FIDELITY IS NOT ENOUGH: TABULAR ANOMALY DETECTION UNDER SYNTHETIC DISTRIBUTION SHIFT

Anonymous authors
Paper under double-blind review

ABSTRACT

Real network anomaly data are difficult to collect and label, which makes it difficult to support benchmarking and motivates the growing use of synthetic data for developing and evaluating network anomaly detectors. However, generated data can preserve distributional properties of real data while altering the feature relationships used by a detector. We investigate whether detectors trained exclusively on real data remain effective on generated data. We test this across multiple tabular generators, datasets, and detection models. We find that distributional fidelity does not reliably predict detection performance: detectors can behave differently across generators even when generated data closely match the real-data distributions, and detector rankings on real data can change substantially on generated data. Filtering data by predicted provenance is similarly unreliable, as it can miss attacks from unseen generators while rejecting real benign traffic. Motivated by these findings, we introduce D^{2}Former, a behavioural anomaly detector that encodes features relative to quantiles of the real training distribution, models cross-feature interactions with self-attention, and aggregates encoder outputs using statistical summaries beyond a single classification token. Across three seeds, D²Former and TabTransformer have comparable overall transfer, while D²Former performs better on BCCC, where the largest transfer losses occur. In these DDoS datasets, generated data provide useful stress tests but do not reliably substitute for held-out real data when evaluating detectors. We therefore argue that original real data should remain the basis for developing and validating network anomaly detectors, with results on generated data treated as supplementary evidence rather than a replacement for evaluation on real data.

1 INTRODUCTION

Real anomaly data are often difficult to collect, label, and share. Synthetic tabular data are therefore used to support classifier development and evaluation. Their quality is usually measured by comparing their feature distributions, correlations, and joint distributions with real data (Xu et al., 2019; Kotelnikov et al., 2023; Kim et al., 2023). However, statistical similarity does not necessarily mean that synthetic data preserve the feature relationships used for classification.

This creates a direct evaluation problem. A generator may produce data that appear similar to real data while changing the boundary between normal and anomalous samples. A classifier that performs well on real data may then perform differently on generated data. Its ranking relative to other classifiers may also depend on the generator. We refer to this problem as *generator-induced distribution shift* and ask whether synthetic data can reliably replace real data when evaluating anomaly classifiers.

We study this question using DDoS classification as the experimental case. Each classifier is trained only on real data and then tested, without retraining or adaptation, on benign and attack samples produced by different generators. We evaluate synthetic data at three levels. First, we measure their marginal, correlation, and joint-distribution fidelity. Second, we test whether classifiers preserve their performance and ranking on generated data. Third, we evaluate whether provenance filtering can identify and reject generated attacks from an unseen generator without rejecting real benign data. The study covers five real-world DDoS datasets, five tabular generators, and five classifiers.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

We also study whether a detector designed around the real training distribution can transfer more reliably to generated data. For this purpose, we propose D^{2}Former. It represents numerical features relative to quantiles estimated from real training data, uses self-attention to model relationships between features, and combines the classification token with statistical summaries of the encoded features. D^{2}Former is trained only on real data and does not require synthetic training samples or generator-specific adaptation.

Our results show that fidelity does not reliably predict classifier performance on generated data. Classifier rankings change across generators, and no classifier is consistently best across all datasets and generated distributions. Provenance filtering does not solve this problem because it may miss attacks from an unseen generator or reject real benign data. D^{2}Former improves transfer under the largest observed shift, but its performance still depends on the generator. These results show that synthetic data are not a reliable replacement for real data in DDoS classifier development and evaluation. Synthetic data should instead be used as additional generator-specific test data.

Our main contributions are as follows:

- We study generator-induced distribution shift in synthetic tabular network traffic using distributional fidelity, train-on-real/test-on-synthetic transfer, and cross-generator provenance filtering. This evaluation tests whether synthetic data preserve the behavior of detectors trained on real traffic, rather than relying only on statistical similarity to the real distribution.
- We show that distributional fidelity does not reliably predict detector performance on synthetic data, detector rankings change across generators, and provenance filtering can fail to distinguish generated attacks from real benign traffic. These results show that synthetic data can be used to test detector robustness across generators, but should not replace held-out real data for model selection and validation.
- We introduce D^{2}Former, a Transformer-based detector for tabular network traffic that combines quantile-based feature tokenization learned from real data, self-attention for feature interactions, and statistical aggregation of encoder outputs. D^{2}Former achieves competitive transfer performance and stronger transfer performance under generator-induced distribution shift.

2 Related Work

Synthetic tabular data are generated using adversarial, diffusion, latent-variable, and flow-based models (Xu et al., 2019; Kotelnikov et al., 2023; Zhang et al., 2024; Shi et al., 2025; Guzmán-Cordero et al., 2025). Their quality is commonly assessed through feature distributions, correlations, real-versus-synthetic classification, downstream utility, and privacy. Recent work also combines multiple generators because a single model may not represent all regions of a complex tabular distribution equally well (Faul et al., 2026). Downstream utility is usually evaluated by training on synthetic data or adding synthetic samples to the training set. Synthetic augmentation, however, does not consistently improve classification on network data (Wolf et al., 2024). These protocols measure whether synthetic data are useful for training. They do not show whether synthetic data provide a reliable test distribution for classifiers trained on real data.

This question connects synthetic-data evaluation to distribution shift. Existing benchmarks study shifts across domains, datasets, or time (Koh et al., 2021; Gulrajani & Lopez-Paz, 2021; Gardner et al., 2023). Drift-Resilient TabPFN addresses temporal shifts in tabular data, while Iterative Robust Satisficing aims to limit performance loss under uncertain target distributions (Helli et al., 2024; Ağırman et al., 2026). In these settings, the target shift already exists and the goal is to improve model robustness. In our setting, the target distribution is deliberately produced by a generator fitted to the source data while retaining the same feature schema and class labels. Out-of-distribution and provenance methods can identify samples that differ from the source distribution or appear to be generated (Corsini & Yang, 2023; Guo et al., 2025; Kindji et al., 2025). However, identifying sample origin does not determine whether a sample is normal or anomalous. Provenance detection therefore cannot establish that the class boundary has been preserved.

