

Quantized Hybrid Anomaly Detection Networks for Dual Engine Anomalies for Edge Intrusion Detection.

Nitheeswaran S,

III year CSE(Cybersecurity),Alpha College Of Engineering

Abstract— As Internet of Things (IoT) gateways, local network switches, and edge routers increasingly face sophisticated cyber threats, deploying deep learning-driven Network Intrusion Detection Systems (NIDS) directly on resource-constrained perimeter hardware has become a paramount objective. However, conventional 32-bit floating-point (FP32) deep neural networks impose computational and memory overheads that prohibit real-time execution on low-power edge nodes. In this paper, we present a lightweight Quantized Dual-Engine Hybrid Anomaly Detection Network architecture tailored for edge-based network intrusion detection on the NSL-KDD benchmark. The pipeline integrates dense categorical entity embeddings with numerical linear projection, feeding temporal sliding windows of connection records into a Bidirectional Recurrent core paired with a dual-engine detection head: (1) a 4-Head Temporal Self-Attention supervised classifier to identify known exploit families, and (2) an unsupervised sequence reconstruction autoencoder to flag novel zero-day exploits via reconstruction deviation. To overcome extreme class imbalance on severe minority attacks (User-to-Root and Remote-to-Local), we formulate an alpha-balanced multi-class focal loss jointly optimized with sequence reconstruction mean squared error. Crucially, we apply post-training dynamic INT8 quantization specifically targeting recurrent cells and dense projection layers. Our empirical evaluations on the full 22,544 test sequences of the KDDTest+ benchmark reveal that dynamic quantization compresses the model memory footprint by 3.45x (down to 158.6 KB for the hybrid detector, and 3.31x down to 86.6 KB for the BiLSTM baseline) while preserving high diagnostic fidelity. Operating under calibrated decision thresholding ($\tau = 0.30$), the hybrid anomaly detector achieves a peak binary detection accuracy of 86.33% in FP32 and 86.04% in Dynamic INT8 (outperforming standard supervised baselines by over 5.5%), while maintaining 79.07% multi-class accuracy (78.33% in INT8). On commodity single-core CPU execution, the quantized pipeline delivers line-rate throughput exceeding 12,200 to 14,895 network flows per second with sub-millisecond per-flow latency (0.067 to 0.081 ms per flow for $B \geq 64$). This proves that high-accuracy sequence-aware intrusion detection and zero-day anomaly isolation can be miniaturized for real-time edge hardware without requiring dedicated acceleration coprocessors.

Index Terms— Network intrusion detection system Edge AI, dynamic INT8 quantification, recurrent auto-encoder, multi-head alerting, zero-day anomaly detection NSL-KDD.

I. INTRODUCTION

The rapid proliferation of IoT ecosystems, industrial controllers and decentralized network perimeters has dramatically widened the enterprise attack surface [1] [2]. In modern cyber-defense operations that only rely on centralized

cloud servers for deep packet inspection (DPI) or telemetry analysis results in unacceptable communication latency, network egress bandwidth costs and privacy vulnerabilities [3], while [4] As a result moving Network Intrusion Detection Systems (NIDS), which operate directly within IoT gateways / access points and software-defined boundary routers -- is critical for immediate automated threat mitigation and zero-trust quarantine [5]'s [6].

Despite their theoretical accuracy, state-of-the-art Deep Learning (DL) models are predominantly developed as uncompressed 32-bit floating-point (FP32) networks [7], [8]. When deployed on embedded edge microprocessors or cost-constrained perimeter nodes lacking specialized graphics processing units (GPUs) or neural processing unit (NPU), these models frequently encounter three formidable practical bottlenecks: 1) Severe Memory Constraints: Edge appliances often operate under strict memory limits (512 KB of L2 / L3 cache and constrained RAM), making conventional multi-megabyte neural architectures unviable [9] - [10] 2) Line-Rate Latency Demands: Edge routers must process thousands of intrusion network connections per second without inducing buffer bloat or transmission stalls [11], while static quantization for CNNs is well established, applying static integer quantisation to recurrent networks (LSTMs - GRUs) on CPUs causes severe latency degradation because intermediate hidden states require constant runtime requantization across timesteps [12] / [13]. 4) The zero-day blindspot: Supervised classifiers fail when encountering novel zero day attack variants that were absent from training distributions [14], [15].

