

Evaluating the Robustness of Machine Learning-Based Network Intrusion Detection Against Feature-Space Adversarial Evasion: An InSDN Study

Furqan Shaikh Mohammed
Junaid¹ Aarti Sanjay Gawai²

¹Student, Dept. of Computer Science, D.G Ruparel College of Arts,
Science and Commerce, Mumbai, Maharashtra, India

²Dept. of Information Technology and Computer Science, D.G Ruparel College of Arts,
Science and Commerce, Mumbai, Maharashtra, India

furkhanshaikh66@gmail.com, aartigawai1611@gmail.com

ABSTRACT-Machine learning-based network intrusion detection systems often perform extremely well on standard test datasets. However, high accuracy on clean data does not necessarily mean that these models can withstand carefully designed input manipulation. In this study, we examine the robustness of three widely used classifiers, namely Random Forest, XGBoost, and a multilayer perceptron (MLP), using the InSDN dataset. The problem is treated as a binary classification task in which network traffic is categorized as either Normal or Attack.

Before training the models, duplicate records were removed, while identifier and timestamp fields were excluded from the dataset. The remaining data was processed using training-only preprocessing, and the 40 features with the highest mutual information scores were selected for the experiments. The three classifiers were initially tested using an untouched test set to establish their performance under normal conditions. They were then evaluated against a controlled feature-space adversarial evasion procedure, where bounded directional perturbations were applied using normalized budgets of $\epsilon \in \{0.01, 0.025, 0.05, 0.10\}$.

The models achieved almost perfect results on the clean test set, with F1 scores of 99.9927% for Random Forest, 99.9952% for XGBoost, and 99.9855% for MLP. However, their performance changed considerably when adversarial perturbations were introduced. For example, with a positive perturbation direction at $\epsilon=0.01$, Random Forest recorded an attack success rate (ASR) of 94.09%. XGBoost reached an ASR of 93.15% at $\epsilon=0.05$, while MLP reached 87.78% at the same perturbation budget. At $\epsilon=0.10$ in the positive direction, both Random Forest and MLP reached an ASR of 100%.

Keywords: Software-Defined Networking (SDN), Intrusion Detection Systems (IDS), Machine Learning, Adversarial Evasion, Feature-Space Perturbation, InSDN Dataset

I. INTRODUCTION

Software-Defined Networking (SDN) separates network control from packet forwarding, making networks more programmable and easier to manage centrally. [1][3] These features have also encouraged the use of machine-learning-based intrusion detection systems for monitoring SDN traffic. The InSDN dataset was developed for this purpose and contains both normal and malicious network traffic [2].

Machine learning can detect traffic patterns without relying entirely on manually defined rules, and different models can learn

different types of relationships within network features. ML and deep-learning methods have therefore been widely studied for DDoS detection in SDN environments [6][7]. However, good performance on clean test data does not necessarily mean that a detector is robust against deliberately modified inputs. Previous studies have shown that intrusion detection systems, including those used in SDN, can be affected by adversarial manipulation [9]–[14].

This study examines this issue through a controlled feature-space adversarial evaluation. Correctly detected malicious samples are modified using bounded directional perturbations and then classified again to measure how often they evade detection. This follows the broader concept of test-time evasion, where an attacker modifies an input while keeping the change constrained [17].

Three different classifiers are evaluated: Random Forest (RF), XGBoost, and a multilayer perceptron (MLP). RF uses an ensemble of decision trees [21], while XGBoost uses gradient boosting [22]. MLP provides a neural-network-based comparison. Using these different model types allows the study to examine whether adversarial sensitivity varies across different learning architectures.

Multiple perturbation budgets and both perturbation directions are tested. This allows the study to examine how evasion changes with perturbation strength and direction. Defining the perturbation size and its constraints is an important part of adversarial machine-learning evaluation [18] [20].