This distinction is important for anomaly detection. Learning-based network detectors include convolutional, recurrent, and transformer models (Yuan et al., 2017; Doriguzzi-Corin et al., 2020; Gor-

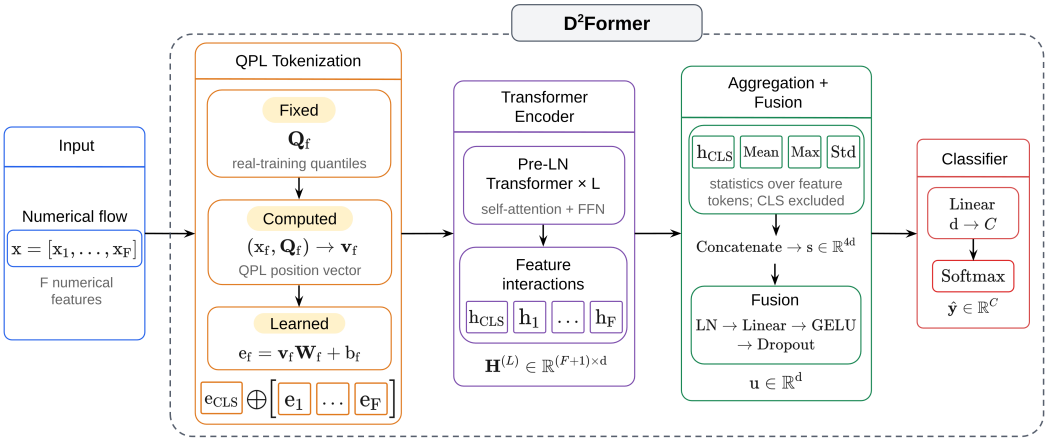

Figure 1: Architecture of D^{2}Former. Quantile boundaries are estimated from the real training partition and remain fixed during evaluation. Mean, maximum, and standard deviation are computed over the encoded feature tokens, excluding the classification token.

ishniy et al., 2021). Recent work such as ALoRa-T uses self-attention for multivariate time-series anomaly detection and localization (Shimillas et al., 2026). Network detectors are generally evaluated within one dataset or across datasets collected under different conditions (Sommer & Paxson, 2010; Apruzzese et al., 2023; Cantone et al., 2024). Cross-dataset evaluation may change the collection process, feature space, and class definitions together, making the source of performance changes difficult to isolate. We instead use DDoS classification as an experimental case, keep the source data, feature schema, and labels fixed, and vary only the generator. This allows us to examine how generated distributions affect classifier performance, model rankings, and provenance filtering.

3 D^{2}FORMER

D^{2}Former is designed for numerical tabular classification when test data may come from real or generated distributions. It makes three design choices. First, QPL tokenization represents each feature relative to its real training distribution. Second, self-attention models relationships among features because a class may be defined by a combination of values rather than one value alone. Third, statistical aggregation retains information across all encoded feature tokens instead of relying only on a classification token.

Figure 1 shows the architecture. Each feature is converted into a QPL position vector and projected into a feature token. A Transformer encoder processes the feature tokens together with a learned classification token. The final classification token is combined with the mean, maximum, and standard deviation of the encoded feature tokens, then passed through a fusion block and classifier. All projections, encoder layers, fusion parameters, and classifier parameters are trained jointly.

Let $\mathbf{x} = [x_1, \ldots, x_F] \in \mathbb{R}^F$ denote an input with F retained numerical features, and let C denote the number of classes.

3.1 DISTRIBUTION-RELATIVE FEATURE TOKENIZATION

Numerical features may have very different scales and distributions. A raw value does not indicate where it lies within the training distribution of its feature. We use quantile piecewise-linear (QPL) embeddings to provide this information (Gorishniy et al., 2022).

For feature f, let

$$q_{f,0} < q_{f,1} < \cdots < q_{f,B_f} \tag{1}$$

be the unique quantile boundaries estimated from the real training partition. Duplicate boundaries are removed because repeated values can produce identical quantiles. The number of intervals B_f may therefore differ across features. The boundaries remain fixed during validation and testing.

For interval $k \in \{0, \ldots, B_f - 1\}$, the QPL component is

$$v_{f,k} = \mathrm{clip}\left(\frac{x_f - q_{f,k}}{q_{f,k+1} - q_{f,k}}, 0, 1\right). \tag{2}$$

Intervals below x_f receive one, intervals above x_f receive zero, and the interval containing x_f records its relative position between the two boundaries. Clipping handles values outside the range observed in the real training partition.

The components form

$$\mathbf{v}_f = [v_{f,0}, \ldots, v_{f,B_f-1}] \in \mathbb{R}^{B_f}. \tag{3}$$

The vector $\mathbf{v}_f$ is determined by x_f and the fixed quantile boundaries; it is not learned.

A feature-specific linear projection converts $\mathbf{v}_f$ into a d-dimensional token:

$$\mathbf{e}_f = \mathbf{v}_f \mathbf{W}_f + \mathbf{b}_f, \qquad \mathbf{W}_f \in \mathbb{R}^{B_f \times d}, \quad \mathbf{b}_f \in \mathbb{R}^d. \tag{4}$$

Each feature has its own projection parameters. The projections retain feature identity after the tokens are mapped to the same dimension, so positional embeddings are not used.

A learned classification token $\mathbf{e}_{\mathrm{CLS}}$ is prepended to the feature tokens:

$$\mathbf{H}^{(0)} = \begin{bmatrix} \mathbf{e}_{\mathrm{CLS}} \\ \mathbf{e}_1 \\ \vdots \\ \mathbf{e}_F \end{bmatrix} \in \mathbb{R}^{(F+1)\times d}. \tag{5}$$

3.2 Feature-Interaction Encoder

Class labels can depend on relationships among several features. A value that appears normal on its own may be informative when considered with other values. We use self-attention to model these relationships, following Transformers for numerical tabular data (Gorishniy et al., 2021).

