An expert evaluation of efficient models for packet level traffic generation

Maurice Falk, Hannes Schwanzer, Adrian Ulges, Martin Gergeleit, Alexander Prange

Angaben zur Veröffentlichung / Publication details:

Falk, Maurice, Hannes Schwanzer, Adrian Ulges, Martin Gergeleit, and Alexander Prange. 2026. “An expert evaluation of efficient models for packet level traffic generation.” In *7th KuVS Fachgespräch on Machine Learning in Networking (MaLeNe), University of Augsburg, Augsburg, Germany, March 19-20, 2026: proceedings*, edited by Michael Seufert, Andreas Blenk, and Björn Richerzhagen. Augsburg: Universität Augsburg.

7TH KUVS FACHGESPRÄCH ON MACHINE LEARNING IN NETWORKING (MaLeNe)

PROCEEDINGS

MARCH 19-20
2026

UNIVERSITY OF AUGSBURG, AUGSBURG, GERMANY

An Expert Evaluation of Efficient Models for Packet Level Traffic Generation

Maurice Falk[1], Hannes Schwanzer[2], Adrian Ulges[1], Martin Gergeleit[1]
RheinMain University of Applied Sciences
Wiesbaden, Germany
[1]{firstname.lastname}@hs-rm.de, [2]{firstname.lastname}@student.hs-rm.de

Alexander Prange
consistec GmbH
Saarbrücken, Germany
alexander.prange@consistec.de

***Abstract*—While state-of-the-art synthetic network traffic generation relies on resource-intensive, byte-level autoregressive models, their computational overhead poses a barrier to scalability. This paper explores first steps towards a more efficient model by utilizing Bayesian networks (BN) for packet-level header synthesis, leveraging structural dependencies and domain heuristics rather than raw byte sequences. Besides protocol compliance checks and training data suitability (TSTR), we evaluate generation approaches through a novel expert discrimination test. Preliminary results show that, while BN-generated flows currently exhibit lower fidelity than byte-level models, they achieve high TCP protocol compliance and experts' error is 53% when discriminating real traffic from generated one.**

***Index Terms*—flow generation, packet header, bayesian network, data synthesis, discrimination test**

I. Introduction

While ML-based network traffic analysis and security is gaining interest in an increasingly digitized world, it is hampered by the lack of annotated real-world network traffic. Research datasets are either constructed in laboratory environments to examine specific aspects such as intrusion detection [12], [13] or encrypted traffic classification [3], [6], are limited to campus networks [14], [17], or are kept private altogether [5], [7], [8]. Training directly on real-world target networks – though known to be vital for ML models [15] – is often infeasible due to privacy concerns. Therefore, recent research has turned towards generating network traffic that replicates the statistical properties of a target network's traffic.

This work addresses traffic generation at the packet level, which – in contrast to aggregated flow-level statistics – offers significantly higher fidelity and greater flexibility, and enables the generation of standard Packet Capturing (PCAP) files, which can be analyzed using tools like *Wireshark* [16]. Recent work frames such packet-level traffic generation as a language modeling task, generating header bytes as words/tokens with large-scale models such as transformers [8], [9], [18] and State Space Models (SSM) [2], as in Chu et al. recent *NetSSM* model, which achieves state-of-the-art results [1].