A. Research Questions

The study addresses three research questions:

- **RQ1:** How vulnerable are ML-based network intrusion detectors to constrained feature-space adversarial evasion?
- **RQ2:** How does the perturbation budget affect evasion success?
- **RQ3:** How does perturbation direction affect evasion across different model architectures?
-

The main contribution of this work is an empirical assessment of detector vulnerability rather than a new attack or defense algorithm. The experiments examine whether models with near-perfect clean-test performance remain reliable under controlled adversarial perturbations. The findings emphasize the need to evaluate adversarial robustness alongside conventional metrics such as accuracy, precision, recall, and F1 score [12] [13].

II. LITERATURE REVIEW

A. ML-Based Intrusion Detection in SDN

Machine-learning-based intrusion detection uses network traffic features to distinguish normal and malicious activity. The InSDN dataset was specifically developed for evaluating intrusion detection in SDN environments and has been used with different machine-learning approaches^[2]. In this study, RF, XGBoost, and MLP are used to compare tree-based and neural-network classifiers^{[21] [22]}.

B. Adversarial Evasion of Network Detectors

Adversarial evasion involves modifying an input during testing to cause a trained model to make an incorrect prediction. Previous work has shown that ML-based intrusion detectors, including those used in SDN environments, can be affected by such manipulation^{[10], [12] –[14]}. Feature-space attacks are useful for controlled robustness testing, although modified feature values do not necessarily represent directly executable packet-level attacks.

C. Adversarial Training and Defensive Studies

Previous studies have also explored defenses against adversarial attacks. Nugraha et al. investigated adversarial training for DDoS detection in SDN^[11], while Novaes et al. studied adversarial deep learning for DDoS detection and defense^[9]. Unlike these works, the present study does not introduce a defense mechanism. Instead, it first establishes a baseline of model vulnerability under a controlled threat model.

D. Research Gap

Most conventional evaluations focus on clean-test performance, which may not fully reveal a detector's behavior under adversarial manipulation. This study addresses this gap by examining how RF, XGBoost, and MLP respond to constrained feature-space perturbations with different budgets and directions. The goal is to determine whether strong clean-test performance remains consistent under the defined adversarial conditions.

III. METHODOLOGY



Fig. 1. Experimental research framework for the InSDN robustness evaluation.

A. Dataset and Binary Problem Formulation

The experiments use the InSDN dataset, developed for intrusion-detection research in Software-Defined Networking

environments^[2]. Three source files were used: Normal_data.csv, metasploitable-2.csv, and OVS.csv. After initial cleaning, the combined dataset contained 343,888 records, including 68,423 normal and 275,465 attack records.

The original attack categories were converted into a binary classification problem. Normal traffic was assigned label 0, while all non-Normal traffic was assigned label 1. Exact duplicates, Flow ID, source IP, destination IP, and Timestamp were removed. Infinite values were converted to missing values, and preprocessing statistics were derived only from the training data to avoid data leakage.

The data was divided using a stratified 70/15/15 split into training, validation, and test sets. Constant features were removed using the training data, followed by mutual-information-based feature selection. The 40 features with the highest mutual-information scores were retained for the final experiments.

B. Selected Feature Representation

The selected features describe packet, flow, rate, and timing characteristics of network traffic. The same feature set was used for all three classifiers to maintain a consistent comparison.

The 40 selected features are:

Fwd Header Len; Bwd Header Len; Dst Port; Pkt Len Max; Pkt Len Std; Pkt Len Var; Pkt Len Mean; Pkt Size Avg; Bwd Seg Size Avg; Subflow Bwd Byts; TotLen Bwd Pkts; Bwd Pkt Len Mean; Bwd Pkt Len Max; Fwd Pkt Len Mean; Fwd Seg Size Avg; Subflow Fwd Pkts; TotLen Fwd Pkts; Bwd Pkt Len Std; Fwd Pkt Len Max; Subflow Fwd Byts; TotLen Fwd Pkts; Subflow Bwd Pkts; TotLen Bwd Pkts; Protocol; Fwd Pkts/s; Src Port; Bwd Pkts/s; Flow IAT Mean; Flow Byts/s; Init Bwd Win Byts; Bwd IAT Mean; Flow Pkts/s; Bwd IAT Max; Flow IAT Max; Bwd IAT Tot; Bwd Pkt Len Min; Flow IAT Std; Flow Duration; Pkt Len Min; Down/Up Ratio.