A pre-layer-normalized Transformer encoder with L layers processes the token sequence:

$$\mathbf{H}^{(L)} = T_\theta\left(\mathbf{H}^{(0)}\right) = \begin{bmatrix} \mathbf{h}_{\mathrm{CLS}} \\ \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_F \end{bmatrix} \in \mathbb{R}^{(F+1)\times d}. \tag{6}$$

Here, $\mathbf{h}_{\mathrm{CLS}}$ is the encoded classification token and $\mathbf{h}_f$ is the encoded representation of feature f. Self-attention allows each output token to depend on all features in the same sample.

3.3 Statistical Aggregation and Fusion

The classification token provides one summary of the encoded sample. We retain it and also summarize the feature tokens directly using their element-wise mean, maximum, and standard deviation:

$$\boldsymbol{\mu}_H = \frac{1}{F}\sum_{f=1}^{F} \mathbf{h}_f, \qquad \mathbf{m}_H = \max_{1\le f\le F} \mathbf{h}_f, \qquad \boldsymbol{\sigma}_H = \mathrm{Std}\left(\mathbf{h}_1, \ldots, \mathbf{h}_F\right). \tag{7}$$

The classification token is excluded from these statistics. The mean describes the average response across feature tokens, the maximum retains the largest response in each dimension, and the standard deviation records variation among the feature tokens.

The four representations are concatenated:

$$\mathbf{s} = [\mathbf{h}_{\mathrm{CLS}}; \boldsymbol{\mu}_H; \mathbf{m}_H; \boldsymbol{\sigma}_H] \in \mathbb{R}^{4d}. \tag{8}$$

A fusion block maps $\mathbf{s}$ back to d dimensions:

$$\mathbf{u} = \mathrm{Dropout}\left(\mathrm{GELU}\left(\mathrm{LN}(\mathbf{s})\mathbf{W}_{\mathrm{fuse}} + \mathbf{b}_{\mathrm{fuse}}\right)\right), \tag{9}$$

where

$$\mathbf{W}_{\mathrm{fuse}} \in \mathbb{R}^{4d\times d}, \qquad \mathbf{b}_{\mathrm{fuse}} \in \mathbb{R}^{d}. \tag{10}$$

We use $d = 512$. Thus, $\mathbf{s}$ has dimension 2048 and $\mathbf{u}$ has dimension 512. The classifier produces the class probabilities

$$\hat{\mathbf{y}} = \mathrm{softmax}\left(\mathbf{u}\mathbf{W}_{\mathrm{cls}} + \mathbf{b}_{\mathrm{cls}}\right) \in \mathbb{R}^{C}, \tag{11}$$

where

$$\mathbf{W}_{\mathrm{cls}} \in \mathbb{R}^{d\times C}, \qquad \mathbf{b}_{\mathrm{cls}} \in \mathbb{R}^{C}. \tag{12}$$

4 Experimental Methodology

4.1 Experimental Setup

The experiments examine how generated distributions affect classifiers trained on real data. We use five flow-level DDoS datasets: APA-DDoS (Kumbam, 2023), SDN (Ahuja et al., 2020), CICDoS2019 (Sharafaldin et al., 2019), CRCDDoS2022 (Hadi et al., 2022), and BCCC (Shafi et al., 2024). These datasets cover binary and multiclass classification and differ in size and class imbalance. Dataset statistics are reported in Appendix A.1.

Each dataset is divided into stratified training, validation, and test partitions containing 59.5%, 10.5%, and 30% of the samples, respectively, using seed 42. Feature selection, scaling, quantile estimation, class weighting, and training-sample selection are determined solely from the real training partition. The validation partition is used for checkpoint selection, while the held-out real test partition provides the in-distribution reference. The preprocessing state and selected model parameters remain fixed during evaluation.

CTGAN, TabDDPM, TabDiff, TabSyn, and TabbyFlow are fitted separately to each source dataset and used to produce class-conditional samples. The generators are fitted on the complete source datasets because the objective is to study classifier behavior on generated distributions, rather than generator generalization to held-out records. For each combination of dataset and generator, one fixed synthetic test set is saved and evaluated by every detector. The five datasets and five generators produce 25 combinations, all of which are included in the fidelity and detector evaluations. Generator configurations and sampling procedures are provided in Appendix A.2. A separate Random Forest TSTR evaluation uses its own stratified real-data split and is reported in Appendix A.6.

We compare D^{2}Former with an MLP, TabTransformer, DistilBERT, and TinySEED (Zhang et al., 2026). Each detector is trained and selected using only real data. Its checkpoint, preprocessing state, and decision rule are then applied unchanged to the real and generated test sets. No retraining or generator-specific calibration is performed. D^{2}Former is trained with class-weighted cross-entropy and MixUp with $\alpha = 0.1$ (Zhang et al., 2018). Architecture, optimization, and checkpoint-selection settings are reported in Appendix A.4.

4.2 Evaluation Protocols

We evaluate generated data through distributional fidelity, cross-generator provenance filtering, and classifier transfer. These protocols measure statistical similarity, detectability of synthetic origin, and preservation of the class boundary learned from real data.

