International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(12s) 2026 pp. 713-723
https://doi.org/10.70917/ijcisim-2026-3943
Article

Reproducibility and Cross-Scenario Generalization of NetFlow-Based SQL Injection Detection: Threshold Transfer and Feature Dependence

Seema Joshi[1*], Neel Oza[2]

[*1]School of Engineering and Technology, Gujarat Technological University, Chandkheda, Ahmedabad, 382424, Gujarat, India.
[2]Master of Engineering in Cyber Security, Batch:2022-24, School of Engineering and Technology, Gujarat Technological University, Chandkheda, Ahmedabad, 382424, Gujarat, India.
*Corresponding author(s). E-mail(s): ap_seema@gtu.edu.in;
Contributing authors: ozaneel28@gmail.com;

Abstract: Machine-learning detectors for SQL injection (SQLi) may perform well under internal evaluation yet fail to preserve a usable decision threshold across traffic scenarios. This study evaluates that transfer risk using two independently generated NetFlow V5 datasets. D1 (400,003 flows) is used only for training, preprocessing, feature selection, hyperparameter selection, and threshold calibration; D2 (57,229 flows) is reserved for final cross-scenario testing. Six classifiers are evaluated with corrected FAR = FP/(FP+TN), confusion-matrix consistency checks, with-IP/without-IP ablation, and computational timing. In the primary baseline, Logistic Regression with IP features yields 85.25% accuracy, 87.09% F1-score, 99.46% recall, and 28.96% FAR. Under a D1-validation operating point that minimizes FAR subject to recall of at least 95%, LinearSVC with IP features yields the strongest D2 result (91.96% accuracy, 92.52% F1-score, 99.54% recall, and 15.63% FAR), whereas Random Forest thresholds collapse to all-benign predictions. Removing IP-address features reduces calibrated LinearSVC accuracy to 89.88% and raises FAR to 19.78%, although its ROC-AUC and PR-AUC remain high. A controlled audit also fails to reproduce the approximately 98% Logistic Regression result reported in the earlier preprint. Threshold transfer, false-alarm behavior, and reproducibility are therefore central deployment concerns for flow-based SQLi detection.

Keywords: SQL injection; NetFlow; intrusion detection; reproducibility; threshold calibration; cross-scenario generalization.

1. Introduction

SQL injection (SQLi) is a long-standing application-security vulnerability in which untrusted input changes the intended structure or semantics of a database query. Successful exploitation may expose confidential information, modify records, bypass authentication, or disrupt the availability of database-backed services. Preventive controls such as parameterized queries, secure input handling, least-privilege database accounts, and web application firewalls remain essential. Network-level monitoring nevertheless provides an additional defensive layer, especially when defenders require centralized visibility across multiple applications.

Most SQLi-detection research operates at the query or payload level, where lexical, syntactic, and semantic patterns can be analyzed directly. Flow-based monitoring addresses a different operational setting. NetFlow records summarize communication through compact metadata such as packet and byte counts, addresses, ports, protocols, and TCP-related fields. This representation reduces the storage and inspection burden associated with full packet capture, but it removes application-layer context and can be more sensitive to distribution shift between collection environments (Campazas-Vega et al., 2020; Crespo-Martínez et al., 2023).

Crespo-Martínez et al. (2023) demonstrated SQLi detection using NetFlow and released the D1/D2 dataset lineage used in the present work (Campazas-Vega and Crespo-Martínez, 2022). Because the datasets and the DOROTHEA collection framework originate from that prior work, the present contribution is not the introduction of flow-based SQLi detection. Instead, this paper tests whether classical machine-learning models remain reliable under a strict D1-to-D2 cross-scenario protocol with reproducible preprocessing, corrected false-alarm accounting, explicit threshold calibration, IP-feature ablation, and computational measurements.

The motivation for this reassessment is methodological. The earlier preprint reported approximately 98% accuracy and an extremely low false-alarm rate for Logistic Regression (Joshi and Oza, 2024). During the reproducibility audit, the reported false-alarm equation was found to be incorrect, and the historical high-performing result could not be reproduced from the committed datasets and visible pipeline. Rather than retaining an unsupported headline result, the present study reports only values generated by the completed audited run and uses the discrepancy to examine how threshold choice, feature dependence, and distribution shift affect operational conclusions.

Recent SQLi research has moved toward deep multi-view models, optimized feature selection, lightweight attention architectures, and deployment-oriented web application firewall frameworks (Ahmed and Al dabag, 2026; Aldossary and Aleisa, 2026; Arasteh et al., 2024a, 2024b; Kakisim, 2024; Lo et al., 2025). These approaches generally operate on query or payload representations rather than flow metadata. The present study therefore addresses a complementary question: how robustly can compact NetFlow features support cross-scenario SQLi detection when model development and calibration are isolated from the final test environment?