C. Adversarial Threat Model

The adversarial evaluation is performed directly in feature space. Only malicious test samples that are correctly classified as attacks on the clean test set are used. Perturbable continuous and statistical features are shifted simultaneously in the same direction, while port and protocol features remain unchanged.

Four perturbation budgets are evaluated: $\epsilon = 0.01, 0.025, 0.05$, and 0.10 , with both directions, $d \in \{-1, +1\}$. Feature-specific modification ranges are based on the 0.1th and 99.9th empirical percentiles.

The perturbed value of feature j is calculated as:

$$x'_j = \text{clip}(x_j + d \epsilon (Q_{0.999,j} - Q_{0.001,j}), L_j, U_j) \quad (1)$$

where x_j is the original feature value, x'_j is the perturbed value, d defines the perturbation direction, and ϵ controls its normalized magnitude. $Q_{0.999,j}$ and $Q_{0.001,j}$ represent the 99.9th and 0.1th percentile values, while L_j and U_j are the clipping bounds.

This approach provides a consistent perturbation scale across features with different numerical ranges. The ϵ values represent normalized feature-space budgets rather than literal percentage changes in packets, bytes, or network traffic. Therefore, the resulting samples are treated as a controlled robustness test rather than a direct simulation of a deployable packet-level attack.



Fig. 2. Offline feature-space adversarial evaluation process.

D. Attack Success Rate

The attack success rate (ASR) is calculated only for malicious samples that were correctly classified as attacks on the clean test set. This prevents samples that were already false negatives from being counted as successful adversarial evasions. ASR is defined as:

$$ASR = (\text{Successful evasions} / \text{Correctly detected malicious samples}) \times 100 (2)$$

A total of 41,316 malicious test samples were correctly detected and therefore form the eligible set for the adversarial experiments. After perturbation, a sample is considered a successful evasion if its prediction changes from **Attack** to **Normal**.

The ASR is interpreted within the specific threat model used in this study. A high ASR indicates that the applied feature-space perturbation frequently changes correct attack predictions to normal predictions. It does not necessarily mean that the same modification could be directly applied to real network traffic while preserving the original attack behavior. Similarly, a low ASR under one direction does not imply robustness against other attack strategies.

The classifiers remain fixed throughout the adversarial evaluation. No adversarial samples are added to the training data and no model is retrained after perturbation. The perturbation ranges are based on empirical feature percentiles to account for the different scales of the selected features, and the resulting values are clipped to the defined bounds.

The clean test set is kept unchanged, and the same 41,316 eligible samples are evaluated under every perturbation budget and direction. This provides a consistent basis for comparing the adversarial results across models.

E. Evaluation Metrics and Experimental Controls

Clean-test performance is evaluated using accuracy, precision, recall, F1 score, ROC-AUC, and false-positive rate. These metrics provide a baseline for conventional classification performance,

while ASR is used specifically to measure the frequency of successful evasions under the defined adversarial procedure.

The final dataset contains 240,721 training records, 51,583 validation records, and 51,584 test records. The dataset remains imbalanced toward the attack class. Balanced class weighting is therefore used for Random Forest during training, while the other classifiers use their specified configurations without modifying the test distribution.

The original test distribution is retained throughout the evaluation. This ensures that both clean and adversarial results are obtained from the same test population and that the reported ASR is not affected by artificial changes to the evaluation set.

IV. RESULTS

A. Clean-Test Performance

All three classifiers produce near-perfect clean-test performance. XGBoost has the highest clean-test F1 at 99.9952%, followed by Random Forest at 99.9927% and MLP at 99.9855%. False-positive rates are below 0.1% for all three models. These results establish that the models can separate the binary classes extremely well under the conventional test distribution.

