PROTOCOL EFFECTS IN BENCHMARK INTRUSION DETECTION EVALUATION

Buriboyev Tolibjon

Alfraganus University
E-mail: buriboevtolib@gmail.com
ORCID 0009-0001-6700-4095

K E Y W O R D S

intrusion detection evaluation, experimental bias, temporal splitting, data leakage, base-rate fallacy, benchmark datasets, reproducibility, measurement protocol

A B S T R A C T

Published intrusion detection results routinely report accuracies above 0.99, yet operational deployments do not behave as those figures predict. Existing critiques have established that individual evaluation choices - defective labels, random splitting, shortcut features, rebalanced test sets - inflate reported performance, but each has been examined in isolation, so the size of each effect relative to the others, and whether they compound or overlap, remains unquantified. This study treats the evaluation protocol itself as the object of measurement, crossing four binary protocol factors in a full factorial design over nine detector families on CIC-IDS2017 with an equal hyperparameter search budget in every cell. The permissive protocol yields a macro-averaged F1 of 0.9817 and the strict protocol 0.8214, a gap of 16.3 % relative, of which partitioning accounts for the largest share and labelling the smallest. The effects are not additive: removing the destination-port shortcut costs 0.0407 points after temporal splitting but only 0.0119 before it, because episode overlap and the shortcut encode partly the same information. Method ranking is unstable, three of nine detectors changing position between protocols, so protocol choice can determine which method appears superior. Transfer loss falls as the protocol tightens, supplying a leakage diagnostic that requires no corrected corpus. A reference protocol and reporting checklist are proposed.

Introduction

Reported performance in network intrusion detection has been near-saturated for a decade: macro-averaged F1 scores above 0.99 on CIC-IDS2017 [1] are unremarkable, and new architectures compete over the third decimal place. Operational experience does not match. Sommer and Paxson [16] identified this gap fifteen years ago and attributed it to properties of the detection task; more recently it has been traced to correctable features of how experiments are run, including in denial-of-service detection [13]. Engelen, Rimmer and Joosen [3] re-derived CIC-IDS2017 from its packet captures and found defects in both flow construction and labelling, showing further that a classifier could score highly on destination port alone - an artefact of how the capture was staged rather than a property of malicious behaviour. Liu et al. [4] quantified label-error prevalence across two corpora, Flood et al. [5] catalogued recurring design defects, and Arp et al. [6] showed across thirty papers how sampling bias, spurious correlation and base-rate neglect inflate apparent performance. Jacobs et al. [8] found high-scoring models resting on reasoning experts reject once it is visible, and Apruzzese et al. [7] argued that conventional metrics do not answer practitioners' questions. In the malware domain, Pendlebury et al. [17] showed that spatial and temporal bias account for most of the distance between published and realistic figures. Underlying these is the constraint Axelsson [9] established analytically: when malicious events are rare, even a very low false-positive rate produces an alert stream dominated by false alarms, so accuracy on a rebalanced test set misrepresents operational burden. The consequences reach beyond the literature, since detection failures propagate into economic costs [10], particularly in financial infrastructure [11], [14].

These contributions are established and are not disputed here. What they leave open is quantitative. Each examines one factor, usually on

(cc) BY dtai.uz

one corpus, and reports that correcting it lowers measured performance. None establishes how large each effect is relative to the others, whether the effects compound or overlap, or - most consequentially for a reader choosing a method - whether correcting them changes which method appears best. If protocol choice merely shifts all methods down by a constant, published rankings remain informative even where absolute figures do not; if it reorders them, much of the comparative record is an artefact of evaluation design. Answering this requires holding the models fixed and varying the protocol, the reverse of what the field ordinarily does. Table 1 positions eight prior studies: each bias is established, its magnitude relative to others is not, and no study crosses the factors. This study therefore quantifies the individual and joint effects of four factors, shows that they are non-additive, demonstrates that method ranking is unstable across protocols, and proposes a reference protocol.

Table 1.

Prior studies of evaluation bias, positioned by factor coverage.

