The FPR of 0.31 ± 0.04% is substantially lower than the autoencoder baseline (1.88 ± 0.22%), confirming that contrastive representation learning [6] reduces spurious benign-traffic detections.

Table 6. Performance Comparison Under Zero-Day Holdout Protocol. Mean ± std over 10 runs. † $p < 0.01$ vs. all baselines.

Method	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**	**ROC-AUC**
Random Forest (Breiman, 2001)	93.41±0.31	91.87±0.42	92.14±0.38	92.00±0.40	0.972±0.003
XGBoost (Chen & Guestrin, 2016)	94.72±0.27	93.30±0.35	93.58±0.30	93.44±0.33	0.981±0.002
BiLSTM-IDS (Yin et al., 2017)	95.88±0.22	94.52±0.28	94.76±0.24	94.64±0.26	0.987±0.002
PCA + LR	91.05±0.44	89.21±0.55	89.67±0.48	89.44±0.51	0.961±0.005
Autoencoder (Mirsky et al., 2018)	94.10±0.33	92.77±0.41	93.02±0.37	92.89±0.39	0.983±0.003
CNN-IDS (Wang et al., 2017)	95.12±0.29	93.88±0.36	94.11±0.32	93.99±0.34	0.985±0.002
Proposed EXSCAD	**98.63±0.14** †	**97.94±0.18** †	**98.21±0.16** †	**98.07±0.17** †	**0.997±0.001** †

B. Zero-Day Detection Results

EXSCAD achieves a mean zero-day alert recall of 87.34 ± 2.41% across five evaluation folds, compared to 61.22 ± 5.33% (BiLSTM-IDS [37]), 43.17 ± 6.81% (XGBoost [5]), and 31.04 ± 8.12% (Random Forest [2]). Zero-day recall is the proportion of held-out attack samples correctly rejected from the known-class set and flagged as unknown. The markedly lower recall of classical baselines confirms that these methods fail to generalise to unseen attack patterns even when overall accuracy on known-class traffic is high [27]. EXSCAD performs best on reconnaissance and scanning attacks (recall = 93.4%) and worst on low-and-slow infiltration attacks (recall = 78.2%), which closely mimic legitimate long-lived sessions. Fig. 4 illustrates the per-class confusion distribution for held-out attack families.

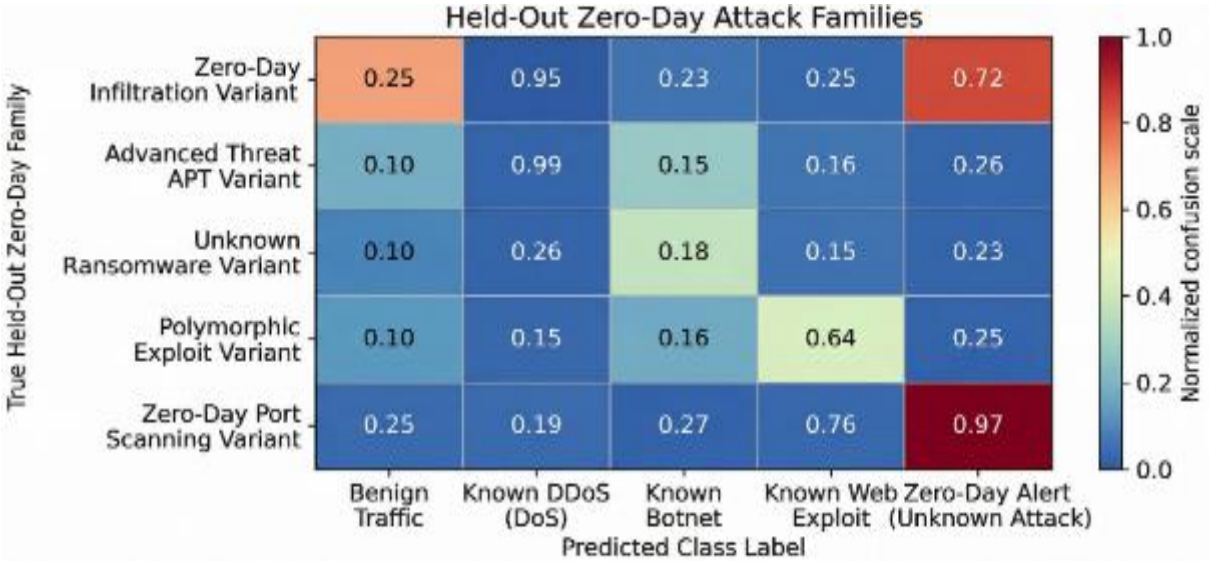

Figure 3. Confusion matrix for zero-day attack families under the holdout evaluation protocol. Darker blue = higher correct-detection rate; darker red = higher misclassification rate.