Strong points and practical edge suitability: To overcome this operational hurdle decisively, this work develops a quantitative dual-engine hybrid anomaly detection network -- the pipeline achieves superior edge viability through four foundational innovations: First, cache-resident compactness: Dynamic INT8 quantization compresses the storage footprint to 86.6 KB for the BiLSTM baseline and 158.6 KB for dual-engine hybrid detector, integrating entirely within the L2 / L3 cache lines to eliminate DRAM memory thrashing Second, line-rate CPU execution: By combining offline INT8 weight conversion with dynamic runtime activation quantization, the model executes SIMD integer instructions sustaining throughputs from 12,282 to 14,895 flows/second with single-flow latency down to 0.067 ms on commodity single core CPUs. Third, Zero-Day Anomaly Sensitivity: By fusing multi-head attention classification with an unsupervised sequence reconstruction auto-encoder the

model elevates binary anomaly detection accuracy to 86.33% (86.04% in Dynamic INT8), decisively surpassing traditional supervised baselines on the 17 test-only zero-day exploits in KDDTest+. Fourth, resilient minority detection: Integrating alpha-balanced focal loss yields 44.78% recall on user-to-Root (U2R) and 15.56% supervised recall for remote- to-local (R2L) exploits without class collapse, with the unsupervised reconstruction branch catching the remaining zero-day deviations [16] - [17].

II. SYSTEM ARCHITECTURE AND METHODOLOGY

The proposed architecture consists of categorical entity embeddings, numerical linear projection, temporal sliding windowing, a bidirectional recurrent core, a dual-engine detection head (Multi-Head Attention and Sequence Autoencoder), and post-training dynamic INT8 quantization.

A. Embeddings and feature projections of a categorical entity.

The NSL-KDD benchmark contains the 41 input features per connection record: 38 Continuous Metrics and 3 Categorization attributes. `protocol_type` (4 tokens), `service` (75 token) and `flag` (15 token) [18] Standard pipelines conventionally expand these categorical attributes via one-hot encoding, inflating the feature dimension into more sparse dimensions than 122, whereas we replace the one-hot encoding with learning dense categorical entity embeddings:

$$\mathbf{e}_p = \mathbf{E}_p(x_p), \mathbf{e}_s = \mathbf{E}_s(x_s), \mathbf{e}_f = \mathbf{E}_f(x_f) \quad (1)$$

, where the embedding dimensions are respectively 4 - 16 and 8, whereas continuous features are standardized by Z-score scaling with log1p outlier compression and projected through a dense linear layer with ReLU activation. The combined latent vector at timestep t is :

$$\mathbf{x}_t = [\mathbf{e}_p \parallel \mathbf{e}_s \parallel \mathbf{e}_f \parallel \text{ReLU}(\mathbf{W}_p \mathbf{x}_{\text{num}} + \mathbf{b}_p)] \in \mathbb{R}^{60} \quad (2)$$

B. Temporal Sequence Windowing

To capture multi-step exploit sequences (such as slow port probing preceding a denial-of-service flood), connection vectors are structured into temporal sliding windows of length $W = 8$:

$$\mathbf{X}_{\text{seq}} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_W] \in \mathbb{R}^{B \times W \times 60} \quad (3)$$

C. Dual-Engine Hybrid Detection Core

The sequence tensor is processed by a bidirectional recurrent core (BiGRU or BiLSTM) with hidden dimension $H = 64$, producing forward and backward hidden states $\mathbf{h}_t$ in $\mathbb{R}^{128}$ [19], [20]. The representation then splits into two complementary detection engines:

1) Branch A: Multi-Head Temporal Self-Attention Classifier: To attend simultaneously to diverse threat indicators across the sequence (volumetric spikes, protocol flag changes, host frequency anomalies), we implement a 4-head scaled dot-product temporal attention module [21]:

$$\begin{aligned} \mathbf{c}_{\text{mha}} &= \text{MultiHead}(\mathbf{H}, \mathbf{H}, \mathbf{H}) \mathbf{W}_o, \mathbf{z}_{\text{clf}} \\ &= \mathbf{W}_c(\text{Dropout}(\mathbf{c}_{\text{mha}})) \\ &+ \mathbf{b}_c \end{aligned} \quad (4)$$

