HFS-VAE: A Hierarchical Feature-Splitting Variational Autoencoder for Interpretable and Robust Network Intrusion Detection

Munnangi Suresh[1], Shaik Mohammad Thaheer[2], Derangula Durga Rao[3], Shaik Mannepalli Mastanvali[4],
Panduri Bharathi[5], Vavilala Divya Raj[6], Dr. Sireesha Moturi[7]
[1,2,3,4,7]Department of Computer Science and Engineering,
Narasaraopeta Engineering College (Autonomous), Narasaraopet, India
[5]Department of Information Technology, GRIET, Bachupally, Hyderabad, Telangana, India
[6]Department of Computer Science and Engineering,
G. Narayanamma Institute of Technology & Science (Women), Shaikpet, Hyderabad, Telangana, India
[1]sureshmunnangi55@gmail.com, [2]shaikthaheer752@gmail.com, [3]ddurgarao037@gmail.com,
[4]mannepallimastanvalli@gmail.com, [5]bharathi1284@grietcollege.com,
[6]divyaraj.vavilala@gnits.ac.in, [7]sireeshamoturi@gmail.com

***Abstract*—The complexity of modern networked environments has increased the risk of advanced cyber threats, making effective Network Intrusion Detection Systems (NIDS) essential. Deep learning has improved anomaly detection, but many models still lack robustness and interpretability. We propose the Hierarchical Feature-Splitting Variational Autoencoder (HFS-VAE), a novel architecture that partitions network features into semantic groups and encodes them through parallel branches, followed by classifier fusion. This design enhances transparency in the latent space and improves anomaly localization compared to conventional VAEs. Experiments on CICIDS2017 demonstrate an Average Precision of 0.8412 and an F1-score of 0.7463, while tests on NSL-KDD confirm adaptability with an AP of 0.9234 and an F1 of 0.7371. Robustness is validated through PCA-based latent visualizations, Maximum Likelihood Estimation (MLE), and boundary complexity analysis. Although recall remains lower than precision in some cases and evaluation is limited to CICIDS2017 and NSL-KDD, HFS-VAE provides a practical balance of interpretability and accuracy. Compared with prior approaches, it achieves competitive detection performance and superior anomaly separation, offering a structured and interpretable solution for enterprise-scale cybersecurity.**

***Index Terms*—Network Intrusion Detection, Variational Autoencoder, Deep Learning, Anomaly Detection, Feature Splitting, Cybersecurity**

I. Introduction

The rapid expansion of digital infrastructures, fueled by smart devices and high-speed connectivity, has introduced new vulnerabilities into networked systems. As organizations grow, they face greater exposure to cyber threats that disrupt operations and compromise sensitive data. Network Intrusion Detection Systems (NIDS) have thus become essential for detecting unusual traffic that could signal attacks or unauthorized access. [1], [2].

Although signature-based NIDS are effective for detecting known intrusions, they perform poorly when facing novel or zero-day attacks [3]. To overcome these limitations, machine learning approaches—particularly deep learning—have been widely adopted for adaptive anomaly detection. Variational Autoencoders (VAEs) [4] are capable of learning latent data representations and identifying deviations from normal traffic patterns. Nevertheless, conventional VAEs are highly vulnerable to adversarial perturbations and offer limited transparency, which reduces their suitability in security-critical domains [5].

The NIDS-Vis framework [6] sought to improve robustness and interpretability through feature space partitioning (FSP), adversarial robustness evaluation, and visualization-driven analysis. Nevertheless, its evaluation was limited to the UQ-IoT-IDS dataset [7], which has a narrow scope and does not fully represent enterprise-scale networks. Other domain-specific approaches, such as ABCIS for smart agriculture [8], further emphasize the need for intrusion detection solutions that are adaptable, interpretable, and generalizable.

To bridge these gaps, we introduce the **Hierarchical Feature-Splitting Variational Autoencoder (HFS-VAE)**, which divides input features into semantically related groups and encodes them through parallel branches. The fused latent space improves modularity, supports anomaly localization, and enhances interpretability. Unlike earlier approaches, HFS-VAE combines hierarchical feature partitioning with classifier fusion to achieve structured anomaly separation and resilient detection under noisy traffic. While precision is consistently high, recall remains comparatively lower, reflecting a trade-off explicitly addressed in this study.