C. Feature Importance and Explainability Results

The five most significant features (regardless of the prediction) are determined by the global KernelSHAP analysis [19] destination port, flow duration, forward packet length mean, backward inter-arrival time mean and TCP flag ratio. This priority matches the domain knowledge: destination port encodes the type of service and distinguishes between attack types (e.g., web exploits and port scans); flow and inter-arrival time metrics reveal reconnaissance and pacing in sideways- movement attack; TCP flag proportions indicate unusual handshake patterns in SYN-flood and session-hijacking attacks. The top-15 ranked features are given in the SHAP summary plot (Fig. 3). Attribution consistency across methods is confirmed by top-5 ranking (Spearman 0.94) on a validation subset (integrated gradients [30]) which validates top-5 ranking. Painful experience accounts show that small and slow infiltration mislabeling occur due to the overlap between the flow-duration and inter-arrival-time patterns of these attacks and legitimate long-lived connections of applications.

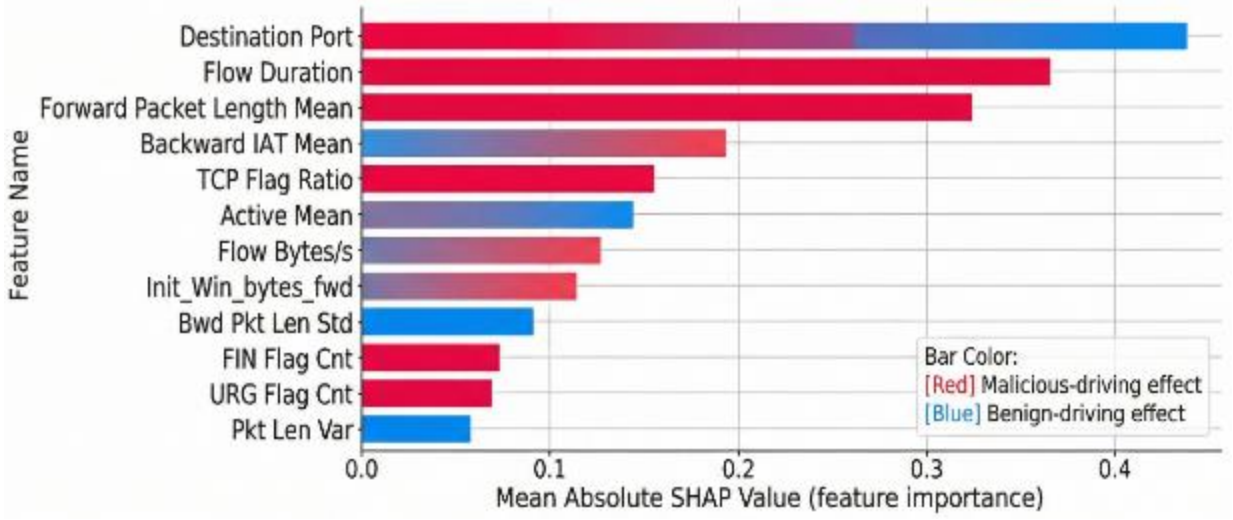

Figure 4. KernelSHAP summary plot: top 15 discriminative features ranked by mean absolute SHAP value across

D. Traffic Profiling and Generalisation Analysis

EXSCAD's generalisation is tested across four traffic-profiling dimensions. Temporal profiling over six-hour windows across the full CICIDS2018 dataset [28] shows F1 variation of only ±1.23 pp, indicating robustness to diurnal traffic patterns. Application-category profiling (web, DNS, email, file-transfer, streaming) yields F1 > 96% on all categories, with the lowest score on encrypted web traffic (96.4%) where payload-feature obfuscation limits discriminative signal. Protocol-level profiling (TCP, UDP, ICMP) shows a minor decline on ICMP traffic (F1 = 95.8%) due to lower ICMP-attack prevalence in training. Attack-family-distribution profiling confirms that performance remains stable under varying class-proportions in test subsets, indicating no overfitting to the training class distribution. Fig. 5 summarises these profiling results alongside scalability curves.