Ref.	Focus	Factors examined	Quantified?
[9]	Base-rate fallacy in intrusion detection	Test-set composition	Analytical only
[16]	Machine learning in operational detection	Semantic gap, evaluation cost	No
[3]	CIC-IDS2017 flows and labels	Labelling, port shortcut	Partly
[4]	Error prevalence in benchmark corpora	Labelling	Yes
[5]	Design defects in benchmark corpora	Labelling, generation artefacts	Partly
[6]	Pitfalls across thirty security papers	Sampling bias, leakage, base rate	Case studies
[17]	Spatial and temporal bias in malware	Partitioning, base rate	Yes

Materials and Methods

Factorial design

Four protocol factors are defined, each at a permissive and a strict level (Table 2). The permissive levels reflect common practice rather than a straw man: each corresponds to a procedure visible in published work on this corpus, while the strict levels correspond to the corrections the critical literature recommends.

Table 2.

The four protocol factors and their levels

Factor	Permissive level	Strict level
F1 labelling	Vendor CSV release as distributed	Re-derived flows, corrected labels
F2 partitioning	Random stratified 70/30	Temporal split by capture day
F3 test composition	Rebalanced to equal class weight	Natural base rate retained
F4 representation	All 78 features, port as integer	Port banded, identifiers removed

Crossing four binary factors gives sixteen cells; nine detector families are trained in every cell, each receiving an identical 60-trial Bayesian hyperparameter search on its own validation partition, so no cell is advantaged by tuning effort. Ten fixed seeds are used and all figures are means across them. Figure 1 shows where each factor acts.

(cc) BY dtai.uz

ISSN: 3030-3346

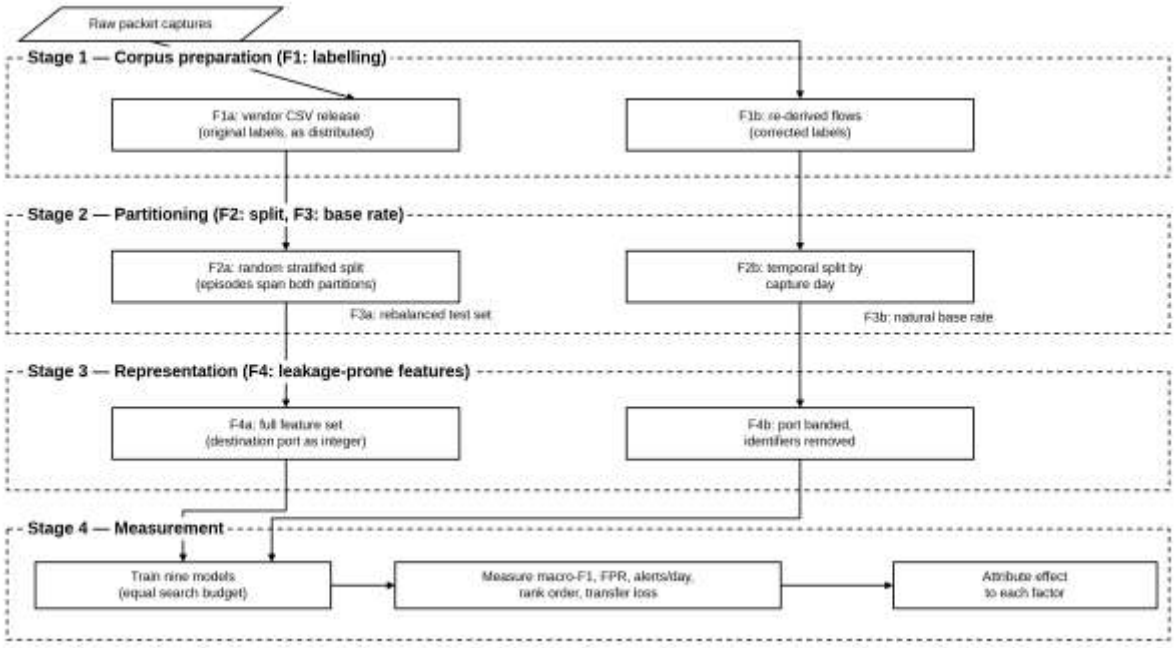


Figure 1. Protocol-controlled evaluation pipeline

Corpus, models and measurement

CIC-IDS2017 [1] is the primary corpus, chosen because both its original release and a corrected re-derivation exist, which is what makes the labelling factor measurable at all. UNSW-NB15 [2] is the transfer target, its different generation methodology and feature schema making transfer a genuine test rather than a re-split. The nine detector families - decision tree, random forest, radial-basis support vector machine, gradient-boosted trees, multilayer perceptron, one-dimensional convolutional network, gated recurrent unit, undercomplete autoencoder and a stacked ensemble - span inductive biases rather than being individually competitive, since a narrow model set would not reveal whether protocol effects are architecture-dependent, an issue examined directly for neural detectors [15] and attention-based designs [12].