In this work, we address two open issues:
(1) Model Efficiency: Models like NetSSM feature >100 mio. parameters, and generate raw packet headers byte by byte (up to 94 tokens per packet), which is resource-intensive. Therefore, we argue that certain fields of a packet header can be heuristically derived (e.g., increasing Sequence Numbers) or do not change across a single flow (e.g., ethernet addresses / IPs / ports). Other generator architectures that only utilize derived fields from the packet headers, for example, NetShare [19] (GAN) or NetDiff [20] (Diffusion), but struggle to model stateful protocol characteristics [1], [18]. Inspired by Schoen et al. [11], who showed that Bayesian networks (BNs) outperform GAN-based models for generating network flows on an aggregated level, we ask the question how a much simpler and efficient BN-based approach fares on *packet-level* flow generation compared to much larger language models.
(2) Model Evaluation: While prior work has evaluated the authenticity of generated traffic only indirectly (by benchmarking classifiers trained on it, a strategy known as "Train on Synthetic, Test on Real" (TSTR) [4]) or sparsely (by checking some heuristic constraints such as TCP handshakes), generated traffic's plausibility to human experts has not been evaluated directly to our knowledge. To this end, we present an expert discrimination test, in which experts are asked to label flows as either real or generated. Particularly, we investigate how well expert-perceived authenticity aligns with the above prior evaluation strategies (i.e., domain checks and as TSTR). We share our code and expert annotations at *https://github.com/lavis-nlp/2026-malene-packet-generation*.

II. Flow Generation using Bayesian Networks

We generate packet data using a BN, where a packet's header fields depend only on the preceding packet. Each network node represents a packet header feature (namely, *payload length*, *inter-packet time*, *direction* (client/server), and *TCP flags*), while edges encode conditional dependencies between them. Other nodes include header fields of the previous packet, the packet index within the flow, a protocol label (such as *WEB_HTTP*), and a binary indicator denoting the flow's last packet. All continuous features are discretized using quantile binning ($n = 10$). The network structure is learned using Hill-Climbing search with the K2 score, and maximum likelihood estimation. To preserve causality, we restrict the structure such that input features, namely the preceding packet and protocol labels, cannot depend on features to be generated, namely the next packet and the end-of-flow indicator. The generation process can be either initialized by a separate BN trained only on first packets or by extracting the required features from a seed packet.

III. Experiments

We compare the flows generated by our BN generation model to the byte-level generation model NetSSM in three tasks: a novel expert discrimination test, domain-specific checks, and ML-based classifier training (TSTR).

Setup: Our experiments were performed using the *CIC-IDS-2017* [12] and Non-VPN traffic of the *ISCX-VPN-2016* [3] dataset. For each dataset, bi-directional flows were identified using the standard 5-tuple [10] with a 120-sec timeout. Since NetSSM targets TCP traffic, all non-TCP flows were discarded. For each dataset, we held out 20% of the flows for testing. As both our BN and NetSSM utilize a class label of the flow, we annotated a majority of the flows using a combination of deep packet inspection, expert domain knowledge, and heuristic rules, assigning their specific protocol or application (e.g., *FTP*). Since there is currently no official implementation of NetSSM shared by the authors, we re-implemented the base model according to the original specifications. However, we added a special end-of-sequence (EOS) token to each flow during training to explicitly determine flow completion during inference. Training of the models on 671k flows took $\approx$ 31 min (BN) and $\approx$ 153 h (NetSSM), respectively.

Expert Discrimination Test: In this experiment the realism of the generated flows is evaluated by experts through a manual discriminative test. Three network experts (with 2, 8, and 25 years of professional experience in the field) are individually given 200 flows for each model (100 real/synthetic) as PCAPs and asked to annotate each flow as either *real*, *likely real*, *likely generated*, or *generated*. The annotators did not have knowledge whether a flow is real or generated.

For evaluation consistency, real and synthetic flows were reconstructed into PCAPs using a shared feature subset. Static fields like IPs and ports were inherited from seed packets, while inter-packet times were fixed to a constant value to accommodate NetSSM's lack of temporal data.

TABLE I
MEAN AND STANDARD DEVIATION OF THE DISCRIMINATION TEST. ON AVERAGE, GENERATED FLOWS WHERE CLASSIFIED AS *(likely) real* IN 53% (BN) AND 70% (NETSSM) OF CASES.