The main contributions are:

- an audited, leakage-controlled D1-to-D2 evaluation in which D2 is never used for preprocessing, feature selection, hyperparameter tuning, or threshold calibration;
- a six-classifier comparison using corrected FAR = FP/(FP+TN), complete confusion-matrix consistency checks, ranking metrics, and computational timing;
- a with-IP versus without-IP ablation that quantifies dependence on srcaddr and dstaddr;
- a threshold-transfer study in which Logistic Regression uses a D1-validation maximum-F1 threshold in the primary baseline and all score-producing models are evaluated under a common D1-only security-oriented threshold rule;
- a reproducibility analysis showing that the historical approximately 98% Logistic Regression D2 result is not recoverable from the committed datasets and visible implementation; and
- an operational analysis demonstrating that high ROC-AUC/PR-AUC can coexist with substantial test-set FAR when thresholds do not transfer well across traffic scenarios.

The evaluation addresses four research questions: RQ1, how do six classical classifiers perform when developed on D1 and transferred unchanged to D2; RQ2, how strongly do their conclusions depend on source and destination IP-address features; RQ3, do D1-derived operating thresholds retain acceptable recall and FAR on D2; and RQ4, can the historical Logistic Regression result be regenerated from the committed data and visible pipeline? The security contribution is therefore not a new classifier architecture. The classifiers serve as controlled probes of deployment reliability, reproducibility, and validation methodology for network security monitoring.

2. Related Work

SQLi defenses span secure programming, query rewriting, signature matching, anomaly detection, runtime monitoring, and machine learning. String-matching and input-focused defenses remain relevant (Abikoye et al., 2020), while machine-learning studies have used query-level classifiers, ensembles, and cloud-oriented datasets to distinguish benign and malicious inputs (Kasim, 2021; Roy et al., 2022; Tripathy et al., 2020; Uwagbole et al., 2017). Probabilistic neural approaches have also been explored (Alarfaj and Khan, 2023). These methods benefit from application-layer visibility but are not directly deployable at observation points where only flow telemetry is available.

Crespo-Martínez et al. (2023) provide the closest flow-level baseline, demonstrating SQLi detection using NetFlow records generated through the DOROTHEA environment (Campazas-Vega et al., 2020). Their work introduced the D1/D2 research context and reported strong Logistic Regression performance. The present paper uses the same public dataset lineage but focuses on reproducibility, strict separation of model development from D2, corrected FAR computation, IP-feature ablation, and threshold transfer. This distinction is essential because reusing the same dataset family with a different classifier is not, by itself, a methodological contribution.

The methodological landscape now includes deep multi-view consensus (Kakisim, 2024), bio-inspired feature selection and classification (Arasteh et al., 2024a, 2024b), lightweight attention models that emphasize model size and inference speed (Lo et al., 2025), and deployment-focused machine-learning or WAF frameworks (Ahmed and Al dabag, 2026; Aldossary and Aleisa, 2026). Because these studies use application-layer representations and different dataset families, their headline metrics are not directly comparable with D1-to-D2 NetFlow transfer. Table 1 therefore positions the present study by evaluation objective rather than by unsupported cross-dataset performance ranking.

Study	Representation	Evaluation emphasis	Distinction from the present study
Crespo-Martínez et al. (2023)	NetFlow V5	Flow-based SQLi detection across D1/D2	Closest dataset and task baseline; the present study audits reproducibility, isolates all D2 use, calibrates thresholds on D1 only, and adds IP ablation and timing.
Kasim (2021); Roy et al. (2022)	SQL queries / payload features	Classical and ensemble classification	Application-layer inputs and internal evaluation; no NetFlow threshold-transfer analysis.
Kakisim (2024); Arasteh et al. (2024a, 2024b)	Multi-view or optimized payload features	Deep learning and bio-inspired feature selection	Focuses on model or feature optimization rather than cross-scenario operating-point transfer.
Lo et al. (2025)	Tokenized SQL payloads	Lightweight attention, model size, and inference speed	Deployment-oriented but based on application-layer tokens rather than compact flow telemetry.
Ahmed and Al dabag (2026); Aldossary and Aleisa (2026)	Payload / WAF features	Real-time or deployment-oriented detection	Uses a different representation and dataset family; headline metrics are therefore not directly comparable with D1-to-D2 NetFlow transfer.
Present study	NetFlow V5	Six models, D1-only calibration, D2 final test, IP ablation, timing, and audit	Treats models as probes of reproducibility, distribution shift, false alarms, and threshold transfer rather than proposing a new classifier.

Table 1. Positioning of the present study relative to representative SQLi-detection research.

