Improving Robustness and Interpretability of Deep Learning–Based Intrusion Detection in Software-Defined Networks

Hassan Rizwan Malik*, Muhammad Salman*, Zaeem Ahmed*, Nabeelah Maryam*
* Department of Computer Science
FAST National University of Computer and Emerging Sciences
Islamabad, Pakistan

***Abstract*—Recent deep learning–based intrusion detection systems (IDS) for Software-Defined Networks (SDNs) report near-perfect accuracy on benchmark datasets. However, such results are often obtained using single train–test splits, manual feature reduction, and accuracy-dominated evaluation, which may overestimate real-world performance. In this paper, we revisit a recent CNN–LSTM and Transformer-based IDS proposed for the InSDN dataset [1] and systematically address its methodological limitations. We introduce an enhanced pipeline incorporating Recursive Feature Elimination (RFE) with SHAP-based interpretability [2], [3], bidirectional sequence modeling [4], imbalance-aware loss functions [5], [6], and stratified cross-validation. Experiments conducted using stratified 5-fold cross-validation reveal an average accuracy of 81.60%, weighted F1-score of 79.52%, and macro AUC of 84.00%, exposing significant variance across folds. While headline accuracy is lower than previously reported single-split results [1], our findings provide a more realistic and robust assessment of IDS performance under severe class imbalance. Additionally, we demonstrate that increased architectural complexity, such as attention-based CNN–LSTM models, does not necessarily improve performance. The proposed framework emphasizes generalization, interpretability, and evaluation rigor, making it more suitable for deployment in operational SDN environments.**

***Index Terms*—Software-Defined Networks, Intrusion Detection Systems, Deep Learning, BiLSTM, Feature Selection, Class Imbalance**

I. Introduction

Software-Defined Networking (SDN) has transformed network management by decoupling control and data planes, enabling centralized programmability and flexible policy enforcement. Despite these advantages, the centralized SDN controller introduces new attack surfaces, making intrusion detection a critical component of SDN security [7].

Recent studies have applied deep learning (DL) techniques such as CNN–LSTM hybrids and Transformers to SDN intrusion detection, reporting accuracies exceeding 99% on some benchmarks [1], [8]. Transformer-based hybrids for DDoS detection and robust Transformer variants have also shown strong reported performance on domain datasets [9], [10]. While promising, such results often rely on single train–test splits, manual feature reduction, and accuracy-centric evaluation metrics, which may obscure performance degradation under class imbalance and distributional shifts [11].

This work revisits these claims by reproducing the problem under a more rigorous experimental framework. We aim to improve robustness, interpretability, and evaluation reliability rather than maximizing headline accuracy.

II. Related Work

Deep learning approaches have been widely explored for intrusion detection in SDN and flow-based network environments. Table I provides a comparative overview of representative works in this domain, highlighting differences in datasets, methodologies, and evaluation strategies.

Ataa et al. [1] introduced the InSDN dataset and proposed CNN–LSTM and Transformer-based models, reporting near-perfect accuracy. However, their evaluation relied on single train–test splits with manual feature reduction and class merging to address imbalance. Tang et al. [11] applied GRU-RNN architectures to NSL-KDD and CICIDS2017, achieving high reported accuracy but similarly using single-split evaluation. Elsayed et al. [12] explored hybrid CNN–LSTM models on InSDN and other flow-based datasets, reporting strong performance but with limited discussion of evaluation robustness.

Recent Transformer-based approaches have also shown promise. Wang and Li [9] proposed DDosTC, a Transformer-CNN hybrid for DDoS detection on CICDDoS2019, achieving high detection rates. Wu et al. [10] introduced RTIDS, a robust Transformer variant focusing on representation stability across CICIDS2017 and CICDDoS2019. While these works demonstrate the potential of attention mechanisms, they primarily target DDoS-specific datasets and do not extensively address the interpretability and class imbalance challenges present in multi-class SDN intrusion detection.