TABLE I. CLEAN-TEST PERFORMANCE

Model	Accuracy	Precision	Recall	F1	ROC-AUC	FPR
Random Forest	0.99988 4	0.99995 2	0.99990 3	0.99992 7	0.99995 0	0.00019 5
XGBoost	0.99992 2	0.99995 2	0.99995 2	0.99995 2	0.99999 6	0.00019 5
MLP	0.99976 7	0.99980 6	0.99990 3	0.99985 5	0.99998 8	0.00077 9

The corresponding confusion-matrix results also show that very few errors occur during clean testing. Random Forest produces 2 false positives and 4 false negatives, while XGBoost produces 2 false positives and 2 false negatives. MLP records 8 false positives and 4 false negatives. Rather than focusing on the small differences between the three models, the more important observation is that all of them provide a very strong baseline on the untouched test set. This establishes a useful reference point for examining how their predictions change once adversarial perturbations are introduced.

B. Adversarial Results

The adversarial evaluation presents a noticeably different picture from the clean-test results. Random Forest shows a particularly strong response to positive-direction perturbations, even at the smallest budget considered. At $\epsilon=0.01$, 38,875 out of the 41,316 eligible malicious samples are successfully misclassified as normal, resulting in an ASR of 94.09%. When the perturbation budget is increased to $\epsilon=0.10$, all 41,316 eligible samples are successfully evaded in the positive direction, corresponding to an ASR of 100%. The negative direction has a much smaller effect, with ASR increasing from 8.96% at $\epsilon=0.01$ to 35.20% at $\epsilon=0.10$.

XGBoost follows a different pattern as the perturbation budget increases. For the positive direction, the ASR begins at 16.52% for $\epsilon=0.01$ and increases to 40.84% at $\epsilon=0.025$. A much larger increase occurs at $\epsilon=0.05$, where the ASR reaches 93.15%. At $\epsilon=0.10$, the positive-direction ASR remains at a similarly high

level of 93.14%. The negative direction is less effective overall, with ASR increasing more gradually from 8.61% at the smallest budget to 31.88% at the largest tested budget.

The MLP behaves differently at the lower perturbation budgets. At $\epsilon=0.01$, its ASR remains below 1% for both directions, indicating very few successful evasions under the smallest perturbation. Its negative-direction ASR also remains below 1% at $\epsilon=0.025$. In contrast, the positive direction becomes considerably more effective as the budget increases. At $\epsilon=0.025$, the ASR reaches 25.46%, followed by a substantial increase to 87.78% at $\epsilon=0.05$. At $\epsilon=0.10$, all eligible malicious samples are successfully evaded in the positive direction, resulting in an ASR of 100%. The negative direction remains below 1% across all four tested budgets.

TABLE II. ADVERSARIAL EVASION RESULTS

Model	ϵ	Dir.	Successful evasions	ASR (%)
RF	0.010	-1	3,701	8.96
RF	0.010	+1	38,875	94.09
RF	0.025	-1	6,245	15.12
RF	0.025	+1	39,180	94.83
RF	0.050	-1	8,767	21.22
RF	0.050	+1	38,489	93.16
RF	0.100	-1	14,544	35.20
RF	0.100	+1	41,316	100.00
XGBoost	0.010	-1	3,557	8.61
XGBoost	0.010	+1	6,827	16.52
XGBoost	0.025	-1	7,453	18.04
XGBoost	0.025	+1	16,875	40.84
XGBoost	0.050	-1	10,236	24.77
XGBoost	0.050	+1	38,488	93.15
XGBoost	0.100	-1	13,171	31.88
XGBoost	0.100	+1	38,484	93.14
MLP	0.010	-1	161	0.39
MLP	0.010	+1	40	0.10
MLP	0.025	-1	212	0.51
MLP	0.025	+1	10,521	25.46
MLP	0.050	-1	216	0.52
MLP	0.050	+1	36,269	87.78
MLP	0.100	-1	270	0.65
MLP	0.100	+1	41,316	100.00