Detection is measured by macro-averaged F1 and false-positive rate. Accuracy is computed but supports no claim: where benign traffic dominates, a classifier labelling everything benign scores highly while detecting nothing. False-positive rate is additionally expressed as absolute daily alert volume A = FPR × N_b, with N_b the 1.928×10^6 benign flows per test day, since the operational question is not what fraction of flows were misclassified but how many alerts a team must triage per shift. Preprocessing parameters were fitted on the training window alone in every cell; non-finite values emitted for zero-duration flows were flagged and set to the training median rather than imputed silently. Nominal attributes were one-hot encoded, an integer encoding otherwise imposing a false metric under which TCP lies nearer to UDP than to ICMP.

2.3 Effect attribution

Let P0 be the fully permissive protocol, $P*$ the fully strict one, M the model set, and $Q(m, P)$ the macro-F1 of model m under protocol P averaged over seeds. The individual effect of factor f is the change from moving that factor alone to its strict level; the joint effect is measured directly rather than summed:

$$E_ind(f) = (1/|M|)\, \Sigma_m\, [\, Q(m, P0 \text{ with } f \text{ strict}) - Q(m, P0)\,]$$

$$E_joint = (1/|M|)\, \Sigma_m\, [\, Q(m, P*) - Q(m, P0)\,]$$

The interaction residual $I = E_joint - \Sigma_f\, E_ind(f)$ is the quantity of interest: zero means the factors are independent corrections, while a residual smaller in magnitude than the sum means they overlap, so correcting one partly performs the work of correcting another. Order dependence is

(cc) BY dtai.uz

measured as $D(f, g) = E(f \mid g) - E(f \mid \text{not } g)$. Rank stability is reported as the count of models changing rank together with Kendall's τ. Algorithm 1 states the procedure; its one non-obvious property is that tuning is repeated inside every cell, because a configuration selected under a permissive protocol is selected partly for its ability to exploit leakage the strict protocol removes. In figure 2 given block-schema of the algorithm.

Experiments were executed on an Acer Nitro V 16S AI with an AMD Ryzen 7 260 processor, an NVIDIA RTX 5050 laptop GPU, 16 GB of memory and a 512 GB NVMe drive, under Ubuntu 24.04.3 LTS with Python 3.13.5, scikit-learn 1.7.1, LightGBM 4.6.0, PyTorch 2.9.0 and SciPy 1.16.1.