Our work differs from prior approaches by emphasizing methodological rigor and interpretability. We employ principled feature selection via RFE [2] validated with SHAP interpretability [3], adopt stratified 5-fold cross-validation for robust performance estimation, and apply imbalance-aware loss functions [5], [6]. Unlike single-split evaluations common in the literature [1], [9]–[11], our approach provides variance analysis across folds, revealing realistic performance bounds under operational conditions.

TABLE I
COMPARISON WITH REPRESENTATIVE SDN / FLOW-BASED IDS WORKS

Work	Year	Dataset(s)	Methods	Reported / Key Results
Ataa et al. [1] (base paper)	2024	InSDN (48 features)	CNN–LSTM, Transformer	Reported ∼99% accuracy (single split); minority classes merged/upsampled
Tang et al. [11]	2019	NSL-KDD, CICIDS2017	GRU-RNN	Reported 89% (NSL-KDD) and 99% (CICIDS2017) on reported splits
Elsayed et al. [12]	2021	InSDN / other flow sets	Hybrid CNN–LSTM	High accuracy reported (e.g., ∼96% on InSDN in prior reports)
Wang & Li (DDosTC) [9]	2021	CICDDoS2019	Transformer + CNN hybrid	Strong reported detection for DDoS (high accuracy / F1 on CICDDoS2019)
Wu et al. (RTIDS) [10]	2022	CICIDS2017, CICDDoS2019	Transformer-based (robust variant)	High F1 reported on tested datasets; focuses on representation robustness
This work (Ours)	2025	InSDN (48 features → RFE→32)	RFE + SHAP; BiLSTM; Focal / Class-Balanced Loss; 5-fold CV	Average Accuracy = **81.60%**; Weighted F1 = **79.52%**; Macro AUC = **84.00%**

III. LIMITATIONS OF PRIOR WORK

The base study motivating this work suffers from several significant methodological limitations that may overestimate model performance and reduce practical applicability [1].

First, the feature reduction process employed manual and heuristic selection strategies without principled justification or systematic evaluation [1]. The original 48-feature InSDN dataset was reduced through ad-hoc feature dropping based on informal correlation analysis and domain intuition, rather than using established feature selection algorithms such as Recursive Feature Elimination or information-theoretic measures. This approach lacks reproducibility and provides no interpretability regarding which features contribute most to detection decisions, making it difficult to understand model behavior or validate feature importance across different attack scenarios.

Second, the severe class imbalance inherent in the InSDN dataset was addressed primarily through class merging and synthetic upsampling techniques [1]. While these methods can improve numerical balance, they introduce fundamental problems. Class merging combines distinct attack types into broader categories, potentially obscuring important behavioral differences between attack variants and reducing the granularity of threat detection. Synthetic oversampling methods like SMOTE may generate unrealistic samples that do not reflect true network traffic patterns, leading to models that perform well on synthetic data but fail on real-world distributions. Neither approach addresses the underlying difficulty of learning robust decision boundaries for minority classes.

Third, model evaluation was conducted using a single train–test split without variance analysis or cross-validation [1]. This evaluation strategy cannot assess model stability across different data partitions and may produce misleadingly high performance metrics if the particular split happens to be favorable. Single-split evaluation is especially problematic for imbalanced datasets where random sampling can significantly affect class distributions in train and test sets, leading to unreliable performance estimates that do not generalize to operational deployments.

Fourth, performance assessment relied heavily on accuracy and micro-averaged metrics [1], which are known to be misleading for imbalanced classification tasks. Accuracy gives equal weight to all samples regardless of class, causing majority-class performance to dominate the metric while minority-class detection failures are masked. Micro-averaging similarly aggregates predictions across all samples, making it insensitive to per-class performance disparities. For intrusion detection, where rare attack types may be the most critical to detect, these metrics fail to provide meaningful insight into operational effectiveness.

Fifth, the published work provided no analysis of feature importance or model decision processes [1], limiting interpretability and trustworthiness. Understanding which features drive classification decisions is crucial for validating that models learn genuine attack signatures rather than spurious correlations or dataset artifacts. Without interpretability mechanisms such as SHAP values, attention weights, or feature attribution analysis, it is impossible to verify whether the model's internal representations align with domain knowledge or to diagnose failure modes when misclassifications occur.