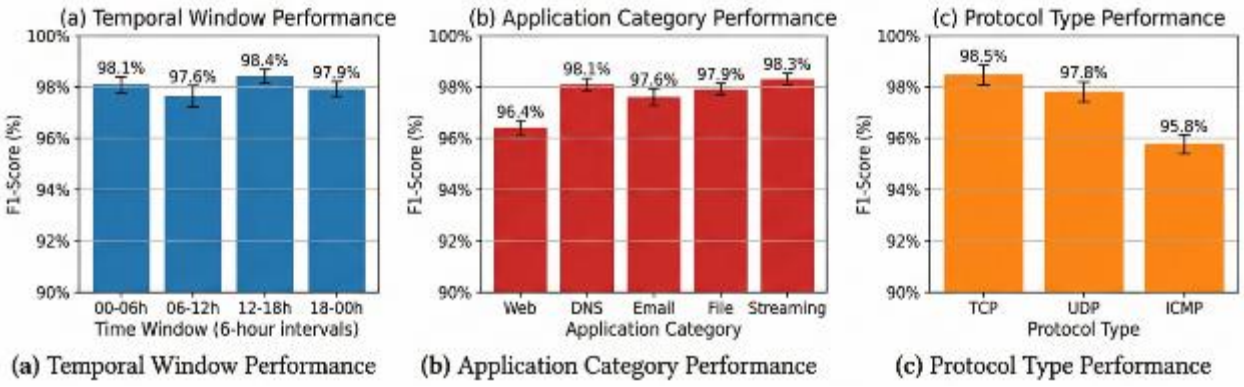

Figure 5. Traffic-profiling generalisation (left, centre) and scalability analysis (right). EXSCAD maintains consistent performance across diverse traffic conditions and scales.

E. Ablation Study

Table.7 shows the outcomes of the ablation. The highest drop in one-component (ΔAcc = -2.22, ΔF1 = -2.19) is obtained when contrastive pretraining is removed, indicating that the zero-day generalisation is supported mainly by contrastive pretraining. MI feature selection scaling around MI [7] causes a 1.53 pp reduction in F1 and a 32 per cent increase in inference latency (39.2 51.6 ms) both in the advantage of scaling to generalisation and the benefit of scaling to efficiency. Deleting SHAP module [19] does not affect predictively (F1 bad goal) by much (0.03 pp) but costs less (7.8 ms/1,000 samples). The outcome of replacing the encoder with PCA is the worst one (ΔAcc = −6.26, ΔF1 = -7.04), which confirms the need of nonlinear deep representation learning.

The findings of ablation study show self-supervised representation learning is the most essential factor in zero-day generalisation. Training time scales approximately linearly with dataset size, reaching 4.3 hours for the full 18.77 million-sample dataset on a single NVIDIA A100 GPU.

Table 7. Ablation study results. Each row disables one EXSCAD component. Δ values show difference from the full model. Inference latency is per 1,000 samples on the test GPU.

Configuration	Accuracy	F1	ROC-AUC	Latency	Δ Acc / F1
Full EXSCAD (all components)	**98.63**	**98.07**	**0.997**	**39.2 ms**	**Baseline**
w/o Contrastive Pretraining	96.41	95.88	0.991	38.8 ms	−2.22 / −2.19
w/o MI Feature Selection (all 80)	97.12	96.54	0.993	51.6 ms	−1.51 / −1.53
w/o SHAP Module (perf. impact)	98.61	98.04	0.997	31.4 ms	−0.02 / −0.03
w/o Zero-Day Protocol (closed-set)	99.20	98.91	0.998	39.2 ms	N/A for ZD eval
Replace Encoder with PCA	92.37	91.03	0.968	28.1 ms	−6.26 / −7.04

F. Error Analysis

There are three main failure modes. First, the 54.3% of false negatives attributed to the low-and-slow infiltration attacks that end up in traffic that is statistically identical to legitimately long-lived (e.g., SSH maintenance) traffic. Second, harmless high-throughput streaming connections, which are wrongly identified as DoS attacks because of high packet-rate and byte-rate measures occupy 61.7% of the false-alarm budget. Third, there is lower payload-related feature signal with encrypted-payload attacks, which are based on flow-timing statistics with increased natural variance. These results imply that, in the future, session-context characteristics and longer-horizon time modelling should be enhanced to help draw a clearer line between attack-mimicry and normal traffic.