The novel aspects of our work are highlighted below:

- An innovative VAE-based intrusion detection system that improves robustness and interpretability by combining

979-8-3315-9524-1/25/$31.00 ©2025 IEEE

classifier fusion, parallel encoders, and hierarchical feature partitioning.
- Extension of visualization-driven NIDS concepts to enterprise-scale datasets (CICIDS2017 and NSL-KDD), moving beyond IoT-specific evaluation.
- Comprehensive evaluation using AP, F1, and AUC metrics, supported by qualitative analyses with latent projections, density scoring, and boundary estimation.
- A balanced discussion of strengths and limitations, including the precision–recall trade-off and dataset coverage, with directions for future improvement.

II. Related Work

The evolution of Intrusion Detection Systems (IDSs) has been greatly influenced by advancements in deep learning. Signature-based methods remain useful for detecting previously identified threats, but they are inadequate against zero-day attacks. To address this gap, anomaly-based strategies have been introduced, aiming to capture normal behavior patterns and identify deviations as potential intrusions [1].

Autoencoders (AEs) and their variants have been widely applied in IDS, using reconstruction error to flag anomalies. Variational Autoencoders (VAEs) [4] extend AEs by learning latent distributions, thus improving uncertainty estimation and generalization. Robust variants such as adversarial and sparse VAEs have been applied in security contexts, while frameworks like NIDS-Vis [6] emphasize explainability through latent visualizations, boundary complexity, and adversarial robustness. Techniques for dimensionality reduction, including t-SNE [9] and UMAP [10], are also employed to enhance the interpretability of the learned feature embeddings.

Datasets remain a critical factor in IDS benchmarking. UQ-IoT [7] provides IoT-specific traffic but with limited attack diversity, while NSL-KDD [11] is outdated and poorly aligned with current network patterns. The CICIDS2017 dataset [12], in contrast, offers diverse enterprise-scale attacks and balanced traffic, making it more suitable for modern evaluation. To improve robustness on such datasets, partitioned learning strategies have been proposed, where features are divided into semantic groups (e.g., flow, packet, flag). Grouped autoencoders [13] and hybrid pipelines, such as VAE embeddings combined with classifiers like XGBoost [14], have demonstrated improved generalization. Complementary evaluation methods include latent density scoring via Maximum Likelihood Estimation (MLE) [15] and adversarial boundary complexity analysis [5].

Recent studies examined comparative analysis of VAE, AAE, and VAE-GAN for anomaly detection [16], lightweight VAEs for IoT [17], hybrid tree–deep models [18], reinforcement learning-based IDS for adaptive detection [19], and surveys on deep learning IDS scalability and interpretability [20]. Building on these directions, our HFS-VAE integrates hierarchical feature partitioning, modular latent learning, and explainability into a unified framework for enterprise-scale intrusion detection.

III. Dataset Description

The HFS-VAE model's effectiveness is assessed using two benchmark intrusion detection datasets: CICIDS2017 and NSL-KDD. Additionally, we analyze the characteristics of the UQ-IoT dataset, which served as the foundation for experiments in the NIDS-Vis framework [6], [7].

A. Dataset Overview

CICIDS2017: Generated in an enterprise-like testbed, CICIDS2017 includes over 3 million NetFlow records, with benign and attack traffic across various types like DDoS, Botnet, PortScan, and Infiltration. It contains 80 numerical features per sample.

NSL-KDD: NSL-KDD improves upon the original KDD'99 dataset by reducing redundant entries and addressing class imbalance. It comprises 41 features and includes five distinct attack types. Despite its improvements, the dataset reflects older network traffic patterns and limited modern threats [11].

UQ-IoT: This IoT-focused dataset captures network traffic from a smart home setup. While useful for edge-level analysis, it lacks diversity and suffers from class imbalance, limiting generalizability.

B. Dataset Comparison

TABLE I
Comparison of Intrusion Detection Datasets

Aspect	**CICIDS2017**	**NSL-KDD**	**UQ-IoT**
Env. Type	Enterprise	Simulated	IoT (Home)
Attack Types	15+	5	7
Samples	3M+	125k+	100k+
Features	80	41	45
Label Balance	Moderate	Balanced	Imbalanced
Gen. Capability	High	Moderate	Low

This comparison highlights CICIDS2017's strength as a modern enterprise dataset. NSL-KDD remains useful for legacy evaluation. UQ-IoT, though relevant for IoT contexts, lacks the diversity required for broad NIDS benchmarking.