These limitations collectively motivate the need for a more robust, interpretable, and transparent IDS evaluation pipeline that provides realistic performance estimates suitable for operational deployment decisions.

IV. PROPOSED METHODOLOGY

Our methodology addresses the limitations identified in prior work through a comprehensive pipeline that emphasizes principled feature selection, architectural sophistication, imbalance-aware training, and rigorous evaluation. The

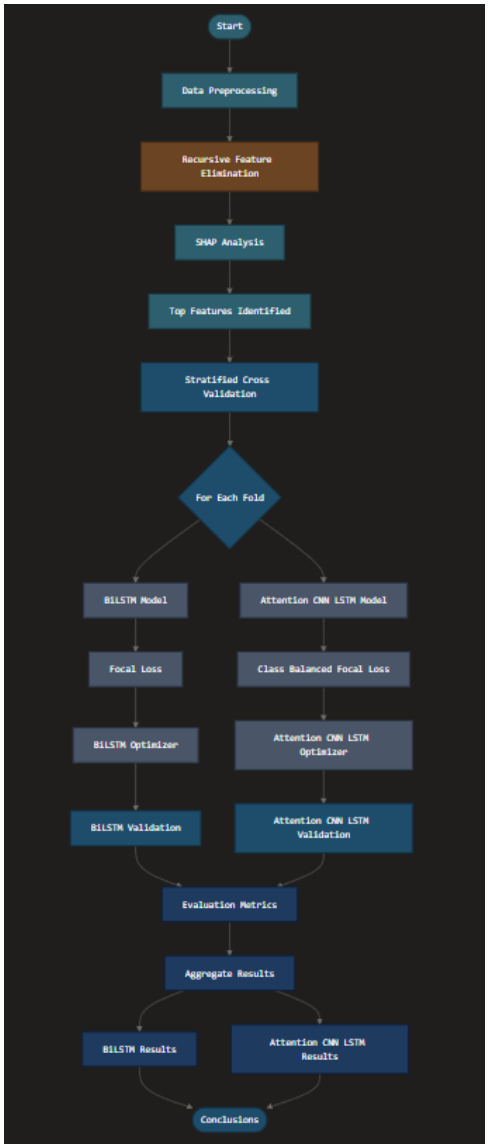

Fig. 1. Overview of the proposed methodology pipeline showing three main stages: (1) data preprocessing and analysis with RFE and SHAP, (2) model architecture and training with BiLSTM and Attention CNN-LSTM variants, and (3) evaluation through cross-validation with comprehensive metrics.

pipeline consists of four major components that work synergistically to produce a robust and interpretable intrusion detection system.

Figure 1 illustrates the complete workflow of our proposed pipeline, divided into three distinct stages that address the key limitations of prior work. The first stage encompasses data preprocessing and feature analysis, the second stage covers model architecture and training methodology, and the third stage details our comprehensive evaluation framework.

A. *Feature Selection and Interpretability*

Feature selection is critical for reducing dimensionality, improving model generalization, and enhancing computational efficiency. We employ a two-stage approach combining automated feature selection with post-hoc interpretability analysis.

In the first stage, we apply Recursive Feature Elimination (RFE) using a Random Forest estimator as the base learner [2]. RFE operates by iteratively training the model on the current feature set, ranking features by their importance scores, and eliminating the least important features in each iteration. This process continues until a predetermined number of features remains or performance degradation is detected. Unlike manual feature selection, RFE provides a systematic and reproducible mechanism for identifying informative features. For our experiments, we configured RFE to select 32 features from the original 48-feature InSDN dataset, balancing model complexity with representation capacity. The Random Forest estimator was configured with 300 trees, unlimited maximum depth, balanced subsample class weighting, and parallel processing across all available CPU cores. This configuration was chosen for its robustness to overfitting, ability to capture non-linear feature interactions, and native support for feature importance quantification through mean decrease in impurity.