Distributional fidelity. For feature f, let P_f and Q_f denote its real and generated distributions. Marginal fidelity is measured by the normalized Wasserstein-1 distance

$$\widehat{W}_1(f) = \frac{W_1(P_f, Q_f)}{q_{0.99}(P_f) - q_{0.01}(P_f)}, \tag{13}$$

where $q_\alpha(P_f)$ is the α-quantile of the real distribution. We report the median across features. Pairwise dependence is measured by

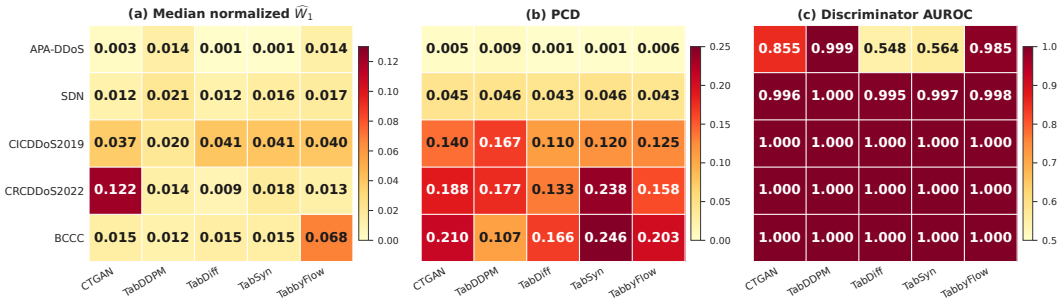

Figure 2: Generator fidelity across the 25 dataset–generator combinations. The panels report (a) median normalized Wasserstein-1 distance, (b) pairwise correlation difference (PCD), and (c) real–synthetic discriminator AUROC. Lower normalized Wasserstein-1 distance and PCD indicate closer agreement, while discriminator AUROC closer to 0.5 indicates weaker real–synthetic separability.

$$\mathrm{PCD} = \frac{1}{F^2} \sum_{i=1}^{F} \sum_{j=1}^{F} \left| \rho_{ij}^{r} - \rho_{ij}^{s} \right|, \tag{14}$$

where ρ^r and ρ^s are the real and generated Spearman correlation matrices. Joint-distribution separability is measured by the AUROC of an XGBoost classifier trained to distinguish real from generated samples. Lower $\widehat{W}_1$ and PCD indicate closer agreement, while an AUROC near 0.5 indicates weak separability between real and generated data. Detailed estimation settings and feature-level comparisons are provided in Appendix A.5.

Cross-generator provenance filtering. We evaluate provenance filtering on APA-DDoS and BCCC. In each round, the classifier is trained to distinguish real attacks from synthetic attacks produced by four generators and tested on attacks from the remaining generator. The threshold is selected on real validation attacks to limit their rejection rate to 1%. We report real-synthetic AUROC, unseen-attack catch rate, and the rejection rates for held-out real attacks and real benign samples. Full sampling, filtering, and threshold-selection details are provided in Appendix A.7.

Classifier transfer. Each classifier is evaluated under train-on-real, test-on-real (TRTR) and train-on-real, test-on-synthetic (TRTS). TRTR uses the held-out real test partition, while TRTS applies the same classifier to generated benign and attack samples. Balanced accuracy is the primary metric. For generator g, the transfer gap is defined as $\Delta_g = \mathrm{BA}_{\mathrm{TRTR}} - \mathrm{BA}^{g}_{\mathrm{TRTS}}$. We also compare classifier rankings across real and generated test distributions. Macro-F1 and per-class results are reported in Appendix A.8.

5 Results and Discussion

5.1 Fidelity Does Not Determine Classifier Transfer

Figure 2 reports fidelity for all 25 dataset–generator combinations. The strongest agreement occurs on APA-DDoS. TabDiff and TabSyn both obtain a normalized Wasserstein-1 distance of 0.001 and a PCD of 0.001. Their discriminator AUROCs are also close to random classification, at 0.548 and 0.564. In contrast, CTGAN, TabbyFlow, and TabDDPM are more easily separated from the real data, with AUROCs of 0.855, 0.985, and 0.999, respectively. Generator choice therefore affects fidelity even when every generator is fitted to the same source dataset.

The results differ on the other datasets. Generated data are almost perfectly distinguishable from real data on CICDDoS2019, CRCDDoS2022, and BCCC. On SDN, discriminator AUROC ranges from 0.995 to 1.000, although the normalized Wasserstein-1 distance remains between 0.012 and 0.021 and PCD remains between 0.043 and 0.046. The generated SDN data can therefore reproduce

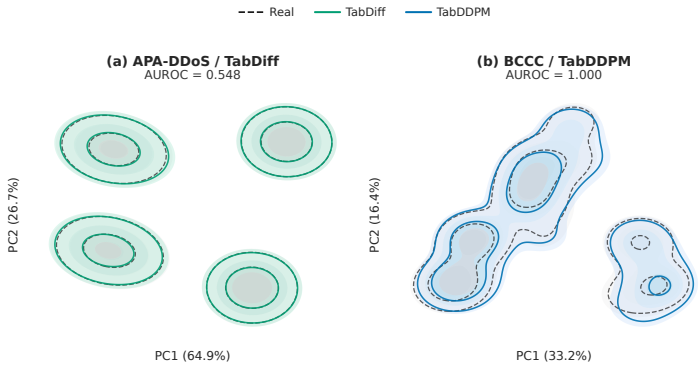

Figure 3: PCA projections of real and generated flows for (a) APA-DDoS–TabDiff and (b) BCCC–TabDDPM. Within each dataset, one PCA projection is fitted jointly to the real sample and samples from all five generators after min–max scaling. Dashed contours denote real flows and solid contours denote generated flows. AUROC is measured by the discriminator in the original feature space.

Table 1: Unseen generator provenance results on APA-DDoS. RA and RB denote rejection rates on real attack and real benign flows.

Unseen generator	AUROC ↑	Catch (%) ↑	RA (%) ↓	RB (%) ↓
CTGAN	0.564	16.66	0.004	0.00
TabDDPM[†]	0.994	0.53	0.027	0.00
TabDiff	0.500	0.02	0.022	0.00
TabSyn	0.500	0.02	0.022	0.00
TabbyFlow	0.986	95.63	0.013	100.00

[†]The TabDDPM result is based on only 187 generated attacks that satisfied the validity constraints.

many individual feature distributions and pairwise relationships while still producing records that differ from real data when all features are considered together.

Figure 3 illustrates two different cases. APA-DDoS with TabDiff shows substantial overlap between real and generated samples, consistent with its discriminator AUROC of 0.548. BCCC with TabDDPM shows clearer separation, consistent with an AUROC of 1.000. The PCA plots are used only for visualization. All fidelity metrics are computed using the original features rather than the two-dimensional PCA projections.