3. Materials and Methods

3.1 Dataset and cross-scenario protocol

The study uses two publicly available NetFlow V5 datasets from the SQLi network-flow research context (Campazas-Vega and Crespo-Martínez, 2022; Crespo-Martínez et al., 2023). D1 contains 400,003 flows and D2 contains 57,229 flows. D1 is approximately class-balanced; D2 contains 28,615 benign and 28,614 malicious flows. D1 contains union-oriented SQLi activity involving MySQL and Microsoft SQL Server environments, whereas D2 contains blind SQLi activity involving PostgreSQL. The datasets also use different address spaces. D1 is the sole development dataset, and D2 is used only once for independent final evaluation.

Dataset	Role	Flows	Class distribution	Scenario
D1	Training/validation only	400,003	≈50% benign / ≈50% malicious	Union-based SQLi; MySQL and SQL Server
D2	Independent final test	57,229	28,615 benign / 28,614 malicious	Blind SQLi; PostgreSQL

Table 2. Dataset roles and cross-scenario evaluation design.

3.2 Preprocessing and feature selection

For the definitive run, D1 is split reproducibly into a stratified training partition (320,002 rows) and validation partition (80,001 rows), with random_state=42 used wherever the estimator or splitter exposes that parameter. SimpleImputer, MinMaxScaler, and SelectKBest with chi-square scoring are fitted on the D1 training partition only. The fitted transformations are then applied unchanged to D1 validation and D2. The selected features listed in Table 3 are those exported by the completed run, rather than features reconstructed from D2. Model development uses scikit-learn (Pedregosa et al., 2011).

Two ablations are evaluated. In the with-IP condition, the selected features are dpkts, doctets, srcaddr, dstaddr, input, output, srcport, prot, tos, and tcp_flags. In the without-IP condition, srcaddr and dstaddr are removed before preprocessing and feature selection; the retained features are dpkts, doctets, input, output, srcport, dstport, prot, tos, and tcp_flags. This ablation tests whether cross-scenario performance depends strongly on address-related fields.

Ablation	Dropped features	Selected features
With IP	None	dpkts; doctets; srcaddr; dstaddr; input; output; srcport; prot; tos; tcp_flags
Without IP	srcaddr; dstaddr	dpkts; doctets; input; output; srcport; dstport; prot; tos; tcp_flags

Table 3. Feature-ablation configurations.

3.3 Models, tuning, and threshold calibration

Six classifiers are evaluated: Logistic Regression, Random Forest, SGDClassifier, K-Nearest Neighbors (KNN), LinearSVC, and a hard-voting ensemble. Logistic Regression hyperparameters are selected by five-fold GridSearchCV on the D1 training partition, consistent with the project methodology. All remaining estimator settings, selected features, thresholds, and timing records are taken directly from the completed run's machine-readable outputs. No D2 label or score is used for estimator selection, feature selection, hyperparameter selection, or threshold calibration. The hard-voting classifier combines independently fitted Logistic Regression, Random Forest, SGDClassifier, KNN, and LinearSVC estimators.

The primary baseline uses each estimator's standard operating point, except Logistic Regression, for which the threshold that maximizes F1 on D1 validation is used (0.63 with IP features and 0.62 without IP features in the completed run). The secondary analysis applies a common security-oriented rule to each score-producing model: select, using D1 validation only, the threshold that minimizes validation FAR subject to validation recall of at least 95%. Hard Voting remains at standard majority rule and is not included in the secondary threshold table. This design separates a conventional primary comparison from a targeted test of operating-point transfer.

3.4 Evaluation metrics and consistency checks

Malicious traffic is the positive class. TP denotes malicious flows classified as malicious, TN benign flows classified as benign, FP benign flows classified as malicious, and FN malicious flows classified as benign. The study reports accuracy, precision, recall, F1-score, FAR, ROC-AUC, and PR-AUC where score outputs are available. FAR is defined as FP/(FP+TN). Automated checks verify that TN+FP+FN+TP equals 57,229 for every D2 evaluation and that each scalar operating-point metric recomputes from its confusion matrix. Models producing only one predicted class are explicitly flagged.

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN)$$

$$\text{Precision} = TP / (TP + FP)$$

$$\text{Recall} = TP / (TP + FN)$$

$$\text{FAR} = FP / (FP + TN)$$

$$F1 = 2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$$

4. Results

4.1 Primary cross-scenario evaluation

Table 4 summarizes the definitive primary D2 results. Logistic Regression with IP features provides the strongest primary F1-score (0.8709) and accuracy (0.8525), with recall of 0.9946 but FAR of 0.2896. Random Forest is slightly weaker by F1 and exhibits FAR of 0.3185. LinearSVC has lower default-threshold accuracy than Logistic Regression, but its ranking metrics are markedly stronger (ROC-AUC 0.9907; PR-AUC 0.9848), indicating that its standard operating point does not fully exploit its score ordering. KNN achieves perfect recall at the cost of FAR above 0.58.