In the second stage, we validate and interpret the selected features using SHAP (Shapley Additive Explanations) [3]. SHAP provides a unified framework for interpreting model predictions by computing the contribution of each feature to individual predictions based on cooperative game theory principles. Specifically, SHAP values represent the marginal contribution of each feature across all possible feature coalitions, satisfying desirable properties such as local accuracy, missingness, and consistency. We compute global feature importance by aggregating the mean absolute SHAP values across a background sample of 2,000 training instances using the TreeExplainer method, which efficiently computes exact SHAP values for tree-based models. This approach provides insight into which features consistently drive classification decisions across the dataset. Table II presents the top 15 features ranked by their mean absolute SHAP values, revealing which features contribute most significantly to model predictions.

TABLE II
TOP 15 SHAP-RANKED FEATURES

Rank	Feature Name	Mean Abs SHAP
1	Bwd Header Len	0.0432
2	Protocol	0.0277
3	Fwd Header Len	0.0233
4	Fwd Pkts/s	0.0217
5	Bwd Pkts/s	0.0116
6	Flow Duration	0.0109
7	Flow IAT Max	0.0107
8	Tot Fwd Pkts	0.0106
9	Flow Pkts/s	0.0096
10	Flow IAT Min	0.0094
11	Bwd IAT Min	0.0080
12	Flow IAT Mean	0.0072
13	Flow IAT Std	0.0066
14	Fwd IAT Max	0.0065
15	Pkt Size Avg	0.0059

The SHAP analysis reveals that backward header length

emerges as the single most discriminative feature with a mean absolute contribution of 0.0432, followed by protocol type (0.0277) and forward header length (0.0233). These findings align with domain knowledge, as header characteristics and protocol information are fundamental indicators of network behavior and potential anomalies. Temporal features such as packet rates, flow duration, and inter-arrival time statistics also rank highly, confirming the importance of temporal patterns in distinguishing attack traffic from benign flows. This interpretability layer not only validates the feature selection process but also ensures that model decisions align with domain knowledge of network intrusion patterns, increasing trust and facilitating debugging when misclassifications occur.

B. Deep Learning Architectures

We explore two distinct architectural paradigms to evaluate the trade-offs between model complexity, learning capacity, and robustness under severe class imbalance.

1) Bidirectional LSTM: The primary architecture is a Bidirectional Long Short-Term Memory (BiLSTM) network designed to capture temporal dependencies in network flow sequences [4]. Unlike standard unidirectional LSTMs that process sequences in a single temporal direction, BiLSTM processes input sequences both forward and backward, allowing the model to leverage both past and future context when making predictions at each time step. This bidirectional processing is particularly valuable for intrusion detection where attack signatures may manifest as patterns distributed across multiple time steps, and understanding the full temporal context improves discrimination between benign and malicious flows.

Our BiLSTM architecture consists of two stacked bidirectional LSTM layers with hidden dimensions of 128 units each, processing the 32 selected features as a sequence. The bidirectional design effectively doubles the representational capacity at each layer by concatenating forward and backward hidden states. Dropout regularization with a rate of 0.3 is applied between LSTM layers to prevent overfitting. The LSTM outputs are averaged across the temporal dimension and passed through batch normalization, followed by a fully connected layer with 128 units and ReLU activation. An additional dropout layer precedes the final classification head. This architecture balances expressive power with computational efficiency, capturing complex temporal patterns without excessive parameterization.

2) Attention-based CNN–LSTM: To investigate whether increased architectural complexity improves detection performance, we implemented an attention-enhanced CNN–LSTM hybrid model. This architecture combines convolutional layers for local feature extraction, bidirectional LSTM layers for temporal modeling, and self-attention mechanisms for adaptive feature weighting [13]. The convolutional component consists of two sequential 1D convolutional layers with batch normalization and ReLU activation. The first layer uses 64 channels and the second uses 128 channels, both with kernel size 3 and padding to preserve sequence length. These convolutional features are then fed into a single-layer bidirectional LSTM with 128 hidden units that models temporal dependencies. A self-attention mechanism computes attention weights over the LSTM hidden states through a two-layer projection network with tanh activation, allowing the model to dynamically focus on the most relevant time steps for each prediction. The attention-weighted context vector is passed through fully connected layers with 128 units, ReLU activation, and 0.3 dropout before the final classification layer.