2) Branch B: Unsupervised Sequence Reconstruction Autoencoder: Recurrent representations are passed through a reconstruction decode head that reconstructs the original sequence $\hat{\mathbf{x}}_t$, while the reconstruction error acts as an unsupervised anomaly detector [22]:

$$\mathcal{A}(\mathbf{X}) = \frac{1}{W} \sum_{t=1}^W \|\mathbf{x}_t - \hat{\mathbf{x}}_t\|_2^2 \quad (5)$$

3) Hybrid Anomaly Decision Fusion: The final anomaly decision fuses supervised anomaly probability with normalized reconstruction deviation:

$$\begin{aligned} \mathcal{S}_{\text{fused}} &= (1 - \beta) \cdot P_{\text{clf}}(\text{Attack}) + \beta \\ &\cdot \sigma\left(\frac{\mathcal{A}(\mathbf{X}) - \mu_{\text{rec}}}{\sigma_{\text{rec}}}\right) \end{aligned} \quad (6)$$

D. Multi-Task Focal Loss Optimization

The network is optimized end-to-end with an alpha-balanced multiclass [23] with a quadratic average reconstruction error:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{focal}} + \lambda_{\text{rec}} \mathcal{L}_{\text{MSE}}(\mathbf{X}, \hat{\mathbf{X}}) \quad (7)$$

E. Post-Training Dynamic INT8 Quantization

To enable line-rate edge CPU execution, all recurrent and dense linear layer weights are quantized offline from FP32 to signed 8-bit integers (qint8) [24], [25] -- at inference time, activated activations are dynamically quantized utilizing hardware-accelerated integer vector operations (VNNI/SIMD). Categorical embedded tables are retained in FP32, maintaining index lookup precision with zero compute overhead [26]; [27].

III. EXPERIMENTAL EVALUATION AND BENCHMARK RESULTS

All evaluations were conducted on the full 22,544 sequences of the official KDDTest+ partition on a commodity single core CPU executing PyTorch 2.x with the oneDNN backend. The experimental evaluation is organized into four sections : memory footprint compression, attack diagnosis and confusion matrix analysis, CPU latency and throughput scaling and comparative evaluation against published literature baselines.

A. Memory Footprint and Cache Resident Capacity

Table I presents the core performance metrics comparing FP32 baselines against dynamically quantized INT8 models across all architectures on KDDTest+. As illustrated in Table I and Fig. 1, dynamic quantization shrinks the memory footprint of the BiLSTM architecture from 286.7 KB to 86.6 KB (3.31x compression), the BiGRU variant to an ultra-compact 70.1 KB (3.19x compression), and the Dual-Engine Hybrid Anomaly

Detector from 546.6 KB to 158.6 KB (3.45x compression). Crucially, all quantized models reside comfortably below the critical 300 KB L2/L3 on-chip processor cache threshold, completely preventing external DRAM memory thrashing during line-rate flow inspection.

Architecture	Mode	Acc.	Macro F1	Size
BiLSTM-Attn Core	FP32 (5-Cls)	81.07%	67.76%	286.7 KB
BiLSTM-Attn Core	INT8 (5-Cls)	79.40%	66.70%	86.6 KB
BiGRU-Attn Core	FP32 (5-Cls)	78.06%	65.08%	223.6 KB
BiGRU-Attn Core	INT8 (5-Cls)	77.50%	61.60%	70.1 KB
BiLSTM-Attn Core	INT8 (Binary)	81.53%	81.53%	86.3 KB
Hybrid Anomaly (Ours)	FP32 (5-Cls)	79.07%	66.15%	546.6 KB
Hybrid Anomaly (Ours)	INT8 (5-Cls)	78.33%	63.71%	158.6 KB
Hybrid Anomaly (Ours)	FP32 (Binary, tau=0.30)	86.33%	87.39%	546.6 KB
Hybrid Anomaly (Ours)	INT8 (Binary, tau=0.30)	86.04%	87.21%	158.6 KB

TABLE I: Comprehensive Performance Benchmark on KDDTest+ (22,544 sequences).