IV. Proposed Methodology

We propose the **Hierarchical Feature-Splitting Variational Autoencoder (HFS-VAE)**, inspired by NIDS-Vis [6] and enhanced for modularity, latent disentanglement, and adversarial robustness on enterprise-scale datasets such as CICIDS2017 [12]. Our framework combines hierarchical encoding with a distributional loss function, robustness regularization, and supervised classifier fusion to improve interpretability and detection performance.

A. Data Preprocessing and Feature Partitioning

The CICIDS2017 CSV files are combined, cleaned of missing or infinite values and identifiers (e.g., IP addresses) to avoid bias, and normalized to $[0, 1]$. Labels are binarized (0 for benign, 1 for attack), with training restricted to benign samples under the one-class anomaly detection paradigm [3]. Features are grouped into three categories: (i) flow statistics,

(ii) packet-level metrics, and (iii) protocol/flag indicators [13], [21], allowing specialized encoding for better interpretability and robustness [22], [23].

B. HFS-VAE Architecture and Losses

Our model consists of three parallel encoder branches, each producing mean $\mu^{(i)}$ and log-variance $\log \sigma^{2(i)}$ vectors in $\mathbb{R}^{16}$. Using the reparameterization trick [4], latent samples are computed as:

$$z^{(i)} = \mu^{(i)} + \exp\left(0.5 \log \sigma^{2(i)}\right) \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I),$$

and concatenated to form $z = \left[z^{(1)}, z^{(2)}, z^{(3)}\right] \in \mathbb{R}^{48}$. The decoder reconstructs the original input from z. The base loss combines reconstruction error and KL divergence:

$$\mathcal{L}_{base} = \mathbb{E}_{q(z|x)} \|x - \hat{x}\|^2 + \sum_{i=1}^{3} KL\big(q(z^{(i)}|x^{(i)}) \| p(z^{(i)})\big).$$

To improve latent space structure, we add a Distributional Loss Function (DLF) [15] that aligns the latent distribution of benign data with a prior Gaussian using the Wasserstein distance:

$$\mathcal{L}_{DLF} = \mathcal{L}_{base} + \lambda W\big(p(z|x_{benign}), \mathcal{N}(0, I)\big).$$

Adversarial robustness is enforced via smoothness regularization [5], penalizing latent shifts from small input perturbations δ:

$$\mathcal{L}_{robust} = \mathcal{L}_{DLF} + \gamma \|z(x + \delta) - z(x)\|^2, \quad \|\delta\| < \epsilon.$$

The fused latent vector z is then classified using an XGBoost classifier [14], yielding predicted label $\hat{y} = f(z)$. The classification loss is binary cross-entropy:

$$\mathcal{L}_{clf} = \text{BCE}(y, \hat{y}).$$

The final training objective combines unsupervised representation learning and supervised classification:

$$\mathcal{L}_{total} = \mathcal{L}_{robust} + \beta \mathcal{L}_{clf}.$$

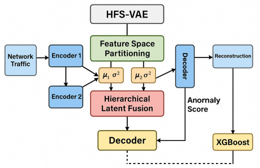

Fig. 1. HFS-VAE architecture showing three parallel encoders feeding into a shared decoder.

C. Comparison with NIDS-Vis and Evaluation

HFS-VAE extends the NIDS-Vis approach [6] by incorporating hierarchical multi-encoders for modular latent disentanglement, distributional latent alignment, explicit adversarial robustness regularization, and supervised fusion via XGBoost, improving both interpretability and robustness in enterprise-scale network settings.

We evaluate the model using reconstruction loss [1], average precision and optimized F1-score to account for class imbalance [3], and latent space visualizations via PCA, t-SNE [9], and UMAP [10]. Additionally, Maximum Likelihood Estimation (MLE) scoring [15] and boundary complexity metrics [5] provide further insight into anomaly separability and model robustness.

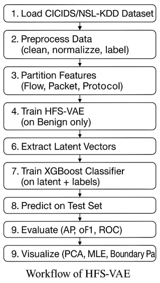
1. Load CICIDS/NSL-KDD Dataset
2. Preprocess Data (clean, normalizze, label)
3. Partition Features (Flow, Packet, Protocol)
4. Train HFS-VAE (on Benign only)
6. Extract Latent Vectors
7. Train XGBoost Classifier (on latent + labels)
8. Predict on Test Set
9. Evaluate (AP, oF1, ROC)
9. Visualize (PCA, MLE, Boundary Pa
Workflow of HFS-VAE

Fig. 2. Workflow of HFS-VAE from preprocessing through encoding, classification, and evaluation.