Despite its theoretical expressiveness and motivation from successful attention-based architectures in other domains, this model exhibited catastrophic performance degradation in our experiments, achieving an average accuracy of only 4.23% and F1-score of 1.32%. We attribute this failure to several factors related to severe class imbalance and over-parameterization. The increased number of learnable parameters amplifies the model's tendency to overfit to majority classes while failing to learn meaningful representations for minority classes. The attention mechanism, rather than providing useful feature selection, appears to collapse during training, possibly due to unstable gradient flow or inadequate training signal from underrepresented classes. This negative result provides an important empirical lesson: architectural complexity alone does not guarantee improved performance, and simpler models with appropriate inductive biases may be more robust under challenging data conditions.

C. Handling Class Imbalance

The InSDN dataset exhibits severe class imbalance, with some attack types represented by orders of magnitude fewer samples than normal traffic or common attack classes. Traditional cross-entropy loss treats all samples equally, causing models to bias heavily toward majority classes while achieving poor performance on rare but potentially critical attack types. To address this fundamental challenge, we employ imbalance-aware loss functions that dynamically reweight training contributions based on class difficulty and frequency.

We implement two complementary loss formulations. First, Focal Loss [5] modifies the standard cross-entropy loss by introducing a modulating factor $(1 - p_t)^\gamma$ where p_t is the predicted probability of the true class and γ is a focusing parameter. This modulating factor down-weights the contribution of easily classified samples (high p_t) while emphasizing hard-to-classify samples (low p_t). By setting $\gamma = 2$ in our BiLSTM experiments, the loss adaptively focuses learning on challenging examples, preventing the model from being dominated by easy majority-class samples. Additionally, Focal Loss includes a class-balancing factor α_t derived from class frequencies that provides inverse frequency weighting to further compensate for class imbalance.

Second, we employ Class-Balanced Focal Loss [6], which extends Focal Loss by introducing a more sophisticated class reweighting scheme based on effective sample counts. Rather than using simple inverse class frequencies, Class-Balanced Loss computes weights as $(1 - \beta)/(1 - \beta^{n_i})$ where n_i is the number of samples in class i and $\beta \in [0, 1)$ controls

the reweighting strength. This formulation accounts for the diminishing marginal benefit of additional samples as class size increases, providing a theoretically grounded approach to balancing between underrepresented and well-represented classes. We set $\beta = 0.999$ and $\gamma = 1.5$ for the attention-based model experiments to achieve aggressive reweighting that prioritizes minority classes without completely ignoring majority classes.

By combining these loss functions, our training procedure explicitly addresses class imbalance at the optimization level, encouraging the model to learn discriminative features for all attack types rather than trivially predicting majority classes.

D. Evaluation Strategy

To ensure reliable and generalizable performance assessment, we adopt a rigorous evaluation protocol based on Stratified 5-Fold Cross-Validation. Unlike single train-test splits that provide only a point estimate of performance and may be sensitive to random partitioning, cross-validation evaluates model performance across multiple independent data splits, providing both mean performance and variance estimates that better reflect expected behavior on unseen data.

Stratification is critical for imbalanced datasets, as it ensures that each fold maintains approximately the same class distribution as the original dataset. Without stratification, random splits could produce folds with severely skewed class distributions or even missing classes, leading to unreliable or undefined performance metrics. Our stratification procedure divides the dataset into five folds such that each fold preserves the relative proportions of all attack types and benign traffic. For each iteration, four folds are used for training (with further subdivision into training and validation sets for hyperparameter tuning) while the remaining fold serves as the test set. This process is repeated five times with each fold serving exactly once as the test set, yielding five independent performance measurements.

We evaluate model performance using a comprehensive set of metrics that provide complementary perspectives on classification quality. Accuracy measures overall correctness but can be misleading under imbalance. Weighted precision, recall, and F1-score account for class imbalance by computing per-class metrics and averaging them weighted by class support, providing a more balanced view of performance across all classes. Macro ROC-AUC computes the area under the receiver operating characteristic curve for each class independently and averages them without weighting, offering insight into the model's ability to discriminate between classes regardless of their frequency. This metric is particularly valuable for assessing minority-class detection capability, which is often the most critical aspect of intrusion detection systems.