Ground truth	Real	Likely Real	Likely Generated	Generated
NetSSM				
Real	13.00±12.53	51.00±17.09	17.67±15.7	18.33±20.79
Generated	17.67±17.04	52.33±18.04	15.33±11.85	14.67±11.02
Bayesian Network				
Real	20.33±17.79	51.00±25.87	15.00±7.00	13.67±13.61
Generated	4.67±4.16	48.67±21.55	26.00±12.29	20.67±7.37

The results in Table I indicate that for both models, the experts would generally tend to classify a flow as *(likely) real*, confirming that the generated flows look realistic (at high standard deviation between experts). For NetSSM, the experts rated truly generated flows as *(likely) real* more often (70%), while the generated flows of the BN-based generator were classified as *(likely) real* around 53%. For the BN, experts occasionally observed implausible TCP behavior, such as data being transmitted after FIN flags, while NetSSM showed more isolated errors, e.g., a TCP packet without flags. Overall, the NetSSM-generated flows were commented as 'more realistic'.

Domain Checks Following previous works [1], we evaluate whether the generated flows comply with TCP session rules. For this, we generated samples for up to 3000 flows per dataset. The generation took $\approx$ 13 mins for BN (single core) and $\approx$ 28 h (129$\times$ more) for NetSSM (batch size of 1). We report the accuracy for different heuristics in Table II.

TABLE II
EVALUATION SCORES FOR DOMAIN CHECKS OF THE REAL/GENERATED FLOWS (UPPER) AND WHEN USED AS TRAINING DATA FOR FLOW CLASSIFICATION (LOWER). c = NUMBER OF CLASSES

	Bayesian Network	NetSSM	Real
Domain Checks			
TCP handshake observed	0.94	**0.96**	**0.96**
No data before handshake	**1.00**	**1.00**	**1.00**
No duplicate handshake	0.89	**0.93**	**0.93**
TCP termination observed	0.96	**0.99**	**0.99**
No data after termination	0.57	0.82	**0.83**
F1 Macro			
CIC-IDS-2017 ($c = 14$)	0.29 ± 0.03	0.72 ± 0.05	$\mathbf{0.80 \pm 0.02}$
ISCX-VPN-2016 ($c = 7$)	0.25 ± 0.04	0.76 ± 0.08	$\mathbf{0.78 \pm 0.07}$

The results show that the generated flows from both models mostly comply with the patterns of typical TCP flows. Similar to the previous test, the generated flows of the BN sometimes continue even after a TCP termination is observed. The scores of NetSSM are close to the real data.

Traffic Classification Utility: A primary use case of the synthetic data is for training of ML models on downstream task. To evaluate whether the generated flows suffice as training data, we compare the performance of Random Forest flow classifiers after supervised trained on the generated/real data. For each flow, we extracted a feature vector of 22 common statistical values [10] (e.g., number of packets send) and use their seed packet's protocol/appl. labels (see **Setup**) as targets.

The results (Table II) show a decline in classifier performance for training on the generated flows, especially for BN-generated flows, but only slightly for NetSSM's flows.

IV. Conclusion

In this paper we presented a Bayesian network approach for network packet header synthesis, and an expert discrimination test to evaluate the authenticity of the generated flows by experts. While the BN-generated flows are more reliably identified than those from the byte-level generator, they are also found as real in 53% of the times, indicating that structural dependencies capture significant protocol semantics. Further, domain checks show a high compliance with TCP session behavior–something similar models struggle with. Despite a current decline in downstream classifier performance, the BN requires just 0.3% and 0.8% of the training and inference time of NetSSM, respectively. These results highlight a promising research direction; our future work will focus on narrowing the fidelity gap and evaluating the utility of BN-generated traffic for training ML-based network security tools.

ACKNOWLEDGMENT

This project was funded by German Federal Ministry of Research, Technology, and Space (BMFTR), Program "KI4KMU", Project "FlowLens" (ID: 01IS24017B).

REFERENCES

[1] Andrew Chu, Xi Jiang, Shinan Liu, Arjun Bhagoji, Francesco Bronzino, Paul Schmitt, and Nick Feamster. Netssm: Multi-flow and state-aware network trace generation using state-space models. *arXiv preprint arXiv:2503.22663*, 2025.