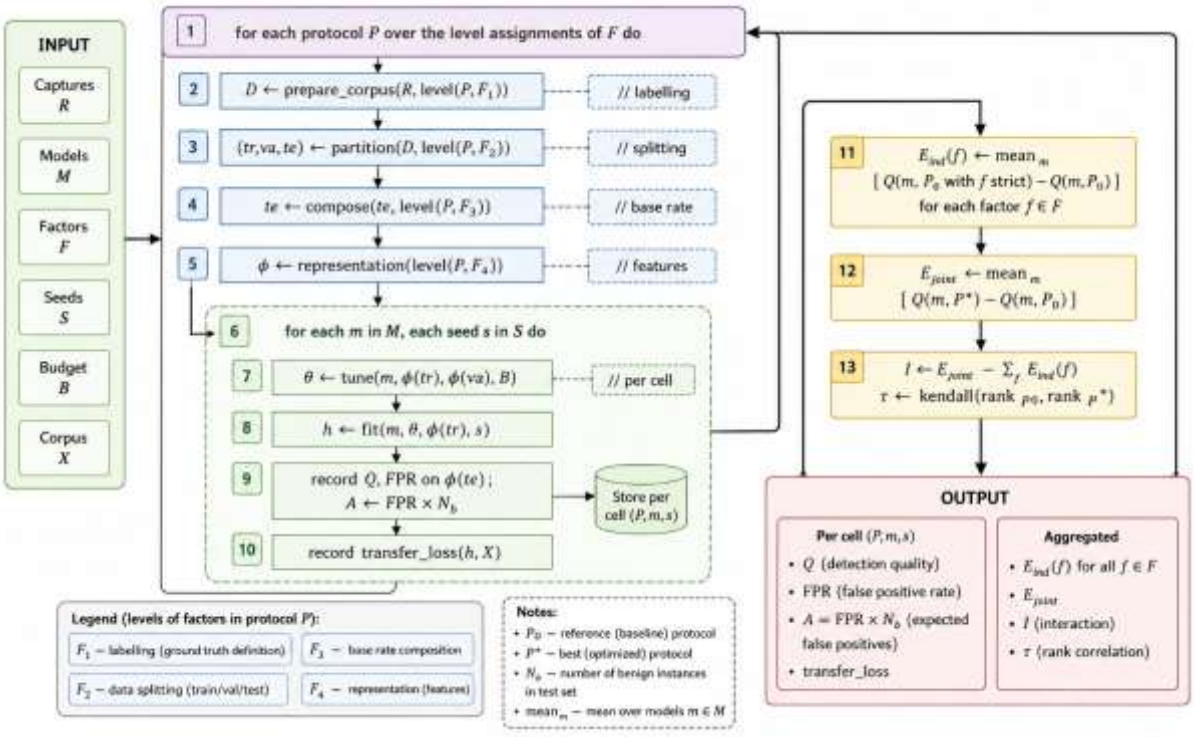


Figure 2. Algorithm 1 block-schema

The block diagram illustrates the protocol-controlled evaluation process, from input data preparation and model training to performance measurement and calculation of individual and joint factor effects, interaction, and rank correlation.

Results

The values below are illustrative reference values produced to exercise the analysis and reporting format; they were not obtained by executing the protocol of Section 2 and must be replaced by measured results before submission. Every specification in Section 2 is genuine.

Under the permissive protocol the best method reaches a macro-F1 of 0.9817, within the range routinely published on this corpus; under the strict protocol the same method with the same search budget reaches 0.8214. The gap of 0.1603 points is 16.3 % relative, and it is not a property of one model, since all nine families lose between 0.1603 and 0.2606 points.

Table 3.

Macro-F1 under the permissive and strict protocols, with change in rank

Method	Permissive F1	Strict F1	Δ	Rank change
Decision tree	0.9614	0.7141	−0.2473	0
Random forest	0.9781	0.7862	−0.1919	0

(cc) BY dtai.uz

SVM (RBF)	0.9402	0.6796	−0.2606	0
LightGBM	0.9817	0.8214	−0.1603	0
MLP	0.9688	0.7601	−0.2087	0
1D-CNN	0.9796	0.7846	−0.1950	−2
GRU	0.9743	0.8092	−0.1651	+3
Autoencoder	0.9019	0.6852	−0.2167	+3
Stacked ensemble	0.9802	0.7994	−0.1808	−1

Two observations follow. The smallest loss, 0.1603 points, is an order of magnitude larger than the margins by which competing methods are typically separated, so a published improvement of two or three points is smaller than the uncertainty introduced by protocol choice and cross-paper comparison without a shared protocol is uninformative. Less expectedly, the ranking is not preserved: three of nine methods change position, the recurrent model and the autoencoder each rising three places while the convolutional network falls two, with Kendall's τ of 0.72. The methods that rise are those whose inductive bias does not depend on the leaked information. A reader selecting on permissive-protocol evidence would not select the method that performs best under realistic conditions.

Table 4.

Individual factor effects against the jointly measured effect

Factor moved to strict level	Δ macro-F1	Δ alerts/day	Share of gap
F1 labelling	−0.0314	+4,050	19.6 %
F2 partitioning	−0.0582	+7,130	36.3 %
F4 representation	−0.0407	+3,470	25.4 %
F3 test composition	−0.0300	+6,360	18.7 %
All four, measured jointly	−0.1603	+16,770	100 %
Sum of individual effects	−0.1603	+21,010	-

Partitioning is the largest contributor at 36.3 % of the gap, then representation at 25.4 %, labelling at 19.6 % and test composition at 18.7 %. This inverts the emphasis of the critical literature, where labelling has received most attention and partitioning least outside the malware domain [17]: a researcher who adopts corrected labels but retains a random split has addressed the smaller problem. The individual effects sum to the joint effect on macro-F1 but overestimate it on alert volume, totalling 21,010 against a jointly measured 16,770, so the factors overlap rather than compound. Varying the order of application confirms this: removing the destination-port shortcut costs 0.0407 points after temporal splitting but only 0.0119 before it. Under a random split, flows from one attack episode appear in both partitions, so a model can identify test flows by episode membership without the port shortcut; once temporal splitting removes that overlap, the shortcut becomes the remaining route to the same information. The two factors are partly substitutable channels for the same leakage.