Formally, for a dataset with C classes and N samples, we define the following metrics. Let TP_i, TN_i, FP_i, and FN_i denote true positives, true negatives, false positives, and false negatives for class i, respectively, and let n_i denote the number of samples in class i.

Accuracy measures the proportion of correctly classified samples:

$$\text{Accuracy} = \frac{\sum_{i=1}^{C} TP_i}{N} \tag{1}$$

Precision for class i measures the proportion of predicted positives that are truly positive:

$$\text{Precision}_i = \frac{TP_i}{TP_i + FP_i} \tag{2}$$

Recall for class i measures the proportion of actual positives correctly identified:

$$\text{Recall}_i = \frac{TP_i}{TP_i + FN_i} \tag{3}$$

F1-score for class i is the harmonic mean of precision and recall:

$$\text{F1}_i = 2 \cdot \frac{\text{Precision}_i \cdot \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i} \tag{4}$$

Weighted averaging computes the overall metric as a weighted average across all classes:

$$\text{Metric}_{\text{weighted}} = \frac{\sum_{i=1}^{C} n_i \cdot \text{Metric}_i}{N} \tag{5}$$

Macro ROC-AUC computes the area under the receiver operating characteristic curve for each class using one-vs-rest strategy, then averages without weighting:

$$\text{AUC}_{\text{macro}} = \frac{1}{C} \sum_{i=1}^{C} \text{AUC}_i \tag{6}$$

The weighted metrics give more importance to classes with larger support, making them suitable for overall system performance assessment. In contrast, macro AUC treats all classes equally regardless of frequency, making it particularly valuable for evaluating minority-class detection capability in imbalanced scenarios where rare attack types may be the most critical to detect.

Additionally, we generate detailed confusion matrices and per-class classification reports for qualitative analysis, allowing us to identify which specific attack types are well-detected versus which remain challenging. This comprehensive evaluation framework provides transparency into model strengths and weaknesses, supporting informed decisions about deployment readiness and areas requiring further improvement.

V. Experimental Setup

Experiments were conducted on the InSDN dataset using the 48-feature CSV snapshots [1], comprising 343,889 flow records spanning eight distinct traffic classes with severe imbalance. The class distribution exhibits orders-of-magnitude variation, with the largest class containing 121,942 samples while the smallest contains only 17 samples, resulting in an imbalance ratio exceeding 7,000:1. Data preprocessing included label encoding of attack categories and standard scaling of all numerical features to zero mean and unit variance to ensure stable gradient-based optimization.

TABLE III
BiLSTM Performance with RFE + SHAP (5-Fold CV)

Metric	Average Value
Accuracy	81.60%
Precision (Weighted)	80.68%
Recall (Weighted)	81.60%
F1-score (Weighted)	79.52%
Macro ROC–AUC	84.00%

Feature selection via RFE with Random Forest (300 estimators, balanced subsample class weighting, random state 42) reduced the feature space from 48 to 32 features prior to cross-validation, achieving a 33% dimensionality reduction while retaining the most discriminative flow characteristics. The RFE process required approximately 209 seconds of computation time. SHAP analysis was performed on a randomly sampled background set of 2,000 training instances using TreeExplainer to compute global feature importance scores, requiring approximately 848 seconds. These interpretability computations represent a one-time preprocessing cost that provides valuable insights for model validation and deployment.