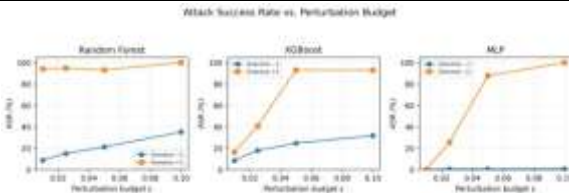


Fig. 3. Attack success rate (ASR) versus perturbation budget for the three models and both directions.

C. Directional Asymmetry

A strong directional asymmetry is present. At $\epsilon=0.01$, RF has 8.96% ASR for direction -1 compared with 94.09% for direction +1. XGBoost has 8.61% versus 16.52%, and MLP has 0.39% versus 0.10%. At larger budgets the asymmetry remains especially pronounced for MLP and RF. This indicates that the magnitude of the perturbation is not the only determinant of evasion; the direction in which the feature vector is moved can substantially change the outcome.

The directional effect should not be interpreted as a universal property of the algorithms in every dataset. It is an empirical property of the trained models, selected feature representation, and perturbation rule used here. Nevertheless, it is important because an evaluation that tests only one perturbation direction could materially understate or overstate observed vulnerability.

D. Clean Performance Versus Adversarial Vulnerability

TABLE III. CLEAN F1 AND REPRESENTATIVE ADVERSARIAL ASR

Model	Clean F1 (%)	Representative setting	ASR (%)
Random Forest	99.9927	$\epsilon=0.01, +1$	94.09
XGBoost	99.9952	$\epsilon=0.05, +1$	93.15
MLP	99.9855	$\epsilon=0.05, +1$	87.78

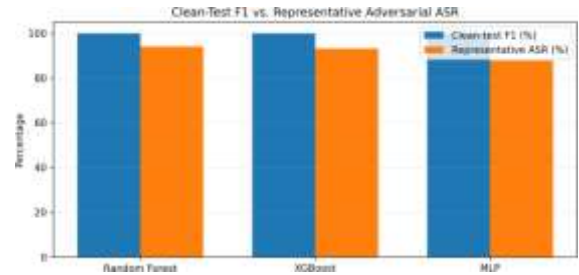


Fig. 4. Clean-test F1 compared with representative adversarial ASR.

The contrast is the main empirical result of the study. Each model has a clean-test F1 above 99.98%, yet each exhibits a high ASR under at least one tested positive-direction setting. Therefore, conventional test performance cannot by itself establish resistance to deliberate feature manipulation. The finding supports evaluating security-sensitive classifiers under an explicit threat model in addition to standard held-out testing.

V. DISCUSSION

A. Interpretation of the Main Finding

The results show that strong clean-test performance does not necessarily mean that a model is robust to adversarial manipulation. All three classifiers performed extremely well on clean data, but directional changes to the input features caused many correctly detected malicious samples to be classified as normal. This shows that the learned decision boundaries can be sensitive to systematic feature changes.

The perturbation budget also affected the results differently across models. RF showed a high ASR even at $\epsilon=0.01$ and reached 100% at $\epsilon=0.10$, while XGBoost and MLP showed larger increases at higher positive-direction budgets. This suggests that robustness should be evaluated across multiple attack strengths rather than at a single perturbation level.

B. Model Architecture and Direction

RF, XGBoost, and MLP responded differently to the same perturbation. This is expected because tree-based models and neural networks learn decision boundaries in different ways. However, the current experiments do not identify the specific features or model components responsible for these differences. Feature attribution, ablation studies, and individual feature-group attacks would be needed for that analysis.

The results should therefore not be treated as a simple ranking of model robustness. They reflect sensitivity under the specific perturbation strategy used in this study. A different threat model or feature constraint could produce different results.