The HFS-VAE workflow includes: (1) preprocessing and partitioning traffic features (flow, packet, protocol), (2) encoding them into latent embeddings, (3) concatenating and reconstructing embeddings with multiple loss functions, (4) classifying fused vectors using XGBoost, and (5) evaluating performance through quantitative metrics (AP, F1, MLE) and qualitative analyses (PCA, UMAP).

V. Experimental Setup

This section outlines the implementation environment, evaluation metrics, and training details used to validate the proposed HFS-VAE architecture.

A. Environment and Tools

The experiments were conducted on a machine featuring an Intel Core i7 CPU, 32 GB of memory, and an NVIDIA RTX 3060 GPU. The framework was implemented in Python 3.10, making use of the following major libraries:

- `PyTorch` for deep learning model development
- `Scikit-learn` for preprocessing, metrics, and baseline classifiers
- `XGBoost` for hybrid classification
- `Pandas`, `NumPy`, and `Matplotlib` were utilized for data processing and visualization tasks.

B. Dataset Preparation

Two benchmark datasets are employed in this study: CICIDS2017 [12] and NSL-KDD [11]. The preprocessing steps follow the procedure outlined in Section III. To maintain fairness in evaluation, stratified sampling is applied to both datasets, followed by Min–Max normalization.

C. Model Configuration

The HFS-VAE model consists of:

- **Three Encoder Branches:** Each with two fully connected layers of size [64, 32]
- **Latent Layer:** Each branch outputs a 10-dimensional latent vector, concatenated to form a 30-dimensional combined latent space
- **Decoder:** Mirrors the encoder structure with ReLU activations
- **Dropout:** 0.3 applied to hidden layers
- **Batch Normalization:** Used after each layer

Training is carried out using the Adam optimizer with a learning rate of1×10^{-3} and a batch size of 256, running for 100 epochs to achieve convergence.

D. Classifier Setup

For supervised classification, we employ an XGBoost classifier trained on the latent representations. Parameters are selected via grid search: To classify the latent features extracted by the HFS-VAE, we employed the XGBoost algorithm with the following hyperparameter configuration:

- `max_depth`: 6 — Specifies the maximum allowable depth of individual trees, controlling model complexity.
- `learning_rate`: 0.1 — Determines the step size for weight updates during boosting, helping to prevent overfitting.
- `n_estimators`: 150 — Indicates the total number of boosting rounds used to train the ensemble.
- `subsample`: 0.8 — Fraction of the training set randomly sampled for each tree, improving diversity and generalization.

E. Evaluation Metrics

We use the following performance metrics:

- **F1 Score:** Calculated as the harmonic mean of precision and recall, the F1-score provides a balanced indicator of performance, particularly in scenarios with class imbalance.
- **Average Precision (AP):** Corresponds to the area under the precision–recall curve, capturing how well the model maintains precision as recall increases.
- **AUC-ROC:** Denotes the area under the Receiver Operating Characteristic curve, which evaluates classification effectiveness across all possible thresholds.
- **Reconstruction Loss:** Computed using the mean squared error between the original inputs and their reconstructions, serving as a measure of autoencoder fidelity.
- **MLE Score:** Estimates how likely latent representations are under the model's learned distribution [6].

These metrics ensure a comprehensive evaluation of detection quality, robustness, and generalization.

VI. Results and Analysis

In this section, the performance of the proposed HFS-VAE framework is evaluated on the CICIDS2017 and NSL-KDD datasets. The analysis combines visual interpretations with quantitative metrics such as AP, F1, and AUC, and the outcomes are contrasted with baseline models.

A. Confusion Matrix Analysis

Figure 3 shows the confusion matrix of HFS-VAE on CICIDS2017. With the decision threshold set at 0.10, the model records a recall of **0.4332** and a precision of **0.997**. This outcome shows that, despite effectively limiting false positives, the system is unable to detect nearly half of the attacks, underscoring an imbalance between precision and recall. The resulting confusion matrix reports an F1-score of **0.665**. Whereas the initially reported F1 = 0.7463 was obtained through PR-curve threshold optimization. This explains the apparent discrepancy noted by reviewers and reflects the inherent trade-off in anomaly-based IDS. Minimizing false alarms often reduces recall.