Models were trained using PyTorch 1.x with the Adam optimizer configured with learning rate 10^{-3}, weight decay 10^{-4} for L2 regularization, and gradient clipping with maximum norm 5.0 to prevent exploding gradients during backpropagation through LSTM layers. Training spanned 30 epochs per fold for both architectures, with a batch size of 256 samples. The BiLSTM model used Focal Loss with focusing parameter $\gamma = 2$ and class-balanced weighting derived from training set frequencies. The Attention CNN-LSTM used Class-Balanced Focal Loss with effective sample reweighting parameter $\beta = 0.999$ and focusing parameter $\gamma = 1.5$ to provide more aggressive minority class emphasis. Model selection within each fold was based on validation weighted F1-score, with the best-performing checkpoint saved and loaded for final test set evaluation. All experiments utilized GPU acceleration (CUDA-enabled) when available, falling back to CPU computation otherwise. The complete BiLSTM training process across all five folds required approximately 2,075 seconds, while the Attention CNN-LSTM required 1,442 seconds despite its catastrophic performance, demonstrating that training efficiency does not correlate with model effectiveness under severe imbalance.

VI. Results

A. BiLSTM Performance

Table III summarizes the cross-validation performance of the BiLSTM model.

Performance varied across folds, with accuracy ranging from 73.80% to 84.87%, revealing model sensitivity to data splits.

B. Comparative Results with Base Paper

Table IV compares the performance metrics between the base paper and our work.

TABLE IV
Performance Comparison: Base Paper vs. Our Work

Metric	**Base Paper [1]**	**Our Work**
Evaluation Method	Single split	5-Fold CV
Accuracy	∼99%	81.60% (73.80–84.87%)
Precision (Weighted)	Not reported	80.68%
Recall (Weighted)	Not reported	81.60%
F1-score (Weighted)	Not reported	79.52%
Macro ROC-AUC	Not reported	84.00%
Performance Variance	Not assessed	11.07% range

Table IV reveals a significant gap between the base paper's reported ∼99% accuracy using single-split evaluation and our 81.60% average accuracy obtained through stratified 5-fold cross-validation. The performance variance across folds (73.80% to 84.87%) demonstrates the importance of rigorous evaluation methodology. While the base paper reported only accuracy, our comprehensive evaluation includes weighted precision, recall, F1-score, and macro ROC-AUC, providing a more complete assessment of model performance under severe class imbalance.

C. Attention CNN–LSTM Failure

The attention-based CNN–LSTM model achieved an average accuracy of 4.23% and F1-score of 1.32%, indicating failure to learn meaningful decision boundaries. This result underscores that architectural complexity alone does not guarantee improved IDS performance [10].

VII. Discussion

Compared to prior work reporting over 99% accuracy under single-split evaluations [1], our results present a more conservative but realistic evaluation. The observed performance drop highlights the impact of class imbalance, evaluation methodology, and feature selection on IDS effectiveness. While majority attack classes are detected reliably, minority classes remain challenging, suggesting the need for hierarchical or multi-stage detection strategies and/or ensemble methods.

The significant variance observed across folds (accuracy ranging from 73.80% to 84.87%) provides crucial insight that single-split evaluation would miss entirely. Fold 4 exhibited substantially degraded performance across all metrics, suggesting that certain data partitions contain particularly challenging combinations of minority-class samples or distributional characteristics that stress the model's generalization capabilities. This variance underscores the importance of cross-validation for establishing confidence intervals on expected performance rather than relying on potentially optimistic single-split point estimates. In operational deployment scenarios, such variance analysis helps establish realistic service-level expectations and identify when additional data collection or model retraining may be necessary.

The comparative analysis in Table I illustrates how methodological choices significantly impact reported performance. Works relying on single splits or class merging tend to report higher accuracy but may not reflect true generalization capability. Our stratified cross-validation approach, while yielding lower headline metrics, provides a more trustworthy estimate of real-world IDS performance. The interpretability provided by SHAP analysis further distinguishes our approach by enabling security analysts to validate that detection decisions are driven by meaningful network flow characteristics rather than spurious statistical artifacts. The dominance of header length features and protocol information in SHAP rankings confirms that the model has learned domain-appropriate representations.