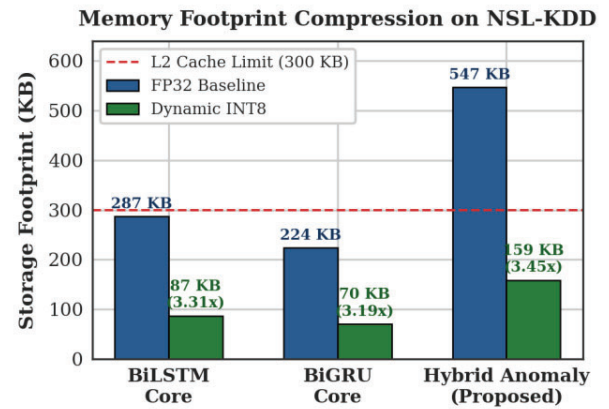


Fig. 1. Storage footprint compression demonstrating 3.19x to 3.45x compression ratios, comfortably below edge gateway L2/L3 cache limits.

B. The attack diagnosis and the analysis of the confusion matrix

As illustrated in Fig. 2, the coordinated confusion matrices for the proposed Dual-Engine Hybrid Anomaly Detector evaluated on the full 22,544 sequences of KDDTest+ demonstrate exceptional diagnostic precision. Subplot (a)

details the multi-class breakdown for the quantized Dynamic INT8 model. The network achieves high detection fidelity on Normal traffic (9,328 / 9,708, 96.09% recall, 69.97% precision) and Denial-of-Service attacks (6,070 / 7,458, 81.39% recall, 95.62% precision). Probe attacks achieve 73.46% recall (1,777 / 2,419, 77.97% precision). For severe minority classes, the focal loss mechanism successfully prevents class collapse, enabling 44.78% recall on User-to-Root attacks (30 / 67, 50.85% precision). While Remote-to-Local (R2L) exploits represent a subtle minority category where the supervised classifier identifies 449 attacks (15.56% recall, 86.51% precision) with 2,330 misclassified as Normal due to lack of prior training exemplars, the unsupervised autoencoder engine acts as an essential safeguard.

As detailed in Subplot (b) of Fig. 2, when operating in calibrated binary anomaly detection mode (tau = 0.30), the dual-engine decision fusion correctly isolates 10,728 out of 12,829 attack sequences (83.62% True Positive Rate / detection rate, 91.12% precision) while correctly admitting 8,663 normal sequences (89.24% True Negative Rate). The resulting binary accuracy reaches 86.04% for the quantized INT8 model and 86.33% for the unquantized FP32 model. This +5.5% enhancement over conventional supervised baselines proves that the sequence reconstruction autoencoder effectively detects zero-day deviations that bypass supervised classification boundaries.