Unlike the provenance experiment, the fidelity discriminator is evaluated on generators represented during its training. On BCCC, its AUROC is 1.000 for all five generators, while D^{2}Former's TRTS balanced accuracy ranges from 0.5516 to 0.8497. On APA-DDoS, discriminator AUROC ranges from 0.548 to 0.999, while D^{2}Former, MLP, and TabTransformer each achieve 1.000 across all five generators. Thus, discriminator AUROC alone does not rank these generated test sets by detector performance.

5.2 Provenance Filtering Does Not Generalize Reliably

Tables 1 and 2 report performance when each generator is excluded from training and used only for testing. AUROC measures how well the filter ranks real and generated attacks. Catch rate measures how many attacks from the unseen generator are rejected at the selected threshold.

On APA-DDoS, the filter fails to recognize attacks from TabDiff and TabSyn, for which AUROC is 0.500 and catch rate is only 0.02%. CTGAN is also difficult to identify, with an AUROC of 0.564 and a catch rate of 16.66%. In these cases, training on four generators does not enable the filter to distinguish attacks from the unseen generator reliably.

A high AUROC does not necessarily produce a useful filter. For APA-DDoS–TabDDPM, AUROC reaches 0.994, but only 0.53% of attacks cross the selected threshold. The result is based on only

Table 2: Unseen generator provenance results on BCCC. RA and RB denote rejection rates on real attack and real benign flows.

Unseen generator	AUROC ↑	Catch (%) ↑	RA (%) ↓	RB (%) ↓
CTGAN	0.994	97.63	0.71	10.19
TabDDPM	0.990	95.21	0.61	10.29
TabDiff	0.992	96.51	0.37	10.27
TabSyn	0.870	42.25	0.65	16.69
TabbyFlow	1.000	99.97	0.40	10.21

187 constraint-valid attacks and should therefore be interpreted cautiously. For TabbyFlow, the filter catches 95.63% of attacks but also rejects every real benign flow. High synthetic-attack rejection is not useful if legitimate traffic is rejected at the same time.

The BCCC filter generalizes better to unseen generators. It catches more than 95% of attacks from CTGAN, TabDDPM, TabDiff, and TabbyFlow. However, it rejects between 10.19% and 16.69% of real benign flows. TabSyn also remains difficult, with a catch rate of 42.25%.

These errors follow from the objective of the filter. It is trained to distinguish real attacks from generated attacks, not to distinguish benign traffic from attacks. A feature pattern that indicates synthetic origin does not necessarily indicate malicious behaviour. Likewise, real benign traffic can differ from the real attacks used during training and may consequently be classified as generated. Provenance filtering can identify some generator artefacts, but it cannot replace an anomaly detector based on traffic behaviour.

5.3 Classifier Performance Depends on the Generator

Table 3 reports balanced accuracy on held-out real data (TRTR) and generated data (TRTS). For seed 42, D^{2}Former achieves the highest overall mean TRTS accuracy at 0.8919, followed by TabTransformer at 0.8848 and the MLP at 0.8323. D^{2}Former is best or tied for best on 14 of the 25 dataset–generator combinations. Its largest advantage appears on BCCC, where its mean TRTS accuracy is 0.7083, compared with 0.6887 for TabTransformer and 0.4588 for the MLP.

However, no classifier performs best in every setting. On BCCC, TabTransformer ranks first on real data, but D^{2}Former performs better on data from all five generators. On SDN, D^{2}Former ranks first on real data, while TabTransformer has the highest mean TRTS accuracy. Performance also changes between generators. For example, D^{2}Former achieves 0.8497 on BCCC data generated by TabDDPM but only 0.5516 on data generated by TabSyn. Classifier performance and ranking therefore depend on which generator produces the test data.

Across three seeds, D^{2}Former and TabTransformer achieve comparable overall mean TRTS accuracies of 0.8859 ± 0.0062 and 0.8860 ± 0.0033, respectively. On BCCC, D^{2}Former achieves 0.6983 ± 0.012, compared with 0.6700 ± 0.019 for TabTransformer and 0.3829 ± 0.066 for the MLP. D^{2}Former therefore has higher observed mean TRTS accuracy on BCCC, but it is not consistently superior across all datasets and generators.

6 Conclusion

In this work, we investigated whether synthetic tabular data can be used to evaluate anomaly detectors trained only on real data, using DDoS detection as an experimental case. Across five public DDoS datasets and five advanced tabular generators, high fidelity did not imply stable TRTS balanced accuracy or detector rankings across generators. Furthermore, provenance filtering did not consistently catch attacks from unseen generators without also rejecting real benign flows. These results show that, in the DDoS settings studied here, predicted synthetic origin cannot replace a detector that distinguishes benign traffic from attacks.

We proposed D^{2}Former, which computes quantiles from real training data and uses them to tokenize input flows relative to the real-data distribution. It achieves the highest mean TRTS balanced accuracy with seed 42 and a smaller performance drop from TRTR to TRTS on BCCC. Across three

Table 3: TRTR and TRTS balanced accuracy. Mean TRTS averages the five generators, and panel (b) reports macro-averages across datasets. D^{2}Former, MLP, and TabTransformer use seed 42; the other detectors use fixed evaluations. Bold indicates the highest result, including ties.