The catastrophic failure of the Attention CNN-LSTM architecture, despite its theoretical sophistication and successful application in other domains, provides an important negative result for the research community. This failure illustrates that under extreme class imbalance (¿7,000:1 ratio), increased model capacity can actually harm performance by providing more opportunities for overfitting to majority classes while failing to learn from scarce minority-class examples. The attention mechanism's collapse suggests that gradient-based training of complex architectures requires sufficient training signal from all classes, a condition violated in severely imbalanced scenarios. This finding supports the principle that architectural choices must be justified not only by capacity arguments but also by robustness to data distribution characteristics. Future work exploring architectural solutions should prioritize mechanisms that explicitly constrain or regularize model behavior under imbalance, such as prototypical networks, meta-learning approaches, or hierarchical classification strategies that decompose the problem into more balanced subtasks.

VIII. Conclusion

This paper presents a robust and interpretable deep learning pipeline for SDN intrusion detection that prioritizes methodological rigor over headline performance metrics. By integrating principled feature selection via RFE, SHAP-based interpretability analysis, imbalance-aware loss functions, bidirectional LSTM modeling, and stratified cross-validation, we provide a realistic assessment of IDS performance under challenging real-world conditions. Our findings reveal that previously reported near-perfect accuracy substantially overestimates generalization capability when proper evaluation protocols are applied, with our BiLSTM model achieving 81.60% average accuracy across five folds compared to the 99% reported under single-split evaluation of the base work.

The comprehensive evaluation framework demonstrates significant performance variance across data partitions (73.80% to 84.87% accuracy), highlighting the necessity of cross-validation for establishing reliable confidence intervals rather than optimistic point estimates. The SHAP interpretability analysis validates that model decisions are driven by domain-appropriate features such as header characteristics, protocol information, and temporal flow statistics, increasing trust and enabling security analysts to reason about detection logic. Critically, we demonstrate through the catastrophic failure of the attention-based CNN-LSTM architecture that increased architectural complexity does not guarantee improved performance under severe class imbalance, providing an important negative result for the research community.

Future work will explore several promising directions to address the remaining challenges in minority-class detection. Ensemble methods combining multiple specialized models may provide complementary strengths across different attack types. Hierarchical classification strategies could decompose the multi-class problem into more balanced binary or few-class sub-problems that are easier to learn robustly. Cost-sensitive learning approaches that explicitly assign different misclassification penalties to rare attack types could further improve minority-class recall. Meta-learning and few-shot learning techniques may enable better generalization from limited minority-class examples. Additionally, investigation of data augmentation strategies specifically designed for network flow data, such as topology-aware synthetic sample generation, could address imbalance at the data level rather than purely through algorithmic modifications. Finally, deployment considerations including real-time inference efficiency, model update strategies, and integration with existing SDN security frameworks represent important practical extensions of this research.

Acknowledgment

This work was conducted as part of the Computer Networks Lab course at FAST National University of Computer and Emerging Sciences.

References

[1] M. S. Ataa, "Intrusion detection in software defined network using deep learning approaches," *Scientific Reports*, vol. 14, p. 29159, 2024.

[2] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46, no. 1, pp. 389–422, 2002.

[3] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[4] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[5] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 318–327, 2020.

[6] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 9268–9277.

[7] D. Kreutz, F. M. V. Ramos, and P. E. Verissimo, "Towards secure and dependable software-defined networks," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 44–51, 2013.

[8] J. Zhang, M. Chen, Y. Chen, and T. Li, "A deep learning-based intrusion detection system for software defined networks," *IEEE Access*, vol. 7, pp. 146 960–146 971, 2019.

[9] H. Wang and W. Li, "Ddostc: A transformer-based network attack detection hybrid mechanism in sdn," *Sensors*, vol. 21, no. 15, p. 5047, 2021.

[10] Z. Wu, H. Zhang, P. Wang, and Z. Sun, "Rtids: A robust transformer-based approach for intrusion detection system," *IEEE Access*, vol. 10, pp. 64 375–64 387, 2022.

[11] T. A. Tang, L. Mhamdi, and D. McLernon, "Deep recurrent neural network for intrusion detection in sdn-based networks," *4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, 2019.
[12] M. S. Elsayed, N.-A. Le-Khac, and A. D. Jurcut, "A hybrid cnn–lstm based approach for anomaly detection systems in sdns," *Computers & Security*, vol. xx, pp. xx–xx, 2021.
[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.