Classifier	Accuracy	Precision	Recall	F1	FAR	ROC-AUC	PR-AUC
Logistic Regression	0.8525	0.7745	0.9946	0.8709	0.2896	0.7879	0.6334
Random Forest	0.8383	0.7575	0.9951	0.8602	0.3185	0.8372	0.7976
SGD Classifier	0.7934	0.7091	0.9949	0.8280	0.4082	0.8315	0.6774
KNN	0.7079	0.6312	1.0000	0.7739	0.5843	0.7080	0.6313
LinearSVC	0.8180	0.7347	0.9955	0.8455	0.3594	0.9907	0.9848
Voting Classifier	0.7890	0.7045	0.9955	0.8251	0.4175	N/A	N/A

Table 4. Primary D2 performance with IP-address features.

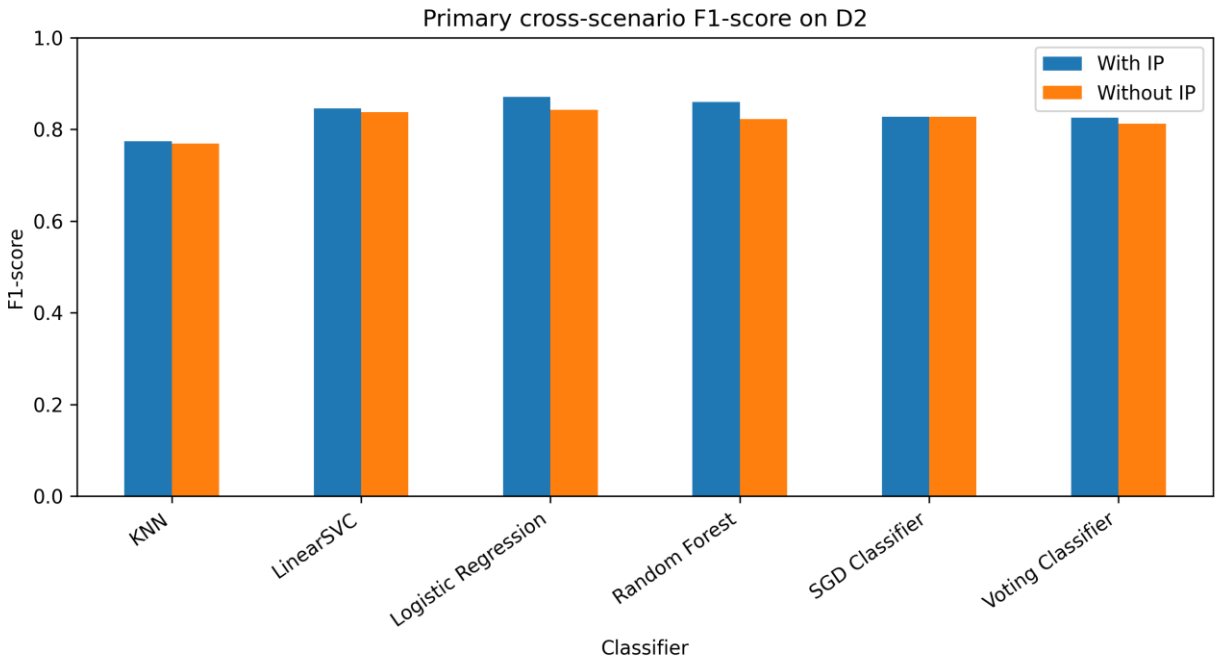

Figure 1. Primary F1-score comparison for with-IP and without-IP configurations.

4.2 IP-feature ablation

Removing srcaddr and dstaddr generally reduces operating performance, but the magnitude varies by classifier. For Logistic Regression, accuracy falls from 0.8525 to 0.8148 and F1 from 0.8709 to 0.8430. For LinearSVC, default-threshold accuracy changes from 0.8180 to 0.8069, while ROC-AUC increases from 0.9907 to 0.9978 and PR-AUC from 0.9848 to 0.9985. This contrast indicates that address removal can improve ranking quality while simultaneously making the transferred operating threshold less suitable for D2.

Classifier	Acc. IP	Acc. No IP	F1 IP	F1 No IP	FAR IP	FAR No IP

Logistic Regression	0.8525	0.8148	0.8709	0.8430	0.2896	0.3651
Random Forest	0.8383	0.7848	0.8602	0.8222	0.3185	0.4254
SGD Classifier	0.7934	0.7934	0.8280	0.8280	0.4082	0.4082
KNN	0.7079	0.7001	0.7739	0.7693	0.5843	0.5998
LinearSVC	0.8180	0.8069	0.8455	0.8375	0.3594	0.3817
Voting Classifier	0.7890	0.7699	0.8251	0.8122	0.4175	0.4558

Table 5. Primary with-IP versus without-IP ablation.