C. Relationship to Prior Work

The findings agree with previous research showing that adversarial manipulation can affect machine-learning-based intrusion detection systems, including those used in SDN environments^{[10] [12] [16]}. Earlier studies have also explored adversarial training and other defensive approaches for improving DDoS detection robustness^{[9] [11]}.

Unlike those defense-oriented studies, this work focuses on establishing a vulnerability baseline. Using InSDN and a controlled feature-space attack, it measures how often initially correct malicious predictions change to normal under the defined threat model. This provides a baseline for evaluating defensive techniques in future work.

D. Practical Research Implication

The results suggest that clean accuracy and F1 score should not be the only measures used to evaluate ML-based intrusion detectors. A more complete evaluation should also define the threat model, perturbation budgets, eligible attack samples, ASR, and perturbation direction.

This baseline can also be used for future defense experiments. For example, adversarial training can be evaluated using the same attack procedure to determine whether it actually reduces the observed vulnerability rather than only improving clean-test performance.

E. Limitations

Several limitations should be considered. First, the attack is an offline feature-space experiment and does not generate or validate modified packets in a live SDN environment. Therefore, the results demonstrate classifier sensitivity rather than the effectiveness of a complete real-world network attack.

Second, the perturbation is applied in a fixed directional manner rather than being optimized individually for each sample. It should therefore be viewed as a controlled robustness probe rather than an exhaustive search for the strongest possible attack.

Third, ϵ represents a normalized feature-space budget and does not correspond directly to a percentage change in network traffic. The experiment also does not fully model dependencies between network features, so some generated feature combinations may not represent physically realizable flows.

Finally, the study uses one dataset and a binary classification setting, and does not include adversarial training or other defenses. Results may therefore differ on other datasets, attack distributions, or threat models. These areas are left for future work.

F. Reproducibility

The experimental workflow includes data preparation, binary labeling, duplicate removal, identifier exclusion, stratified splitting, training-only preprocessing, mutual-information feature selection, model training, clean-test evaluation, and adversarial

generation. A fixed random seed of 42 is used during model development.

The final test set contains 51,584 samples, including 41,316 malicious samples that were correctly detected before perturbation. These samples form the same eligible set for all adversarial experiments. The perturbation budgets and directions are fixed in advance, and the original test set remains unchanged.

These settings provide a reproducible procedure for repeating the experiment using the same InSDN data, selected features, model configurations, and adversarial parameters. This is important because the reported ASR depends on the exact preprocessing, eligible-sample definition, perturbation bounds, and attack direction.

VI. CONCLUSION

This study evaluated RF, XGBoost, and MLP on the InSDN dataset using a controlled feature-space adversarial evasion procedure. Although all three models achieved over 99.98% F1 on clean data, their performance dropped significantly under adversarial perturbations. RF reached 94.09% ASR at $\epsilon=0.01$ and 100% at $\epsilon=0.10$, while XGBoost and MLP reached 93.15% and 87.78% ASR at $\epsilon=0.05$, respectively. MLP also reached 100% ASR at $\epsilon=0.10$.

These results show that high clean-test performance does not necessarily indicate adversarial robustness. The effect of an attack also depends on its strength and direction. This study is limited to an offline feature-space evaluation, while future work can examine packet-level attacks, feature constraints, adversarial training, model-aware attacks, and cross-dataset testing.