[2] Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality, 2024.

[3] Gerard Draper-Gil, Arash Habibi Lashkari, Mohammad Saiful Islam Mamun, and Ali A Ghorbani. Characterization of encrypted and vpn traffic using time-related. In *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)*, pages 407–414, 2016.

[4] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*, 2017.

[5] Jorge Luis Guerra, Carlos Catania, and Eduardo Veas. Datasets are not enough: Challenges in labeling network traffic. *Computers & Security*, 120:102810, 2022.

[6] Arash Habibi Lashkari, Gerard Draper Gil, Mohammad Saiful Islam Mamun, and Ali A. Ghorbani. Characterization of tor traffic using time based features. In *Proceedings of the 3rd International Conference on Information Systems Security and Privacy - ICISSP*, pages 253–262. INSTICC, SciTePress, 2017.

[7] Jonas Höchst, Lars Baumgärtner, Matthias Hollick, and Bernd Freisleben. Unsupervised traffic flow classification using a neural autoencoder. In *2017 IEEE 42Nd Conference on local computer networks (LCN)*, pages 523–526. IEEE, 2017.

[8] Xuying Meng, Chungang Lin, Yequan Wang, and Yujun Zhang. Netgpt: Generative pretrained transformer for network traffic. *arXiv preprint arXiv:2304.09513*, 2023.

[9] Jian Qu, Xiaobo Ma, and Jianfeng Li. Trafficgpt: Breaking the token barrier for efficient long traffic analysis and generation. *arXiv preprint arXiv:2403.05822*, 2024.

[10] Ola Salman, Imad H Elhajj, Ayman Kayssi, and Ali Chehab. A review on machine learning–based approaches for internet traffic classification. *Annals of Telecommunications*, 75(11):673–710, 2020.

[11] Adrien Schoen, Gregory Blanc, Pierre-François Gimenez, Yufei Han, Frédéric Majorczyk, and Ludovic Me. A tale of two methods: unveiling the limitations of gan and the rise of bayesian networks for synthetic network traffic generation. In *2024 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 273–286. IEEE, 2024.

[12] Iman Sharafaldin, Arash Habibi Lashkari, Ali A Ghorbani, et al. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp*, 1(2018):108–116, 2018.

[13] Ali Shiravi, Hadi Shiravi, Mahbod Tavallaee, and Ali A Ghorbani. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *computers & security*, 31(3):357–374, 2012.

[14] Stanislav Špaček, Petr Velan, Pavel Čeleda, and Daniel Tovarňák. Encrypted web traffic dataset: Event logs and packet traces. *Data in Brief*, 42:108188, 2022.

[15] Zahra Taghiyarrenani and Hamed Farsi. Domain adaptation with maximum margin criterion with application to network traffic classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 159–169. Springer, 2022.

[16] The Wireshark Foundation. Wireshark, 2026. Network protocol analyzer.

[17] UNIBS. Available traces with associated ground truth. http://netweb.ing.unibs.it/~ntw/tools/traces/. Accessed: 2025-10-23.

[18] Qineng Wang, Chen Qian, Xiaochang Li, Ziyu Yao, Gang Zhou, and Huajie Shao. Lens: A foundation model for network traffic. *arXiv preprint arXiv:2402.03646*, 2024.

[19] Yucheng Yin, Zinan Lin, Minhao Jin, Giulia Fanti, and Vyas Sekar. Practical gan-based synthetic ip header trace generation using netshare. In *Proceedings of the ACM SIGCOMM 2022 Conference*, pages 458–472, 2022.

[20] Shiyuan Zhang, Tong Li, Depeng Jin, and Yong Li. Netdiff: a service-guided hierarchical diffusion model for network flow trace generation. *Proceedings of the ACM on Networking*, 2(CoNEXT3):1–21, 2024.