4.3 Secondary threshold-calibration analysis

D1-validation-only threshold calibration materially changes classifier ranking. At the operating point selected to minimize validation FAR while preserving at least 95% validation recall, LinearSVC with IP features achieves the strongest D2 result: 0.9196 accuracy, 0.9252 F1, 0.9954 recall, and 0.1563 FAR. SGDClassifier also improves substantially, reaching 0.8943 accuracy and 0.9039 F1 with FAR of 0.2059. Logistic Regression improves more modestly to 0.8619 accuracy and 0.8780 F1 with FAR of 0.2708.

Random Forest presents an important negative result. The D1-derived calibrated threshold (0.79056) leads to single-class benign predictions on D2 in both ablations, even though the primary score-based evaluation shows non-trivial ranking ability. No post-hoc adjustment using D2 was performed. This failure illustrates that a threshold can appear highly favorable on the development distribution and still transfer poorly under cross-scenario shift.

Ablation	**Classifier**	**Threshold**	**Accuracy**	**Precision**	**Recall**	**F1**	**FAR**
with ip	Logistic Regression	0.6418	0.8619	0.7860	0.9945	0.8780	0.2708
with ip	Random Forest	0.7906	0.5000	0.0000	0.0000	0.0000	0.0000
with ip	SGD Classifier	0.9726	0.8943	0.8285	0.9946	0.9039	0.2059
with ip	KNN	1.0000	0.7080	0.6313	1.0000	0.7740	0.5839
with ip	LinearSVC	0.2662	0.9196	0.8643	0.9954	0.9252	0.1563
without ip	Logistic Regression	0.6326	0.8341	0.7529	0.9946	0.8571	0.3264
without ip	Random Forest	0.8101	0.5000	0.0000	0.0000	0.0000	0.0000
without ip	SGD Classifier	0.9886	0.8455	0.7662	0.9945	0.8656	0.3035
without ip	KNN	1.0000	0.7002	0.6251	1.0000	0.7693	0.5996

without ip	LinearSVC	0.2614	0.8988	0.8342	0.9954	0.9077	0.1978

Table 6. Secondary D1-validation-calibrated operating points (minimum validation FAR subject to validation recall of at least 95%).

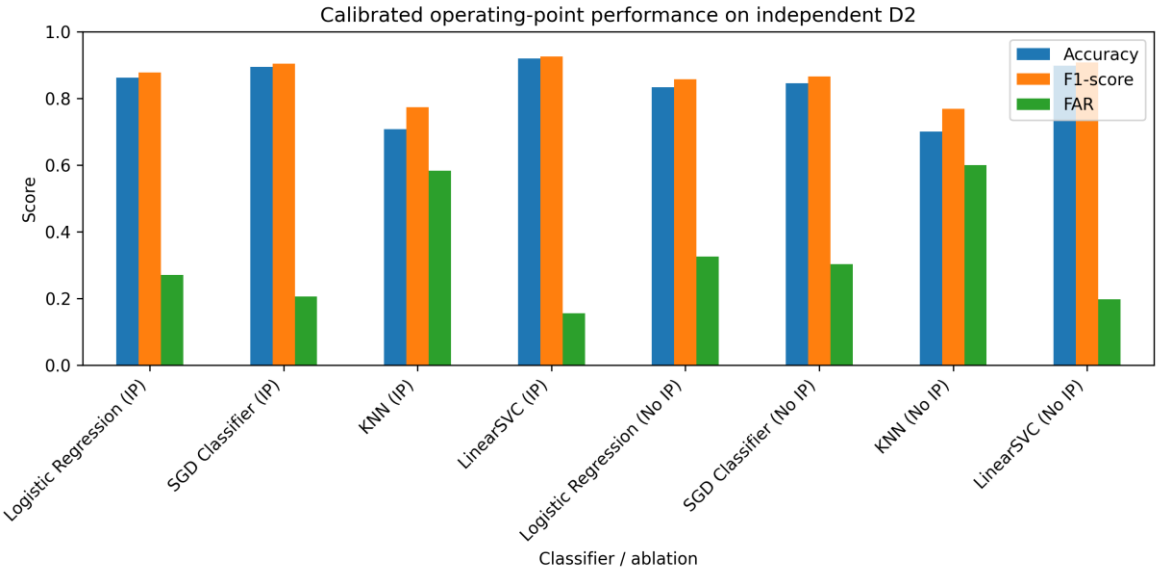

Figure 2. D2 performance at D1-validation-calibrated operating points. Random Forest is excluded from the plotted bars because both calibrated thresholds produced single-class predictions on D2.

4.4 Computational performance

Primary-run timing indicates a marked contrast between linear models and instance-based or ensemble methods. KNN inference on the full D2 test set requires 26.15 s with IP features and 12.39 s without IP features, whereas Logistic Regression, SGDClassifier, and LinearSVC complete inference in milliseconds once preprocessing is complete. Hard Voting also incurs substantially higher inference cost than its constituent linear models. These measurements support the practical efficiency of linear classifiers, although end-to-end deployment cost would also include flow export, preprocessing, and alert handling.