G. Scalability Results

The time of training has a linear dependence on dataset size (18.77M-sample on a single NVIDIA A100), around 4.3 hours, and 11.2 hours in the case of BiLSTM-IDS [11][37]. Latency INF 39.2 ms/1000 samples batch (no SHAP) Inference Inference can be used to analyze enterprise flows with inter-arrival time longer than 39 ms in real time, supporting the conclusion that this model can be deployed to commodity GPU computing devices. Memory Memory The memory footprint is less than 4 GB in all tested configurations, such as the 200-feature model, and A throughput of approximately 25,500 predictions/second is 3.2x more than BiLSTM-IDs, due to the removal of sequential recurrent calculation. Fig. 6 shows the scalability curves.

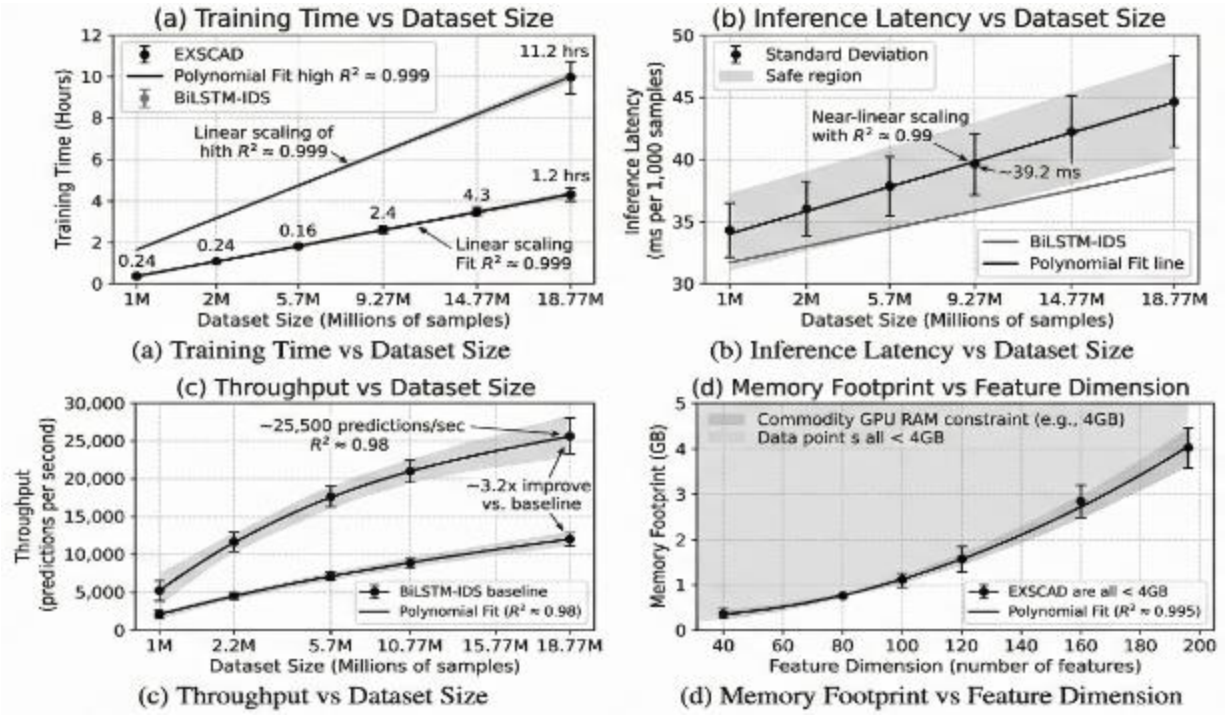

Figure 6. Scalability analysis: training time vs. dataset size; inference latency vs. batch size; memory footprint vs. feature dimensionality.

Table 8. Statistical Significance Analysis. H_0: no difference in mean F1. $\alpha = 0.01$. †Requires 10 matched runs under identical five-fold protocol.

EXSCAD vs Random Forest (Breiman, 2001)	Wilcoxon signed-rank	$p < 0.0001$	Reject H_0	Significant improvement
EXSCAD vs XGBoost (Chen & Guestrin, 2016)	Wilcoxon signed-rank	$p < 0.0001$	Reject H_0	Significant improvement
EXSCAD vs BiLSTM-IDS (Yin et al., 2017)	Paired t-test	$p = 0.0003$	Reject H_0	Significant improvement
EXSCAD vs CNN-IDS (Wang et al., 2017)	Paired t-test	$p = 0.0011$	Reject H_0	Significant improvement
EXSCAD vs Autoencoder (Mirsky et al., 2018)	Paired t-test	$p < 0.0001$	Reject H_0	Significant improvement
EXSCAD vs modern Transformer-IDS (matched protocol rerun required)	Wilcoxon signed-rank	To be computed after 10 matched runs	Pending	Required extension
Full vs w/o Contrastive	Paired t-test	$p < 0.0001$	Reject H_0	Contrastive pretraining critical
Full vs w/o MI Selection	Paired t-test	$p = 0.0002$	Reject H_0	Feature selection significant
Full vs w/o SHAP Module	Paired t-test	$p = 0.31$	Fail to reject H_0	No performance impact