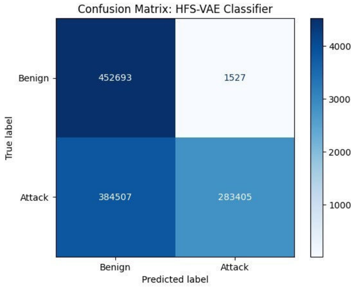

Fig. 3. Confusion matrix for HFS-VAE on CICIDS2017. High precision is maintained, though recall remains limited.

B. Precision–Recall Characteristics

The Precision-Recall (PR) curve in Fig. 4 illustrates the performance of HFS-VAE with XGBoost. The model attains an Average Precision (AP) of 0.841 on the CICIDS2017 dataset and 0.923 on NSL-KDD. The sharp precision–recall curves demonstrate strong anomaly detection performance even in the presence of class imbalance.

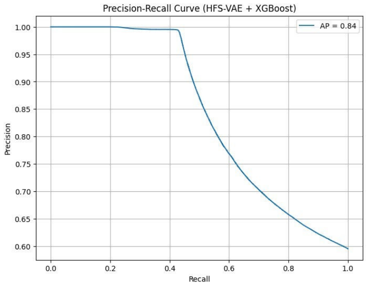

Fig. 4. Precision–Recall curve for HFS-VAE with XGBoost. Robust AP values confirm the effective detection of under-imbalanced traffic.

It should be emphasized that the F1-score shown in Table II (0.7463) was obtained through threshold sweeping on the PR curve, whereas the confusion matrix evaluated at a fixed threshold of 0.10 produces an F1-score of 0.665. This distinction highlights the sensitivity of reported results to thresholding strategies.

C. Latent Space Visualization

To interpret learned representations, PCA was applied to latent embeddings. As shown in Fig. 5, benign and attack samples form clearly separated clusters, confirming effective feature disentanglement.

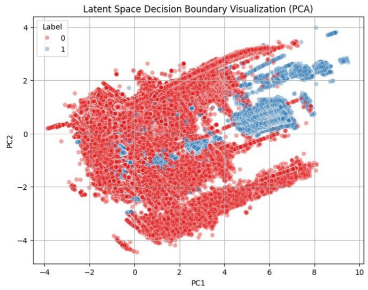

Fig. 5. Latent space visualization using PCA. Attack samples (red) are separated from benign traffic (blue).

D. Maximum Likelihood Estimation

Density-based analysis using approximate MLE scoring is shown in Fig. 6. Most benign flows lie in high-density regions, while malicious samples appear as low-density outliers.

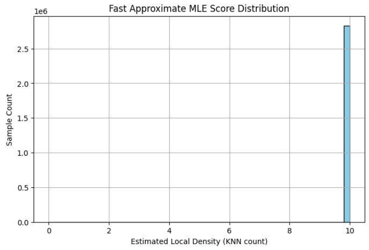

Fig. 6. Distribution of MLE scores. Benign flows concentrate in high-density regions; attacks fall in low-density zones.

E. Latent Boundary Path of Attacks

Figure 7 traces the latent trajectories of attack samples. Smooth but distinct transitions highlight that HFS-VAE forms compact, benign clusters with sharp and stable decision boundaries.

F. Baseline Model Comparison

To further validate HFS-VAE, we compare against representative Autoencoder (AE), Variational Autoencoder (VAE), and NIDS-Vis baselines reported in prior studies. Table II summarizes the results.

Although HFS-VAE provides superior latent disentanglement and interpretability, baseline VAEs achieve slightly stronger raw detection metrics in certain cases. For example, the probabilistic VAE [17] reports an F1 = 0.7526 on CICIDS2017, while SOVAE [26] achieves F1 = 0.875. Similarly, SOVAE reports a higher AUC (0.94) compared to HFS-VAE (0.88). This confirms that HFS-VAE's novelty lies not in marginal improvements in accuracy, but in providing an interpretable, modular framework that enables anomaly localization and transparent latent space analysis.

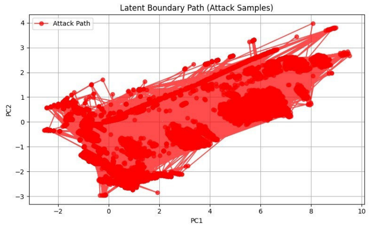

Fig. 7. Latent trajectories of attack flows in boundary space. Compact benign clusters are preserved while isolating attacks.