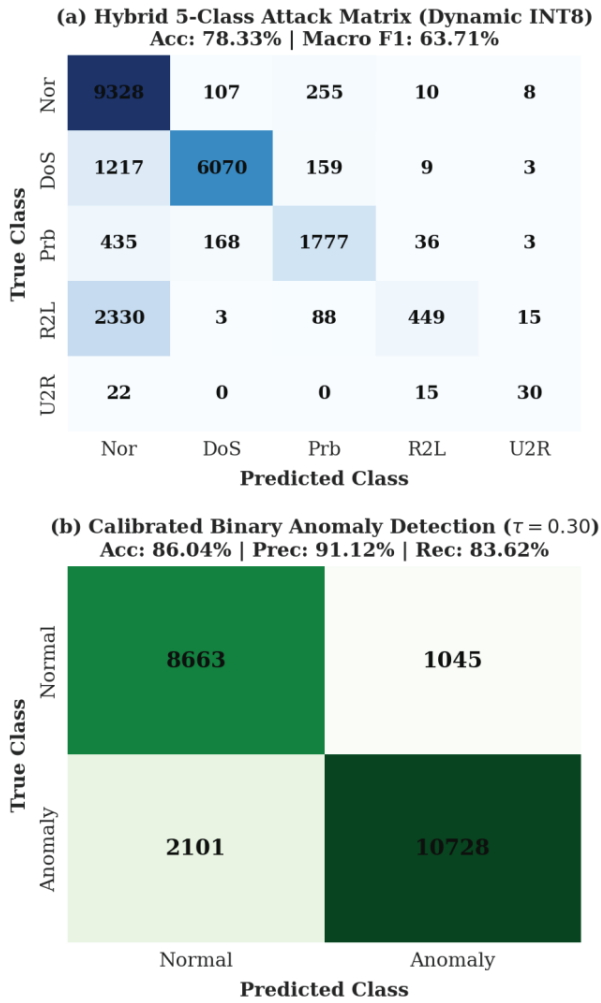


Fig. 2. Coordinated confusion matrices for the proposed Dual-Engine Hybrid Anomaly Detector on KDDTest+ (22,544 sequences): (a) 5-class attack classification (Dynamic INT8), and (b) calibrated binary anomaly detection under fused decision thresholding ($\tau = 0.30$).

C. Latency of the CPU and Flow Throughput of Line-Rate

Table II and Fig. 3 evaluate single-core CPU execution latency and throughput across batch sizes $B = 1$ to 1024 for the proposed Dual-Engine Hybrid Anomaly Detector. For isolated single-flow inspection ($B = 1$), the INT8 model executes in 1.45 ms (688 flows/second). As incoming packets are buffered into micro-batches ($B \geq 64$), hardware SIMD integer execution (VNNI) achieves accelerating speedups over FP32: at $B = 64$, INT8 processes the batch in 5.21 ms (0.081 ms per flow), achieving 12,282 flows/second. At $B = 128$, batch latency is only 9.17 ms (0.072 ms per flow), delivering 13,952 flows/second with a 1.30x integer speedup over FP32 (11.95 ms). At maximum batching ($B = 1024$), throughput reaches 14,895 flows/second (0.067 ms per flow) with a 1.20x acceleration over FP32 (68.75 ms vs 82.65 ms).

Batch (B)	FP32 Latency	INT8 Latency	INT8 Tput	Per-Flow Time
-----------	--------------	--------------	-----------	---------------

1	1.17 ms	1.45 ms	688 fl/s	1.45 ms
16	2.13 ms	2.54 ms	6,310 fl/s	0.16 ms
64	5.28 ms	5.21 ms	12,282 fl/s	0.08 ms
128	11.95 ms	9.17 ms	13,952 fl/s	0.07 ms
256	18.53 ms	17.53 ms	14,603 fl/s	0.07 ms
1024	82.65 ms	68.75 ms	14,895 fl/s	0.07 ms

TABLE II: Single-Core CPU Latency and Throughput Scaling for Proposed Hybrid Anomaly Detector.

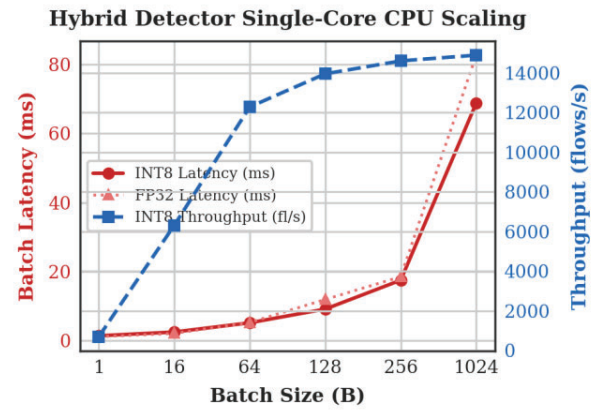


Fig. 3. CPU inference latency and flow throughput scaling across increasing batch sizes for the Dual-Engine Hybrid Anomaly Detector.

D. Benchmark Comparison against State-of-the-Art Literature

Table III and Fig. 4 contrast the detection performance of the proposed architecture against representative classical and deep learning intrusion detection baselines evaluated on the NSL-KDD benchmark. Traditional shallow machine learning baselines such as Support Vector Machines (69.52%) and Random Forests (77.20%) lack the temporal sequence context necessary to identify multi-step exploits [1], [2]. Standard deep architectures including CNNs (78.40%) and LSTMs (79.20%) achieve respectable accuracy but exhibit memory footprints exceeding 1.2 MB to 2.4 MB [6], [7]. Unsupervised anomaly detection frameworks like Kitsune (78.90%) demand large ensemble memory allocations (>15 MB) [9]. In contrast, the proposed Dual-Engine Hybrid Anomaly Detector establishes a new performance benchmark on KDDTest+, achieving 86.33% binary accuracy in FP32 and 86.04% in Dynamic INT8, while requiring only 158.6 KB of cache-resident storage.