(a) Per-dataset TRTR and TRTS results

Dataset	Detector	TRTR ↑	TRTS balanced accuracy ↑					Mean TRTS ↑
			CTGAN	TabDDPM	TabDiff	TabSyn	TabbyFlow	
APA-DDoS	D^{2}Former	**1.0000**	1.0000	1.0000	1.0000	1.0000	1.0000	**1.0000**
	MLP	**1.0000**	1.0000	1.0000	1.0000	1.0000	1.0000	**1.0000**
	TabTransformer	**1.0000**	1.0000	1.0000	1.0000	1.0000	1.0000	**1.0000**
	DistilBERT	0.6667	0.6667	0.6536	0.6365	0.6373	0.6441	0.6476
	TinySEED	0.6667	0.6667	0.3382	0.3667	0.3660	0.3465	0.4168
SDN	D^{2}Former	**0.9991**	0.9196	0.9533	0.9743	0.8942	0.9244	0.9331
	MLP	0.9900	0.8541	0.9674	0.9724	0.9380	0.9432	0.9350
	TabTransformer	0.9915	0.9321	0.9667	0.9656	0.9244	0.9413	**0.9460**
	DistilBERT	0.9918	0.5944	0.5669	0.6347	0.5461	0.6335	0.5951
	TinySEED	0.9942	0.5760	0.5928	0.5493	0.5930	0.6288	0.5880
CICDDoS2019	D^{2}Former	**0.8856**	0.8661	0.7973	0.8621	0.8807	0.8022	**0.8417**
	MLP	0.8511	0.7946	0.8533	0.8423	0.8450	0.8336	0.8338
	TabTransformer	0.8004	0.7928	0.8025	0.7862	0.7982	0.7834	0.7926
	DistilBERT	0.7435	0.6586	0.6485	0.6852	0.6870	0.6516	0.6662
	TinySEED	0.6745	0.5663	0.5405	0.6143	0.6505	0.5757	0.5894
CRCDDoS2022	D^{2}Former	0.9975	0.9263	0.9912	0.9912	0.9875	0.9860	0.9764
	MLP	**1.0000**	0.8938	0.9410	0.9853	0.9853	0.8643	0.9339
	TabTransformer	**1.0000**	0.9823	1.0000	1.0000	1.0000	1.0000	**0.9965**
	DistilBERT	0.9580	0.8076	0.5179	0.5835	0.5338	0.5413	0.5968
	TinySEED	0.9555	0.5944	0.5778	0.6866	0.6888	0.6177	0.6331
BCCC	D^{2}Former	0.9781	0.7346	0.8497	0.6676	0.5516	0.7380	**0.7083**
	MLP	0.9772	0.4750	0.4559	0.4702	0.4299	0.4631	0.4588
	TabTransformer	**0.9819**	0.7067	0.8353	0.6570	0.5314	0.7132	0.6887
	DistilBERT	0.8216	0.5899	0.5576	0.5574	0.5436	0.5594	0.5616
	TinySEED	0.8154	0.4079	0.4124	0.4071	0.4017	0.4048	0.4068

(b) Macro-average across the five datasets

Detector	Mean TRTR ↑	Mean TRTS ↑
D^{2}Former	**0.9721**	**0.8919**
MLP	0.9637	0.8323
TabTransformer	0.9548	0.8848
DistilBERT	0.8363	0.6135
TinySEED	0.8213	0.5268

Table 4: Three-seed results for the stochastic tabular detectors, reported as mean ± sample standard deviation over seeds 42–44. Overall Mean TRTS averages generators within each dataset and then averages across datasets. Bold indicates the highest result.

Detector	Mean TRTR ↑	Overall Mean TRTS ↑	BCCC Mean TRTS ↑
D^{2}Former	**0.9724±0.0004**	0.8859±0.0062	**0.6983±0.012**
MLP	0.9637±0.0001	0.8104±0.0196	0.3829±0.066
TabTransformer	0.9615±0.0059	**0.8860±0.0033**	0.6700±0.019

random seeds, however, D^{2}Former performs comparably to TabTransformer, and neither classifier is consistently best. Optimizing the anomaly detector therefore does not guarantee stable performance across generated distributions. Synthetic data can reveal sensitivity to specific generator-induced shifts, but they should supplement rather than replace real data in model selection and validation.

Future work will test whether anomalies produced by generators can also occur in real networks and measure their effects in controlled testbeds. We will also study how to collect real traffic containing new attacks, so that detectors can be improved without relying on patterns created by an individual generator.

AI USE STATEMENT

In this work, we used general-purpose generative AI tools for language polishing, grammar checking, and LaTeX formatting, and assistance with debugging data-processing code. The authors supplied the scientific content and made all decisions concerning the research questions, experimental design, analysis, interpretation, and conclusions.

The synthetic datasets examined in this study were produced by the five tabular generative models described in the paper. General-purpose generative AI tools were not used to collect or modify experimental data, perform the evaluation, produce visualizations, or generate or estimate numerical results.

The literature review and selection of cited works were conducted by the authors. The authors reviewed all AI-assisted text and code, tested the relevant code, and take responsibility for the final content of this work.

ETHICS STATEMENT

This study uses publicly available flow-level datasets and does not involve human participants or new data collection. All experiments were conducted offline, without sending attack traffic to operational networks. DDoS generation and detection have potential dual-use implications. This work focuses on evaluating detector reliability and does not provide instructions for attacking deployed systems.

REPRODUCIBILITY STATEMENT

The main paper describes the data splits, training procedure, evaluation protocols, and metrics. The appendix provides details of preprocessing, generator and detector configurations, sampling procedures, random seeds, checkpoint selection, and additional results. Each detector was evaluated using a fixed preprocessing state and model checkpoint. The same saved generated test sets were used across detectors without retraining or generator-specific adaptation. Code and experiment configurations are available at `https://github.com/Synthetic-Anomaly-Detection/d2former-tabular-shift.git`.

REFERENCES

Enes Ağırman, Artun Saday, and Cem Tekin. Iterative robust satisficing: Minimizing performance degradation under distribution shift. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL `https://openreview.net/forum?id=M0LllT3KX0`.

Nisha Ahuja, Gaurav Singal, and Debajyoti Mukhopadhyay. DDOS Attack SDN Dataset. Mendeley Data, 2020.

Giovanni Apruzzese, Pavel Laskov, and Johannes Schneider. SoK: Pragmatic assessment of machine learning for network intrusion detection. In *2023 IEEE 8th European Symposium on Security and Privacy*, pp. 592–614, 2023. doi: 10.1109/EuroSP57164.2023.00042.

Marco Cantone, Claudio Marrocco, and Alessandro Bria. Machine learning in network intrusion detection: A cross-dataset generalization study. *IEEE Access*, 12:144489–144508, 2024. doi: 10.1109/ACCESS.2024.3472907.