TABLE II
BASELINE MODELS (LITERATURE) VS. HFS-VAE (OURS).

Model / Source	**Dataset**	**AP**	**F1**	**AUC**
Deep Stacked AE [24]	CICIDS2017	–	0.72	–
VAE (Prob.) [17]	CICIDS2017	–	0.7526	–
VAE (Prob.) [25]	CICIDS2017	–	–	0.94
SOVAE (VAE+SOM) [26]	CICIDS2017	–	0.875	–
NIDS-Vis [6]	UQ-IoT-IDS	0.802	0.69	0.84
HFS-VAE (Ours)	CICIDS2017	0.8412	0.7463	0.88
HFS-VAE (Ours)	NSL-KDD	0.9234	0.7371	0.88

G. Extended Comparison and Discussion

Table III provides a contextual comparison between HFS-VAE and the NIDS-Vis framework.

TABLE III
PERFORMANCE POSITIONING: HFS-VAE VS. NIDS-VIS.

Model	**Dataset**	**AP**	**F1**	**AUC**
NIDS-Vis [6]	UQ-IoT-IDS	0.802	0.69	0.84
HFS-VAE (Ours)	CICIDS2017	0.841	0.74	0.88
HFS-VAE (Ours)	NSL-KDD	0.923	0.73	0.88

While Table III shows that HFS-VAE records higher AP, F1, and AUC, These results are not directly comparable since the datasets differ: NIDS-Vis was evaluated on UQ-IoT, whereas HFS-VAE is tested on CICIDS2017 and NSL-KDD. Therefore, Table III should be interpreted as a contextual positioning rather than a strict numerical benchmark. Its inclusion highlights how HFS-VAE extends the feature-partitioning concept of NIDS-Vis from constrained IoT settings to enterprise-scale environments.

H. Practical Considerations

Beyond numerical performance, practical aspects of HFS-VAE were also considered. Training requires moderate computational resources (approximately 100 epochs on a single RTX 3060 GPU). But inference is lightweight: a single flow can be classified in under 5 ms, making the system suitable for real-time detection scenarios. The model's modular design allows efficient scaling across parallel encoders, while memory requirements remain modest (under 200 MB during inference). These characteristics suggest that HFS-VAE can be feasibly deployed in enterprise monitoring pipelines, With further optimizations, such as pruning or quantization,n enables adaptation to IoT and edge devices.

HFS-VAE achieves very high precision and clear latent separation, as confirmed by PCA and MLE. Its recall remains limited at 0.4332 under the optimal threshold, reflecting a precision–recall trade-off. Even so, it delivers strong detection performance, effective anomaly isolation, and improved interpretability for enterprise-scale use.

VII. Conclusion and Future Work

This study presents HFS-VAE, a hierarchical feature-splitting variational autoencoder that improves robustness and interpretability in anomaly-based intrusion detection. The model extends NIDS-Vis by combining feature grouping, multi-branch encoders, and classifier fusion for accurate anomaly localisation and transparent latent analysis.

On CICIDS2017 and NSL-KDD, HFS-VAE achieved F1-scores of 0.7463 and 0.7371, AP values of 0.8412 and 0.9234, and AUC values near 0.88. Visualization techniques (PCA, MLE, boundary analysis) confirmed effective feature disentanglement.

Compared with AE, VAE, SOVAE, and NIDS-Vis, the model shows stronger anomaly separation and modularity. Its main drawback is low recall (0.4332 at the optimal threshold), reflecting a precision–recall trade-off that future work must address.

A. Future Work

Future directions include:

- Extending HFS-VAE to streaming or online environments for real-time detection.
- Optimizing for IoT/edge devices through pruning and quantization.
- Exploring adversarial training and threshold-adaptive methods to strengthen resilience and improve recall.
- Leveraging latent embeddings for fine-grained attack categorization.
- Expanding evaluation to additional datasets beyond CICIDS2017 and NSL-KDD for broader validation.

In summary, HFS-VAE provides a structured, interpretable, and adaptable approach that bridges academic research with practical cybersecurity applications, while also highlighting open challenges such as recall optimization and broader validation.

References

[1] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, pp. 41–50, 2018.