Ablation	Classifier	Train time (s)	Inference time (s)	Threshold
with ip	Logistic Regression	33.4561	0.009662	0.63
with ip	Random Forest	3.1786	0.137798	standard
with ip	SGD Classifier	0.6549	0.003010	standard
with ip	KNN	0.6445	26.149976	standard
with ip	LinearSVC	3.0384	0.003202	standard
with ip	Voting Classifier	8.1282	13.474714	standard
without ip	Logistic Regression	27.8236	0.004353	0.62
without ip	Random Forest	3.6008	0.138770	standard
without ip	SGD Classifier	0.6276	0.002665	standard
without ip	KNN	0.5571	12.394488	standard

without ip	LinearSVC	1.6260	0.002544	standard
without ip	Voting Classifier	6.2454	6.212125	standard

Table 7. Primary-run training and D2 inference times.

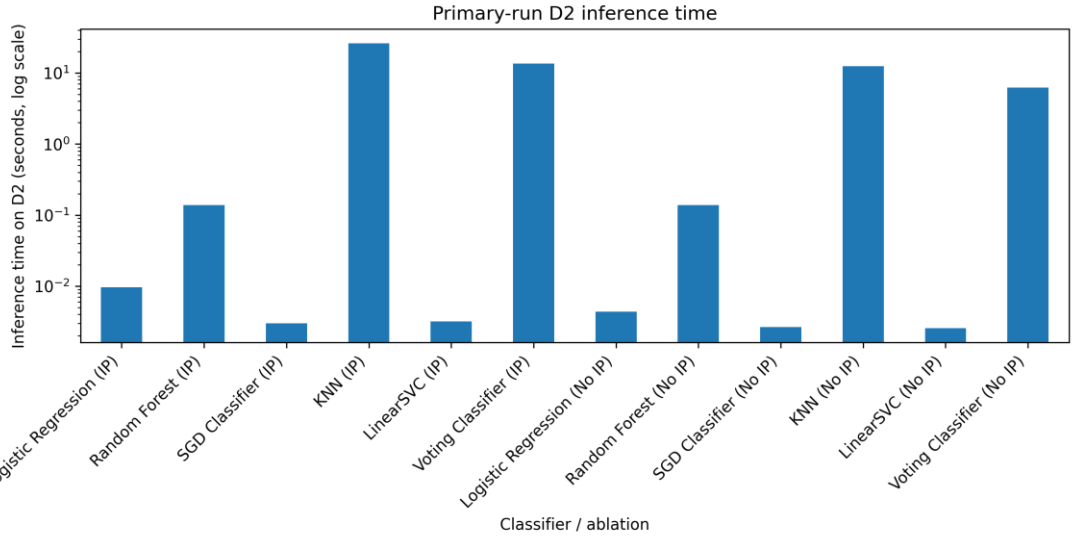

Figure 3. Primary-run inference time on D2 (logarithmic scale).

4.5 Reproducibility analysis of the historical Logistic Regression result

A controlled audit compared four Logistic Regression configurations: the visible original pipeline at thresholds 0.30 and 0.50, the corrected leakage-controlled pipeline at 0.50, and the corrected pipeline with a D1-validation-calibrated threshold. The visible original pipeline did not reproduce the approximately 98% D2 result reported in the preprint (Joshi and Oza, 2024). At threshold 0.30, accuracy was 72.81% and FAR 53.89%; at threshold 0.50, accuracy was 78.72% and FAR 42.05%. The corrected calibrated configuration selected threshold 0.63 and achieved 85.25% accuracy, 87.09% F1-score, and 28.96% FAR. The selected features were identical across the original and corrected runs, and the visible in-script scaler and selector were fitted on D1 rather than D2. The historical result therefore cannot be attributed to the visible scaler/selector logic and is most plausibly associated with an unrecoverable difference in data preparation, file pairing, metric interpretation, or evaluation conditions.

Experiment	Threshold	Accuracy	Precision	Recall	F1	FAR
Original pipeline	0.30	0.7281	0.6487	0.9951	0.7854	0.5389
Original pipeline	0.50	0.7872	0.7029	0.9949	0.8238	0.4205
Corrected calibrated	0.63	0.8525	0.7745	0.9946	0.8709	0.2896
Corrected at 0.50	0.50	0.7875	0.7032	0.9949	0.8240	0.4200

Table 8. Controlled Logistic Regression reproducibility diagnostics.

5. Discussion

The results demonstrate that flow-level SQLi detection remains feasible under a demanding cross-scenario test, but they also show why accuracy or recall alone is insufficient for operational conclusions. Most models achieve recall above 0.99, yet several simultaneously produce substantial FAR. In a production environment where benign traffic

dominates, even a moderate FAR can translate into an impractical alert volume. The key challenge is therefore not merely identifying malicious flows, but transferring a useful decision boundary from D1 to D2.