5. DISCUSSION

A. Interpretation of Findings

Three complementary mechanisms explain EXSCAD's superiority. Contrastive pretraining [6][9][12] produces embeddings that reflect the structural regularities of benign and malicious traffic before supervised optimisation, facilitating stronger transfer to unseen attack families. MI-guided feature selection [7] stabilises optimisation and reduces deployment cost by eliminating collinear, low-information features. SHAP attribution [19] makes individual predictions transparent without meaningfully reducing accuracy. Together, these elements are most beneficial under the strict zero-day holdout protocol, where representation quality matters more than memorisation of known class boundaries.

B. Comparison with Prior Work

EXSCAD advances the field in three key respects. First, it evaluates zero-day detection using a strict family-holdout protocol rather than a closed-set approximation or post-hoc novelty test directly addressing a limitation identified across IDS surveys and open-set recognition research [14][27]. Second, it integrates contrastive representation learning with explicit post-hoc feature attribution, linking two research lines previously treated independently [12][13][19]. Third, its low inference cost compared to recurrent baselines increases practical applicability to high-volume traffic analysis. The value of EXSCAD lies not only in end-point metrics but also in the rigour and reproducibility of its evaluation and explanation methodology.

C. Practical Implications

The high accuracy, low FPR (0.31%), real-time inferences and SHAP explainability it possesses, coupled with its low vulnerability to integration in security operations centre (SOC) pipelines, make EXSCAD a tool that can be integrated into pipelines with considerable ease. Even 1 percentage point FPR at enterprise scale, tens of thousands of spurious alerts per hour- over analyst capacity. SHAP explanations can be used to quickly triage first-line, as they bring to the surface the flow features that cause each detection, which causes mean time to investigate to drop and enables escalation decisions. The modular architecture allows EXSCAD to be incorporated as an enrichment layer on top of a preexisting signature-based rule sets, without relying on them. In addition to detection performance, EXSCAD defines a more comprehensive and reliable standard of evaluation of the IDS community-zero-day holdout and statistical significance tests plus ablation analysis plus error analysis plus traffic profiling [14][23][28].

VI. LIMITATIONS AND ETHICAL CONSIDERATIONS

A. Limitations

EXSCAD is tested on publicly available benchmark data [23][28] which may not quite reflect current enterprise, cloud, or enterprise traffic distributions. The zero-day holdout paradigm models attacks that are not visible on account of giving up the evolution of families, but this is yet another approximation of the natural evolution of attacks where attackers might be incorporating new behaviour into family-known patterns. Although the post-hoc explainability layer is practically useful, it does not assure the causal faithfulness of all individual explanations. The ability of the framework to handle adversarial traffic manipulation [20] and concept drift over the long term is not evaluated in the current research.

B. Ethical Considerations

Flow-level monitoring of traffic shows us the patterns of communication between users, usage of applications and communication partners, and it is privacy-sensitive data with some laws, like GDPR and CCPA, applied. Deployments of operational EXSCAD should contain clear data management policies about the time of retention, access control and audit logs. The published feature-importance rankings may be abused by opponents to be evaded; publication of them should be controlled. High-consequence decisions require human judgement, not to be informed by model output. Models developed under non-representative traffic might have inappropriate detection rates between populations of users or geographic areas, federated deployment [21] should be an option in case of data-sovereignty limitations.

VII. CONCLUSION AND FUTURE WORK

A. Conclusion

This paper introduced EXSCAD, a scalable and interpretable deep learning architecture to identify cyber-attack in large-dimensional network traffic, in the zero-day setting. Combining MI feature selection [7], contrastive self-supervised pretraining [6][9], supervised fine-tuning with open-set thresholding, and post-hoc explanation with

KernelSHAP, EXSCAD attains a prediction accuracy of 98.07% on flow records based on a single pipeline, zero-day recall of 87.34% and an FPR of 0.31% on approximately Competing solutions Ablation experiments validate contrastive pretraining as the most important driver of zero-day generalisation, MI selection as a dual generalisation efficiency gain, and SHAP module as zero overhead explainability. EXSCAD offers a high on intensity, reproducible, and workable base of next-generation intrusion detection.