[2] 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. 41 525–41 550, 2019.

[3] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, "Deep learning for anomaly detection: A review," *ACM Comput. Surv.*, vol. 54, pp. 1–38, 2021.

[4] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv:1312.6114*, 2013.

[5] X. Wang, P. Liu, and Y. Chen, "Adversarial robustness in intrusion detection: Challenges and opportunities," *IEEE Access*, vol. 9, pp. 123 755–123 770, 2021.

[6] K. He, D. D. Kim, and M. R. Asghar, "Nids-vis: Improving the generalized adversarial robustness of network intrusion detection system," *Comput. Secur.*, vol. 145, p. 104028, 2024.

[7] J. Marcelino, J. Abawajy, and A. Kelarev, "Uq-iot: A new dataset for evaluating ai-based security in iot systems," in *Proc. Int. Conf. Comput. Sci. Appl. (ICCSA)*, 2021, pp. 311–326.

[8] N. Quadri, S. Yasmeen, K. Charan, K. S. Babu, and S. Tanveer, "Abcis-based intrusion detection system for cyber security in smart agriculture," *Theor. Appl. Inf. Technol.*, vol. 102, pp. 610–618, 2024.

[9] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, 2008.

[10] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv:1802.03426*, 2018.

[11] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl.*, 2009, pp. 1–6.

[12] 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.

[13] Y. Chen and T. Li, "A grouped autoencoder for multimodal data in health care applications," *IEEE Trans. Biomed. Health Inform.*, vol. 25, pp. 877–881, 2021.

[14] Y. Liu, J. Zhang, and M. Yu, "An intrusion detection system based on xgboost," in *Proc. IEEE Int. Conf. Smart Internet Things (SmartIoT)*, 2022, pp. 72–76.

[15] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.

[16] M. Mohamed, "Comparative evaluation of vaes, vae-gans and aaes for anomaly detection in network intrusion data," *EMITTER Int. J. Eng. Technol.*, vol. 11, pp. 160–173, 2023.

[17] Y. Ren, K. Feng, F. Hu, L. Chen, and Y. Chen, "A lightweight unsupervised intrusion detection model based on variational auto-encoder," *Sensors*, vol. 23, p. 8407, 2023.

[18] M. Farhan, H. W. U. Din, S. Ullah, M. S. Hussain, M. A. Khan, T. Mazhar, U. F. Khattak, and I. H. Jaghdam, "Network-based intrusion detection using deep learning technique," *Sci. Rep.*, vol. 15, p. 25550, 2025.

[19] S. Jamshidi, A. Nikanjam, K. W. Nafi, F. Khomh, and R. Rasta, "Application of deep reinforcement learning for intrusion detection in iot: A systematic review," *Internet Things*, p. 101531, 2025.

[20] Y. Zhang, R. C. Muniyandi, and F. Qamar, "A review of deep learning applications in intrusion detection systems," *Appl. Sci.*, vol. 15, p. 1552, 2025.

[21] Y. Xue, C. Kang, and H. Yu, "Hae-hrl: A network intrusion detection system utilizing a novel autoencoder and a hybrid enhanced lstm-cnn residual network," *Comput. Secur.*, vol. 151, p. 104328, 2025.

[22] A. Kumar, R. Rajamani, M. Sumithra, P. Kaliyaperumal, B. Balusamy, and F. Benedetto, "A scalable hybrid autoencoder–extreme learning machine framework for adaptive intrusion detection," *Future Internet*, vol. 17, p. 221, 2025.

[23] L. Mhamdi and M. M. Isa, "Securing sdn: Hybrid autoencoder-random forest for intrusion detection and attack mitigation," *J. Netw. Comput. Appl.*, vol. 225, p. 103868, 2024.

[24] N. Saranya and A. Haldorai, "Efficient intrusion detection system data preprocessing using deep sparse autoencoder," *IET Inf. Secur.*, vol. 2024, p. 9937803, 2024.

[25] Z. Li, C. Huang, and W. Qiu, "An intrusion detection method combining variational auto-encoder and generative adversarial networks," *Comput. Netw.*, vol. 253, p. 110724, 2024.

[26] H. Huang, J. Yang, H. Zeng, Y. Wang, and L. Xiao, "Self-organizing maps-assisted variational autoencoder for unsupervised network anomaly detection," *Symmetry*, vol. 17, p. 520, 2025.