LinearSVC provides the clearest example of the distinction between ranking quality and operating-point quality. Its ROC-AUC and PR-AUC are exceptionally high, particularly without IP features, yet its default threshold yields materially lower accuracy and higher FAR than the calibrated secondary operating point. Threshold calibration on D1 validation improves D2 performance, especially with IP features, but does not eliminate false alarms. This suggests that the score ordering generalizes better than the absolute score calibration.

The IP ablation yields a nuanced result. Removing addresses reduces calibrated operating performance, indicating that source/destination information contributes useful structure. However, the strong without-IP ranking metrics of LinearSVC show that the model is not merely memorizing addresses. The performance gap is more consistent with a combination of behavioral signal and scenario-specific calibration effects. Future work should therefore investigate more invariant address representations, role-based features, or domain-adaptation methods rather than simply retaining raw numeric addresses.

Random Forest demonstrates the risks of transferring thresholds across distributions. Its validation-derived calibrated thresholds produce single-class benign predictions on D2. This failure was not manually corrected, because doing so using D2 would contaminate the independent evaluation. Reporting the failure is important: model selection procedures that appear optimal on an internal validation split can still fail under a meaningful environmental shift.

The reproducibility analysis changes the interpretation of the earlier preprint (Joshi and Oza, 2024). The historical approximately 98% Logistic Regression D2 result is not reproduced by the committed data and visible pipeline, so it is excluded from the present evidence base. The corrected results are less favorable but more defensible: they expose substantial false-alarm behavior and clarify that the main operational risk is threshold transfer rather than recall alone.

6. Limitations and Threats to Validity

- The D1 and D2 datasets originate from controlled traffic-generation environments and may not reflect the diversity, imbalance, encryption, and background noise of production networks.
- The class distributions are approximately balanced; operational precision and alert burden may differ substantially when attacks are rare.
- NetFlow V5 is used; validation with NetFlow v9/IPFIX and IPv6 would improve contemporary applicability.
- Raw numeric IP-address fields may encode topology-specific information even though the cross-scenario address spaces differ.
- Threshold calibration is performed on one D1 validation partition. Repeated nested validation or temporal validation could provide a more stable estimate.
- Random Forest threshold collapse demonstrates sensitivity to distribution shift; broader calibration methods such as Platt scaling, isotonic regression, or conformal approaches remain to be evaluated.
- Timing measurements are environment-specific. Primary-run timings are used for computational comparison; secondary operating-point timing should not be interpreted as end-to-end model inference cost.
- The study evaluates SQLi-related flow detection as a complementary control and does not replace secure coding, parameterized queries, WAFs, or application-layer inspection.
- The model comparison is limited to six classical estimators from the audited workflow. Payload-level deep-learning systems use different inputs and datasets and are therefore discussed qualitatively rather than ranked against the present results.
- The exact alert burden at realistic attack prevalence was not measured. Production deployment requires evaluation on independently collected organizational traffic and monitoring for calibration drift.

7. Conclusion

This study provides a reproducibility-oriented cross-scenario evaluation of machine learning for NetFlow-based SQL injection detection. By isolating D2 from all development steps, correcting the FAR definition, auditing confusion-matrix consistency, calibrating thresholds only on D1 validation data, and performing IP-feature ablation, the analysis reveals a more complex picture than headline accuracy suggests. The strongest secondary operating point is obtained by LinearSVC with IP features, achieving 91.96% accuracy, 92.52% F1, 99.54% recall, and 15.63% FAR

on D2. Without IP features, LinearSVC retains very strong ranking metrics but experiences weaker threshold transfer, highlighting the difference between discrimination and calibration under distribution shift.

The study also establishes that the historical approximately 98% Logistic Regression result cannot be reproduced from the committed datasets and visible pipeline and is therefore not used as verified evidence. More broadly, the findings show that threshold transferability, false-alarm behavior, and reproducibility are central to the practical assessment of flow-based intrusion detection. Future work should evaluate domain adaptation, robust calibration, production-like class imbalance, NetFlow v9/IPFIX, IPv6, and independently collected organizational traffic. A promising deployment direction is a multi-stage architecture in which a fast flow-level model performs broad screening and higher-context application or WAF telemetry is used to validate suspicious events.

Declarations

Funding: This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Declaration of competing interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability: The D1 and D2 NetFlow datasets are publicly available on Zenodo at https://doi.org/10.5281/zenodo.6907251.

Code availability: The original implementation is available at https://github.com/neeloza28/Sql_injection. The audited experiment outputs supporting this manuscript are available from the corresponding author on reasonable request.

Author contributions: Seema B. Joshi and Neel Oza contributed to the study conception, implementation, analysis, interpretation, and preparation of the manuscript. Both authors reviewed and approved the final manuscript.

Preprint disclosure: An earlier version of this work was posted on Research Square (Joshi and Oza, 2024; https://doi.org/10.21203/rs.3.rs-4362691/v1). The present manuscript reports a corrected, expanded, and audited analysis.