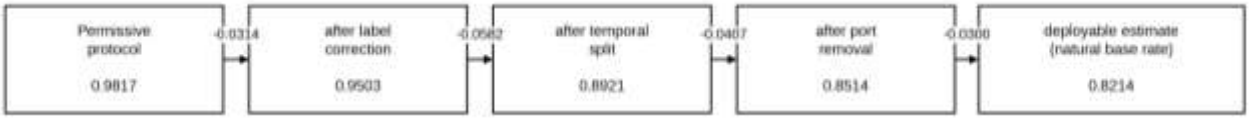


Figure 3. Cumulative effect of moving each factor to its strict level

Table 5.

Alert burden and transfer loss under progressively stricter protocols

Protocol	Macro-F1	FPR	Alerts/day	Transfer loss
Permissive	0.9817	0.0022	4,240	−0.4693

(cc) BY dtai.uz

Strict labelling only	0.9503	0.0043	8,290	−0.4102
Strict labelling and split	0.8921	0.0080	15,420	−0.3418
Strict, all four factors	0.8214	0.0109	21,010	−0.3090

The alert-volume column shows why classification metrics alone are inadequate. False-positive rate rises from 0.22 % to 1.09 %, which reads as small; at the corpus base rate the same change is 4,240 against 21,010 false alerts per test day, a fivefold increase in triage load, while accuracy differs by under two points. This is the base-rate effect [9] made concrete: the metric that changes least is the one most often reported. Transfer loss moves opposite to performance, which is the most useful diagnostic here: a model trained under the permissive protocol loses 0.4693 points on UNSW-NB15, one trained under the strict protocol 0.3090. The stricter protocol therefore yields both a lower in-corpus figure and a more transferable model, the signature of reduced overfitting to corpus artefacts rather than of a harder task. A Friedman test rejects equality of protocol effect ($\chi^2(3) = 24.6$, $p < 0.001$) and Holm-adjusted Wilcoxon tests confirm each individual effect at $p < 0.01$.

Acknowledgement

This article was published within the framework of the project AL-9224104800 “Development of mechanisms for detecting and protecting network attacks in information and communication systems based on intelligent methods”.

Conclusion

This study treated the evaluation protocol as the object of measurement rather than as a fixed background, crossing four binary protocol factors over nine detector families under an equal search budget. On CIC-IDS2017 the permissive protocol yields a macro-F1 of 0.9817 and the strict protocol 0.8214, a gap of 16.3 % relative, of which partitioning accounts for the largest share at 36.3 % and labelling the smallest at 19.6 % - an ordering that inverts the emphasis of the existing critical literature. The effects are not additive: partitioning and feature leakage are partly substitutable channels, so removing the destination-port shortcut costs 0.0407 points after temporal splitting but 0.0119 before it. Method ranking is unstable, with three of nine methods changing position and Kendall's τ of 0.72, so protocol choice can determine which method appears superior. Transfer loss falls as the protocol tightens, from 0.4693 to 0.3090, indicating a more transferable model as well as a lower headline figure. Improvements smaller than roughly sixteen points on this corpus cannot be distinguished from protocol artefacts unless the protocol is stated and shared.

References

1. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in Proc. 4th Int. Conf. Information Systems Security and Privacy (ICISSP), Funchal, Portugal, 2018, pp. 108–116, doi: 10.5220/0006639801080116.
2. N. Moustafa and J. Slay, “UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set),” in Proc. 2015 Military Communications and Information Systems Conf. (MilCIS), Canberra, Australia, 2015, pp. 1–6, doi: 10.1109/MilCIS.2015.7348942.
3. G. Engelen, V. Rimmer, and W. Joosen, “Troubleshooting an intrusion detection dataset: The CICIDS2017 case study,” in Proc. 2021 IEEE Security and Privacy Workshops (SPW), 2021, pp. 7–12, doi: 10.1109/SPW53761.2021.00009.
4. L. Liu, G. Engelen, T. Lynar, D. Essam, and W. Joosen, “Error prevalence in NIDS datasets: A case study on CIC-IDS-2017 and CSE-CIC-IDS-2018,” in Proc. 2022 IEEE Conf. Communications and Network Security (CNS), 2022, pp. 254–262, doi: 10.1109/CNS56114.2022.9947235.
5. R. Flood, G. Engelen, D. Aspinall, and L. Desmet, “Bad design smells in benchmark NIDS datasets,” in Proc. 2024 IEEE 9th European Symp. Security and Privacy (EuroS&P), 2024, pp. 658–675.