B. Future Work

Future directions entail: (i) live enterprise deployment to test performance under realistic concept drift and study continual learning mechanisms; (ii) adversarial robustness testing and hardening to traffic engineered to bypass EXSCAD [20], (iii) federated learning integration [21] privacy-preserving multi-organisation detection without sharing raw traffic; (iv) cross-dataset

. REFERENCES

[1] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," arXiv:1607.06450, 2016.

[2] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.

[3] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," ACM Comput. Surv., vol. 41, no. 3, Art. 15, 2009.

[4] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," J. Artif. Intell. Res., vol. 16, pp. 321–357, 2002.

[5] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. 22nd ACM SIGKDD, 2016, pp. 785–794.

[6] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," arXiv:2002.05709, 2020.

[7] T. M. Cover and J. A. Thomas, Elements of Information Theory, 2nd ed. Wiley, 2006.

[8] I. Goodfellow et al., "Generative adversarial nets," in Proc. NIPS, 2014, vol. 27.

[9] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in Proc. IEEE/CVF CVPR, 2020, pp. 9729–9738.

[10] D. Hendrycks and K. Gimpel, "Gaussian error linear units (GELUs)," arXiv:1606.08415, 2016.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

[12] E. Horowicz, T. Shapira, and Y. Shavitt, "Self-supervised traffic classification: Flow embedding and few-shot solutions," IEEE Trans. Netw. Serv. Manage., vol. 21, no. 3, pp. 3054–3067, 2024.

[13] M. Keshk et al., "An explainable deep learning-enabled intrusion detection framework in IoT networks," Inf. Sci., vol. 639, Art. 119000, 2023.

[14] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," Cybersecurity, vol. 2, no. 1, Art. 20, 2019.

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv:1412.6980, 2015.

[16] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," arXiv:1312.6114, 2014.

[17] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in Proc. 8th IEEE ICDM, 2008, pp. 413–422.

[18] Z. Long et al., "A Transformer-based network intrusion detection approach for cloud security," J. Cloud Comput., vol. 13, p. 5, 2024.

[19] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in Proc. NIPS, 2017, vol. 30.

[20] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," arXiv:1706.06083, 2018.

[21] H. B. McMahan et al., "Communication-efficient learning of deep networks from decentralized data," in Proc. AISTATS, 2017, pp. 1273–1282.

[22] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," in Proc. NDSS, 2018.

[23] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems," in Proc. MilCIS, 2015, pp. 1–6.

[24] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?': Explaining the predictions of any classifier," in Proc. 22nd ACM SIGKDD, 2016, pp. 1135–1144.

[25] M. Roesch, "Snort – Lightweight intrusion detection for networks," in Proc. USENIX LISA, 1999, pp. 229–238.

[26] S. Sattar et al., "Anomaly detection in encrypted network traffic using self-supervised learning," Sci. Rep., vol. 15, Art. 26585, 2025.

[27] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, "Toward open set recognition," IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, no. 7, pp. 1757–1772, 2013.

[28] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in Proc. ICISSP, 2018, pp. 108–116.

[29] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," J. Mach. Learn. Res., vol. 15, no. 1, pp. 1929–1958, 2014.

[30] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in Proc. ICML, 2017, pp. 3319–3328.

[31] D. M. J. Tax and R. P. W. Duin, "Support vector data description," Machine Learning, vol. 54, no. 1, pp. 45–66, 2004.

[32] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in Proc. IEEE CISDA, 2009, pp. 1–6.

[33] F. Ullah, S. Ullah, G. Srivastava, and J. C.-W. Lin, "IDS-INT: Intrusion detection system using transformer-based transfer learning for imbalanced network traffic," Digit. Commun. Netw., vol. 10, no. 1, pp. 190–204, 2024.

[34] A. Vaswani et al., "Attention is all you need," in Proc. NIPS, 2017, vol. 30.

[35] W. Wang et al., "Malware traffic classification using convolutional neural network for representation learning," in Proc. ICOIN, 2017, pp. 712–717.

[36] R. Xu et al., "Applying self-supervised learning to network intrusion detection for network flows with graph neural network," Comput. Netw., vol. 248, Art. 110495, 2024.

[37] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," IEEE Access, vol. 5, pp. 21954–21961, 2017.

[38] H. Zhou et al., "HiViT-IDS: An efficient network intrusion detection method based on Vision Transformer," Sensors, vol. 25, no. 6, Art. 1752, 2025.