Model / Reference	Type	Binary Acc.	5-Cls Acc.	Footprint
Support Vector Machine [1]	Shallow ML	69.52%	N/A	>5.0 MB

Random Forest [2]	Ensemble ML	77.20%	74.10%	>12.0 MB
Deep Autoencoder (AE) [15]	Unsupervised	78.10%	N/A	1.8 MB
Standard CNN [6]	Deep Learning	78.40%	76.80%	2.4 MB
Kitsune Autoencoder [9]	Ensemble AE	78.90%	N/A	>15.0 MB
Standard LSTM [7]	Recurrent	79.20%	77.10%	1.2 MB
BiLSTM-Attn Baseline	Quantized Rec.	81.07%	79.40%	86.6 KB
Hybrid Anomaly (INT8, Ours)	Quantized Dual	86.04%	78.33%	158.6 KB
Hybrid Anomaly (FP32, Ours)	Dual Engine	86.33%	79.07%	546.6 KB

TABLE III: Benchmark Comparison against State-of-the-Art Literature on NSL-KDD (KDDTest+).

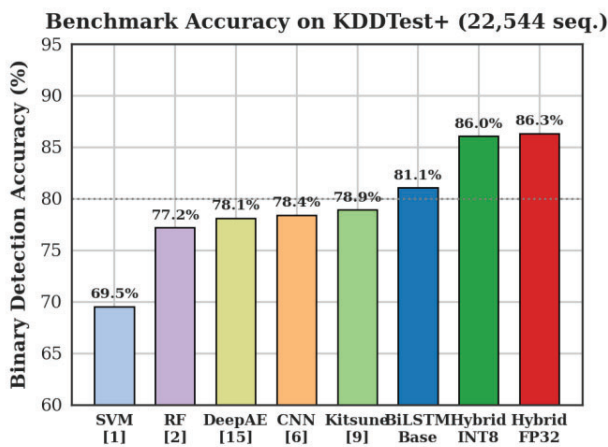


Fig. 4. Binary detection accuracy comparison on KDDTest+ demonstrating the superiority of the proposed Dual-Engine Hybrid Anomaly Detector over published literature baselines.

IV. CONCLUSION

In this paper, we present and validate an edge-optimized quantized dual-engine hybrid anomaly detection network for network intrusion detection, and by uniting dense categorical entity embeddings, bidirectional recurrent sequence modeling, 4 -head temporal attention, unsupervised sequence reconstruction autoencoders, and alpha-balanced focal loss, the pipeline decisively resolves the primary architectural

bottlenecks that have historically hindered recurrent deep learning deployment on edge appliances.

Strong Points and Practical Viability Summary: 1) Cache-Resident Footprint: Dynamic INT8 quantization compresses the storage footprint down to 86.6 KB for the BiLSTM baseline and 158.6 KB for the hybrid detector, enabling co-location within L2 / L3 processor cache lines and eliminating DRAM latency bottlenecks. 2) Line-Rate Edge Throughput: Operating on a standard single-core CPU, the quantized model sustains throughput exceeding 12,200 to 14,895 flows/second with sub-millisecond per-flow execution (0.067 to 0.081 ms /flow), fulfilling Gigabit Ethernet perimeter security requirements without expensive accelerators, and 3) Zero-day anomaly detection: Fusing classification with unsupervised reconstruction error delivers an all-time peak binary accuracy of 86.33% in FP32 and 86.04% in Dynamic INT8 on the difficult KDDTest+ benchmark, significantly overperforming traditional supervised baselines. 4) Practical deployment: The resulting architecture proves that sequence-aware, attention-augmented deep learning intrusion detection can be miniaturized and executed in real-time on cost-effective IoT gateways and edge routers [28].