REFERENCES

- [1] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolkly, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015, doi: 10.1109/JPROC.2014.2371999.
- [2] M. Said Elsayed, N.-A. Le-Khac, and A. D. Jurdut, "InSDN: A novel SDN intrusion dataset," *IEEE Access*, vol. 8, pp. 165263–165284, 2020, doi: 10.1109/ACCESS.2020.3022633.
- [3] O. Yurekten and M. Demirci, "SDN-based cyber defense: A survey," *Future Generation Computer Systems*, vol. 115, pp. 126–142, 2021.
- [4] M. P. Singh, A. K. Pradhan, and S. K. Singh, "New-flow based DDoS attacks in SDN: Taxonomy, rationales, and research challenges," *Computer Communications*, vol. 154, pp. 509–525, 2020, doi: 10.1016/j.comcom.2020.02.085.
- [5] O. S. Sahoo et al., "An early detection of low rate DDoS attack to SDN based data center networks using information distance metrics," *Future Generation Computer Systems*, vol. 89, pp. 685–697, 2018, doi: 10.1016/j.future.2018.07.017.
- [6] A. Ahmed Bahashwan, M. Anbar, S. Manickam, T. A. Al-Amiedy, M. A. Aladaileh, and I. H. Hasbullah, "A systematic literature review on machine learning and deep learning approaches for detecting DDoS attacks in software-defined networking," *Sensors*, vol. 23, no. 9, p. 4441, 2023, doi: 10.3390/s23094441.
- [7] E. A. Al Mandhari et al., "A systematic literature review on the accuracy of machine learning based DDoS detection

techniques in Software Defined Networking,” *Discover Computing*, vol. 29, p. 147, 2026.

[8] W. Wang, Y. Liu, Q. Meng, and Z. Chen, “DDoS attack detection and defense techniques in software defined networks: A survey,” *Computer Science Review*, vol. 60, p. 100921, 2026, doi: 10.1016/j.cosrev.2026.100921.

[9] M. P. Novaes, L. F. Carvalho, J. Lloret, and M. L. Proença Jr., “Adversarial deep learning approach detection and defense against DDoS attacks in SDN environments,” *Future Generation Computer Systems*, vol. 125, pp. 156–167, 2021, doi: 10.1016/j.future.2021.06.047.

[10] A. A. Aiken and S. Scott-Hayward, “Investigating adversarial attacks against network intrusion detection systems in SDNs,” in *Proc. IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 2019, doi: 10.1109/NFV-SDN47374.2019.9040101.

[11] B. Nugraha, N. Kulkarni, and A. Gopikrishnan, “Detecting adversarial DDoS attacks in software-defined networking using deep learning techniques and adversarial training,” in *Proc. IEEE International Conference on Cyber Security and Resilience (CSR)*, 2021, pp. 448–454, doi: 10.1109/CSR51186.2021.9527967.

[12] K. He, D. D. Kim, and M. R. Asghar, “Adversarial machine learning for network intrusion detection systems: A comprehensive survey,” *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 538–566, 2023, doi: 10.1109/COMST.2022.3233793.

[13] A. Alotaibi and M. A. Rassam, “Adversarial machine learning attacks against intrusion detection systems: A survey on strategies and defense,” *Future Internet*, vol. 15, no. 2, p. 62, 2023, doi: 10.3390/fi15020062.

[14] H. Jmila and M. Ibn Khedher, “Adversarial machine learning for network intrusion detection: A comparative study,” *Computer Networks*, vol. 214, p. 109073, 2022, doi: 10.1016/j.comnet.2022.109073.

[15] L. A. P. da Silva et al., “Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues,” *Information Sciences*, vol. 239, pp. 201–225, 2013, doi: 10.1016/j.ins.2013.03.022.

[16] H. A. Alatwi and C. Morisset, “Adversarial machine learning in network intrusion detection domain: A systematic review,” 2021.

[17] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, “Evasion attacks against machine learning at test time,” in *Machine Learning and Knowledge Discovery in Databases*, Springer, 2013, pp. 387–402, doi: 10.1007/978-3-642-40994-3_25.

[18] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” in *Proc. International Conference on Learning Representations (ICLR)*, 2015.

[19] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” in *Proc. International Conference on Learning Representations (ICLR)*, 2014.

[20] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” in *Proc. International Conference on Learning Representations (ICLR)*, 2018.

[21] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

[22] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in *Proc. 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794, doi: 10.1145/2939672.2939785.

[23] F. Pedregosa et al., “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

BIOGRAPHIES



Furqan Mohammed Junaid Shaikh
M.Sc. Computer Science
Student at D.G. Ruparel College