Andrea Corsini and Shanchieh Jay Yang. Are existing out-of-distribution techniques suitable for network intrusion detection? In *2023 IEEE Conference on Communications and Network Security (CNS)*, 2023. doi: 10.1109/CNS59707.2023.10288685.

Roberto Doriguzzi-Corin, Stuart Millar, Sandra Scott-Hayward, Jesús Martínez-del Rincón, and Domenico Siracusa. LUCID: A practical, lightweight deep learning solution for DDoS attack detection. *IEEE Transactions on Network and Service Management*, 17(2):876–889, 2020. doi: 10.1109/TNSM.2020.2971776.

Antoine Faul, Xiao Zhou, Ossi Räisä, Mihaela van der Schaar, and Cem Tekin. Modelling complex tabular datasets with a mixture of diverse generative models. *Transactions on Machine Learning Research*, 2026. URL https://openreview.net/forum?id=3y3mHAldp7.

Josh Gardner, Zoran Popovic, and Ludwig Schmidt. Benchmarking distribution shift in tabular data with TableShift. *Advances in Neural Information Processing Systems*, 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/a76a757ed479a1e6a5f8134bea492f83-Abstract-Datasets_and_Benchmarks.html.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. In *Advances in Neural Information Processing Systems*, volume 34, pp. 18932–18943, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/9d86d83f925f2149e9edb0ac3b49229c-Abstract.html.

Yury Gorishniy, Ivan Rubachev, and Artem Babenko. On embeddings for numerical features in tabular deep learning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24991–25004, 2022.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=lQdXeXDoWtI.

Kaiyu Guo, Zijian Wang, Tan Pan, Brian C. Lovell, and Mahsa Baktashmotlagh. Improving out-of-distribution detection via dynamic covariance calibration. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 21067–21082. PMLR, 2025. URL https://proceedings.mlr.press/v267/guo25m.html.

Andrés Guzmán-Cordero, Floor Eijkelboom, and Jan-Willem van de Meent. Exponential family variational flow matching for tabular data generation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 21516–21529. PMLR, 2025. URL https://proceedings.mlr.press/v267/guzman-cordero25a.html.

Hassan Jalil Hadi, Umer Hayat, Numan Musthaq, Faisal Bashir Hussain, and Yue Cao. Developing realistic distributed denial of service (DDoS) dataset for machine learning-based intrusion detection system. In *2022 9th International Conference on Internet of Things: Systems, Management and Security*, 2022. doi: 10.1109/IOTSMS58070.2022.10062034.

Kai Helli, David Schnurr, Noah Hollmann, Samuel Müller, and Frank Hutter. Drift-resilient TabPFN: In-context learning temporal distribution shifts on tabular data. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/b2e2774c8e76afe191b5bf518f5cb727-Abstract-Conference.html.

Jayoung Kim, Chaejeong Lee, and Noseong Park. STaSy: Score-based tabular data synthesis. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1mNssCWt_v.

G. Charbel N. Kindji, Lina M. Rojas Barahona, Elisa Fromont, and Tanguy Urvoy. Cross-table synthetic tabular data detection. In *Proceedings of the 1st Workshop on GenAI Content Detection (GenAIDetect)*, pp. 78–84, Abu Dhabi, UAE, January 2025. International Conference on Computational Linguistics. URL https://aclanthology.org/2025.genaidetect-1.5/.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton Earnshaw, Imran Haque, Sara M. Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. WILDS: A benchmark of in-the-wild distribution shifts. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5637–5664. PMLR, 2021. URL https://proceedings.mlr.press/v139/koh21a.html.

Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. TabDDPM: Modelling tabular data with diffusion models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 17564–17579. PMLR, 2023. URL https://proceedings.mlr.press/v202/kotelnikov23a.html.

Yashwanth Kumbam. APA-DDoS Dataset. Kaggle, 2023. URL https://www.kaggle.com/datasets/yashwanthkumbam/apaddos-dataset.

MohammadMoein Shafi, Arash Habibi Lashkari, Vicente Rodriguez, and Ron Nevo. Toward generating a new cloud-based distributed denial of service (DDoS) dataset and cloud intrusion traffic characterization. *Information*, 15(4):195, 2024. doi: 10.3390/info15040195.

Iman Sharafaldin, Arash Habibi Lashkari, Saqib Hakak, and Ali A. Ghorbani. Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy. In *2019 International Carnahan Conference on Security Technology*, pp. 1–8, 2019. doi: 10.1109/CCST.2019.8888419.

Juntong Shi, Minkai Xu, Harper Hua, Hengrui Zhang, Stefano Ermon, and Jure Leskovec. TabDiff: A mixed-type diffusion model for tabular data generation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=swvURjrt8z.

Charalampos Shimillas, Kleanthis Malialis, Konstantinos Fokianos, and Marios M. Polycarpou. Low rank transformer for multivariate time series anomaly detection and localization. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=ZtPIBpVojC.

Robin Sommer and Vern Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *2010 IEEE Symposium on Security and Privacy*, pp. 305–316, 2010. doi: 10.1109/SP.2010.25.

Maximilian Wolf, Julian Tritscher, Dieter Landes, Andreas Hotho, and Daniel Schlör. Benchmarking of synthetic network data: Reviewing challenges and approaches. *Computers & Security*, 145: 103993, 2024. doi: 10.1016/j.cose.2024.103993.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional GAN. In *Advances in Neural Information Processing Systems*, volume 32, pp. 7335–7345. Curran Associates, Inc., 2019.

Xiaoyong Yuan, Chuanhuang Li, and Xiaolin Li. DeepDefense: Identifying DDoS attack via deep learning. In *2017 IEEE International Conference on Smart Computing (SMARTCOMP)*, pp. 1–8, 2017. doi: 10.1109/SMARTCOMP.2017.7946998.