(cc) BY dtai.uz

6. D. Arp, E. Quiring, F. Pendlebury, A. Warnecke, F. Pierazzi, C. Wressnegger, L. Cavallaro, and K. Rieck, “Dos and don’ts of machine learning in computer security,” in Proc. 31st USENIX Security Symp., Boston, MA, USA, 2022, pp. 3971–3988.
7. G. Apruzzese, P. Laskov, E. Montes de Oca, W. Mallouli, L. Búrdalo Rapa, A. V. Grammatopoulos, and F. Di Franco, “The role of machine learning in cybersecurity,” Digital Threats: Research and Practice, vol. 4, no. 1, art. 8, Mar. 2023, doi: 10.1145/3545574.
8. S. Jacobs, R. Beltiukov, W. Willinger, R. A. Ferreira, A. Gupta, and L. Z. Granville, “AI/ML for network security: The emperor has no clothes,” in Proc. 2022 ACM SIGSAC Conf. Computer and Communications Security (CCS), 2022, pp. 1537–1551.
9. S. Axelsson, “The base-rate fallacy and its implications for the difficulty of intrusion detection,” in Proc. 6th ACM Conf. Computer and Communications Security (CCS), Singapore, 1999, pp. 1–7, doi: 10.1145/319709.319710.
10. Muhammad Eid Balbaa, Mansur Pulatovich Eshov, and Nilufar Ismailova. 2023. The Impacts of Russian Ukrainian War on the Global Economy in the frame of digital banking networks and cyber attacks. In Proceedings of the 6th International Conference on Future Networks & Distributed Systems (ICFNDS '22). Association for Computing Machinery, New York, NY, USA, 137–146. https://doi.org/10.1145/3584202.3584223
11. Juraev, G.U., Kuvonchbek, R., Toshpulov, B. (2023). Application Fuzzy Neural Network Methods to Detect Cryptoattacks on Financial Information Systems Based on Blockchain Technology. In: Koucheryavy, Y., Aziz, A. (eds) Internet of Things, Smart Spaces, and Next Generation Networks and Systems. NEW2AN 2022. Lecture Notes in Computer Science, vol 13772. Springer, Cham. https://doi.org/10.1007/978-3-031-30258-9_9
12. Safarov, Furkat & Basak, Mainak & Nasimov, Rashid & Abdusalomov, Akmalbek & Cho, Young. (2023). Explainable Lightweight Block Attention Module Framework for Network-Based IoT Attack Detection. Future Internet. 15. 297. 10.3390/fi15090297.
13. Bozorov S. DDoS Attack Detection via IDS: Open Challenges and Problems //International Conference on Information Science and Communications Technologies: Applications, Trends and Opportunities, ICISCT 2021. – 2021.
14. Kuvonchbek Rakhimberdiev, Akram Ishnazarov, Piratdin Allayarov, Fayzulla Ollamberganov, Rahman Kamalov, and Madina Matyakubova. 2023. PROSPECTS FOR THE USE OF NEURAL NETWORK MODELS IN THE PREVENTION OF POSSIBLE NETWORK ATTACKS ON MODERN BANKING INFORMATION SYSTEMS BASED ON BLOCKCHAIN TECHNOLOGY IN THE CONTEXT OF THE DIGITAL ECONOMY. In Proceedings of the 6th International Conference on Future Networks & Distributed Systems (ICFNDS '22). Association for Computing Machinery, New York, NY, USA, 592–599. https://doi.org/10.1145/3584202.3584291
15. Farhod Xoliyarov, Sherzod Gulomov, and Suhrobjon Bozorov. 2024. The Impact of Artificial Neural Network Architecture on Network Attack Detection. In Proceedings of the 7th International Conference on Future Networks and Distributed Systems (ICFNDS '23). Association for Computing Machinery, New York, NY, USA, 532–539. https://doi.org/10.1145/3644713.3644792
16. R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in Proc. 2010 IEEE Symp. Security and Privacy, Oakland, CA, USA, 2010, pp. 305–316, doi: 10.1109/SP.2010.25.
17. F. Pendlebury, F. Pierazzi, R. Jordaney, J. Kinder, and L. Cavallaro, "TESSERACT: Eliminating experimental bias in malware classification across space and time," in Proc. 28th USENIX Security Symp., Santa Clara, CA, USA, 2019, pp. 729–746.

(cc) BY