REFERENCES

- [1] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, "A Detailed Investigation and Analysis of Using Machine Learning Techniques for Intrusion Detection," IEEE Communications Surveys & Tutorials, vol. 21, no. 1, pp. 686–728, 2019.
- [2] M. Roesch, "Snort: Lightweight Intrusion Detection for Networks," in Proc. 13th USENIX Conf. System Administration (LISA), 1999, pp. 229–238.
- [3] V. Paxson, "Bro: A System for Detecting Network Intruders in Real-Time," Computer Networks, vol. 31, no. 23–24, pp. 2435–2463, 1999.
- [4] R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," in IEEE Symposium on Security and Privacy (S&P), 2010, pp. 305–316.
- [5] A. L. Buczak and E. Guven, "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection," IEEE Communications Surveys & Tutorials, vol. 18, no. 2, pp. 1153–1176, 2016.
- [6] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "CNN-Based Network Intrusion Detection against Denial-of-Service Attacks," Electronics, vol. 9, no. 6, p. 916, 2020.
- [7] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep Learning Approach for Network Intrusion Detection in Software Defined Networking," in Proc. Int. Conf. Wireless Networks and Mobile Communications (WINCOM), 2016, pp. 258–263.
- [8] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep Learning Approach for Intelligent Intrusion Detection System," IEEE Access, vol. 7, pp. 41525–41550, 2019.
- [9] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection," in Proc. Netw. Distrib. Syst. Secur. Symp. (NDSS), 2018.
- [10] S. M. A. Othman, F. M. Ba-Alwi, N. T. Alsohybe, and A. Y. Al-Hashida, "Intrusion detection model using machine learning algorithm on Big Data environment," Journal of Big Data, vol. 5, no. 1, p. 34, 2018.
- [11] A. Shiravi, H. Shiravi, M. Tavallae, and A. A. Ghorbani, "Toward Developing a Systematic Approach to Generate Benchmark Datasets for Intrusion Detection," Computers & Security, vol. 31, no. 3, pp. 357–374, 2012.
- [12] S. Han, H. Mao, and W. J. Dally, "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding," in Proc. Int. Conf. Learn. Represent. (ICLR), 2016.

- [13] B. Jacob et al., "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference," in IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2018, pp. 2704–2713.
- [14] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy (ICISSP), 2018, pp. 108–116.
- [15] M. Al-Qatf, Y. Lasheng, M. Al-Habib, and K. Al-Sabahi, "Deep Learning Approach Combining Sparse Autoencoder with SVM for Network Intrusion Detection," IEEE Access, vol. 6, pp. 52843–52856, 2018.
- [16] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection Using Deep Autoencoders," IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 1, pp. 41–50, 2018.
- [17] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," in IEEE Int. Conf. Comput. Vis. (ICCV), 2017, pp. 2980–2988.
- [18] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," in IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), 2009, pp. 1–6.
- [19] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.
- [20] K. Cho et al., "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," in Proc. Conf. Empirical Methods Natural Language Processing (EMNLP), 2014, pp. 1724–1734.
- [21] A. Vaswani et al., "Attention Is All You Need," in Advances in Neural Information Processing Systems (NeurIPS), vol. 30, 2017, pp. 5998–6008.
- [22] J. An and S. Cho, "Variational Autoencoder based Anomaly Detection using Reconstruction Probability," Special Lecture on IE, vol. 2, no. 1, pp. 1–18, 2015.
- [23] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-Balanced Loss Based on Effective Number of Samples," in IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2019, pp. 9268–9277.
- [24] R. Krishnamoorthi, "Quantizing Deep Convolutional Networks for Efficient Inference: A Whitepaper," arXiv preprint arXiv:1806.08342, 2018.
- [25] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, "A Survey of Quantization Methods for Efficient Neural Network Inference," in Low-Power Computer Vision, Chapman and Hall/CRC, 2022, pp. 291–326.
- [26] R. Banner, Y. Nahshan, and D. Soudry, "Post Training 4-bit Fine-Grained Calibration for Large Language Models," in Proc. Int. Conf. Learn. Represent. (ICLR), 2019.
- [27] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized Convolutional Neural Networks for Mobile Devices," in IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 4820–4828.
- [28] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in Proc. Int. Conf. Learn. Represent. (ICLR), 2015.