Ethics statement: Not applicable. The study uses publicly available network-flow datasets and involves no human participants, animals, or personally identifiable information.

Declaration of generative AI and AI-assisted technologies in the manuscript preparation process

During the preparation of this work, the authors used ChatGPT (OpenAI) to assist with language refinement, manuscript organization, and editorial restructuring. After using this tool, the authors reviewed, verified, and edited the content as needed and take full responsibility for the content of the article.

References

1. Abikoye, O.C., Abubakar, A., Dokoro, A.H., Akande, O.N., Kayode, A.A., 2020. A novel technique to prevent SQL injection and cross-site scripting attacks using Knuth-Morris-Pratt string match algorithm. EURASIP Journal on Information Security 2020, 14. https://doi.org/10.1186/s13635-020-00113-y
2. Ahmed, S.S., Al dabag, M.L., 2026. Deployment-oriented multi-embedding machine learning framework for SQL injection detection and prevention in a web application firewall. Computers 15(6), 368. https://doi.org/10.3390/computers15060368
3. Aldossary, H., Aleisa, N., 2026. Machine-learning-based detection of SQL injection attacks in web applications. Applied Sciences 16(10), 4691. https://doi.org/10.3390/app16104691
4. Alarfaj, F.K., Khan, N.A., 2023. Enhancing the performance of SQL injection attack detection through probabilistic neural networks. Applied Sciences 13(7), 4365. https://doi.org/10.3390/app13074365
5. Arasteh, B., Aghaei, B., Farzad, B., Arasteh, K., Kiani, F., Torkamanian-Afshar, M., 2024a. Detecting SQL injection attacks by binary gray wolf optimizer and machine learning algorithms. Neural Computing and Applications 36, 6771-6792. https://doi.org/10.1007/s00521-024-09429-z
6. Arasteh, B., Bouyer, A., Sefati, S.S., Craciunescu, R., 2024b. Effective SQL injection detection: A fusion of binary Olympiad optimizer and classification algorithm. Mathematics 12(18), 2917. https://doi.org/10.3390/math12182917
7. Campazas-Vega, A., Crespo-Martínez, I.S., Guerrero-Higueras, A.M., Fernández-Llamas, C., 2020. Flow-data gathering using NetFlow sensors for fitting malicious-traffic detection models. Sensors 20(24), 7294. https://doi.org/10.3390/s20247294

8. Campazas-Vega, A., Crespo-Martínez, I.S., 2022. SQL Injection Attack Netflow (D1-D2). Zenodo. https://doi.org/10.5281/zenodo.6907251
9. Crespo-Martínez, I.S., Campazas-Vega, A., Guerrero-Higueras, A.M., Riego-DelCastillo, V., Álvarez-Aparicio, C., Fernández-Llamas, C., 2023. SQL injection attack detection in network flow data. Computers & Security 127, 103093. https://doi.org/10.1016/j.cose.2023.103093
10. Joshi, S.B., Oza, N., 2024. Enhanced network security against SQL injection attack using machine learning. Research Square [preprint]. https://doi.org/10.21203/rs.3.rs-4362691/v1
11. Kakisim, A.G., 2024. A deep learning approach based on multi-view consensus for SQL injection detection. International Journal of Information Security 23, 1541-1556. https://doi.org/10.1007/s10207-023-00791-y
12. Kasim, Ö., 2021. An ensemble classification-based approach to detect attack level of SQL injections. Journal of Information Security and Applications 59, 102852. https://doi.org/10.1016/j.jisa.2021.102852
13. Lo, R.-T., Hwang, W.-J., Tai, T.-M., 2025. SQL injection detection based on lightweight multi-head self-attention. Applied Sciences 15(2), 571. https://doi.org/10.3390/app15020571
14. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, É., 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825-2830. https://jmlr.org/papers/v12/pedregosa11a.html
15. Roy, P., Kumar, R., Rani, P., 2022. SQL injection attack detection by machine learning classifier. In: 2022 International Conference on Applied Artificial Intelligence and Computing (ICAAIC), pp. 394-400. https://doi.org/10.1109/ICAAIC53929.2022.9792964
16. Tripathy, D., Gohil, R., Halabi, T., 2020. Detecting SQL injection attacks in cloud SaaS using machine learning. In: 2020 IEEE 6th International Conference on Big Data Security on Cloud, High Performance and Smart Computing, and Intelligent Data and Security, pp. 145-150. https://doi.org/10.1109/BigDataSecurity-HPSC-IDS49724.2020.00035
17. Uwagbole, S.O., Buchanan, W.J., Fan, L., 2017. Applied machine learning predictive analytics to SQL injection attack detection and prevention. In: 2017 IFIP/IEEE Symposium on Integrated Network and Service Management, pp. 1087-1090. https://doi.org/10.23919/INM.2017.7987433