Hengrui Zhang, Jiani Zhang, Balasubramaniam Srinivasan, Zhengyuan Shen, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. Mixed-type tabular data synthesis with score-based diffusion in latent space. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=4Ay23yeuz0.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

Yinuo Zhang, Zulfiqar Ahmad Khan, Simin Nadjm-Tehrani, and Monowar Bhuyan. TinySEED: A lightweight transformer architecture with channel-attention for DDoS detection. In *Proceedings of the 41st ACM/SIGAPP Symposium on Applied Computing*, pp. 395–404. ACM, 2026. doi: 10.1145/3748522.3779832. URL https://doi.org/10.1145/3748522.3779832.

Table 5: Real datasets used in the experiments.

Dataset	Flows	Classes	Class composition	Imbalance
APA-DDoS	151K	3	Benign and two DDoS classes	$\approx$ 2:1
SDN	104K	2	Normal and attack	$\approx$ 3:2
CICDDoS2019	20.3M	8	Benign and seven DDoS classes	$\approx$ 143:1
CRCDDoS2022	860K	2	Normal and DDoS	$\approx$ 1270:1
BCCC	700K	3	Benign, suspicious, and attack	$\approx$ 7:1

Table 6: Tabular generators used in the experiments.

Generator	Family	Generation mechanism
CTGAN (Xu et al., 2019)	Conditional GAN	Adversarial learning
TabDDPM (Kotelnikov et al., 2023)	Diffusion	Mixed-type denoising
TabDiff (Shi et al., 2025)	Diffusion	Feature-wise denoising
TabSyn (Zhang et al., 2024)	Latent diffusion	Latent-space denoising
TabbyFlow (Guzmán-Cordero et al., 2025)	Normalizing flow	Invertible transformation

A Appendix

A.1 Dataset and Preprocessing Details

Table 5 summarizes the five real datasets. Imbalance is the ratio between the largest and smallest classes.

Row identifiers are assigned before preprocessing. Each dataset is divided into stratified training, validation, and test partitions containing 59.5%, 10.5%, and 30% of the samples, respectively, using seed 42. The same split manifests are used by all classifiers.

Feature selection, scaling, quantile estimation, class weighting, and training-sample selection are fitted only on the real training partition. The fitted preprocessing state is applied unchanged to the validation, real test, and generated test sets. The validation partition is used only for checkpoint selection. Neither real nor generated test samples influence classifier training or model selection.

A.2 Generator Configuration

Table 6 summarizes the five generators. A separate instance of each generator is fitted to every source dataset. Generation is conditioned on the class label so that the generated data retain the label space of the source dataset.

The generators are fitted on the complete source datasets. For each combination of dataset and generator, one class-balanced synthetic evaluation set is saved and reused by all classifiers. The per-class sampling budget is fixed within each dataset. All 25 combinations are included in both the fidelity and classifier evaluations.

Generated samples are not used to train, tune, or select the behavioral classifiers. For the provenance experiment, generated attacks are filtered using the predefined validity rules. Protocol values must belong to the permitted domain, numerical values must be finite, and packet-count and byte-count relationships must satisfy the required consistency checks. Only constraint-valid attacks are used for provenance training and testing.

A.3 Baseline Classifier Details

The baseline classifiers are an MLP, TabTransformer, DistilBERT, and TinySEED. TinySEED uses a standardized six-layer TinyBERT configuration with squeeze-and-excitation (Zhang et al., 2026). All baseline classifiers are trained and selected using only the real training and validation partitions. Their selected checkpoints, preprocessing states, and decision rules remain fixed during evaluation on real and generated data.

A.4 D^{2}Former Implementation Settings

Table 7 gives the architecture and training settings used for D^{2}Former.

Table 7: D^{2}Former architecture and training settings.

Setting	Value
Encoder layers L	8
Token dimension d	512
Attention heads H	8
FFN dimension d_{ff}	1,024
Maximum QPL intervals B	128
Aggregated representation dimension $4d$	2,048
Fused representation dimension	512
Dropout	0.15
Loss	Class-weighted cross-entropy
Optimizer	AdamW
Learning rate	2×10^{-4}
Weight decay	3×10^{-4}
Learning-rate schedule	Cosine decay
Batch size	256
Maximum epochs	300
MixUp α	0.1
Checkpoint criterion	Validation balanced accuracy
Training seeds	42, 43, and 44

D^{2}Former is trained on the real training partition using

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} w_c y_{i,c} \log \hat{y}_{i,c}, \tag{15}$$

where N is the number of training samples, C is the number of classes, w_c is the weight assigned to class c, and $y_{i,c}$ and $\hat{y}_{i,c}$ are the target and predicted probability, respectively. Class weights are determined using only the real training partition and remain fixed during training. MixUp is applied with $\alpha = 0.1$ (Zhang et al., 2018).

The checkpoint with the highest balanced accuracy on the real validation partition is retained. If several checkpoints reach the same value, the earliest is used. The selected model, retained features, preprocessing state, and quantile boundaries are then fixed for all real and generated test sets. Generated samples are not used for training or model selection, and no test-time adaptation or generator-specific calibration is performed.

A.5 Fidelity Estimation and Feature-Level Comparisons

Fidelity is estimated separately for each combination of dataset and generator. For every retained numerical feature, we calculate the normalized Wasserstein-1 distance between the real and generated samples. The median across features gives the marginal fidelity score reported in the main paper. Pairwise dependence is evaluated by the mean absolute difference between the real and generated Spearman correlation matrices. Joint separability is measured using the held-out AUROC of an XGBoost classifier trained to distinguish real from generated samples. The discriminator is used only as a fidelity measure and is not involved in anomaly detection.

The feature-level plots below provide a closer view of the marginal differences summarized by the normalized Wasserstein-1 score. For each dataset, we show representative features with clear differences across generators. Kernel density estimates compare the shape of each distribution, while quantile–quantile plots show where generated values depart from the real-data quantiles.

For APA-DDoS, TabDiff and TabSyn closely reproduce the displayed marginal distributions. Both obtain a median normalized Wasserstein-1 distance of 0.001 and a pairwise correlation difference of 0.001. However, their discriminator AUROCs are 0.548 and 0.564, respectively, showing that small marginal and pairwise differences do not imply complete agreement in the joint distribution.

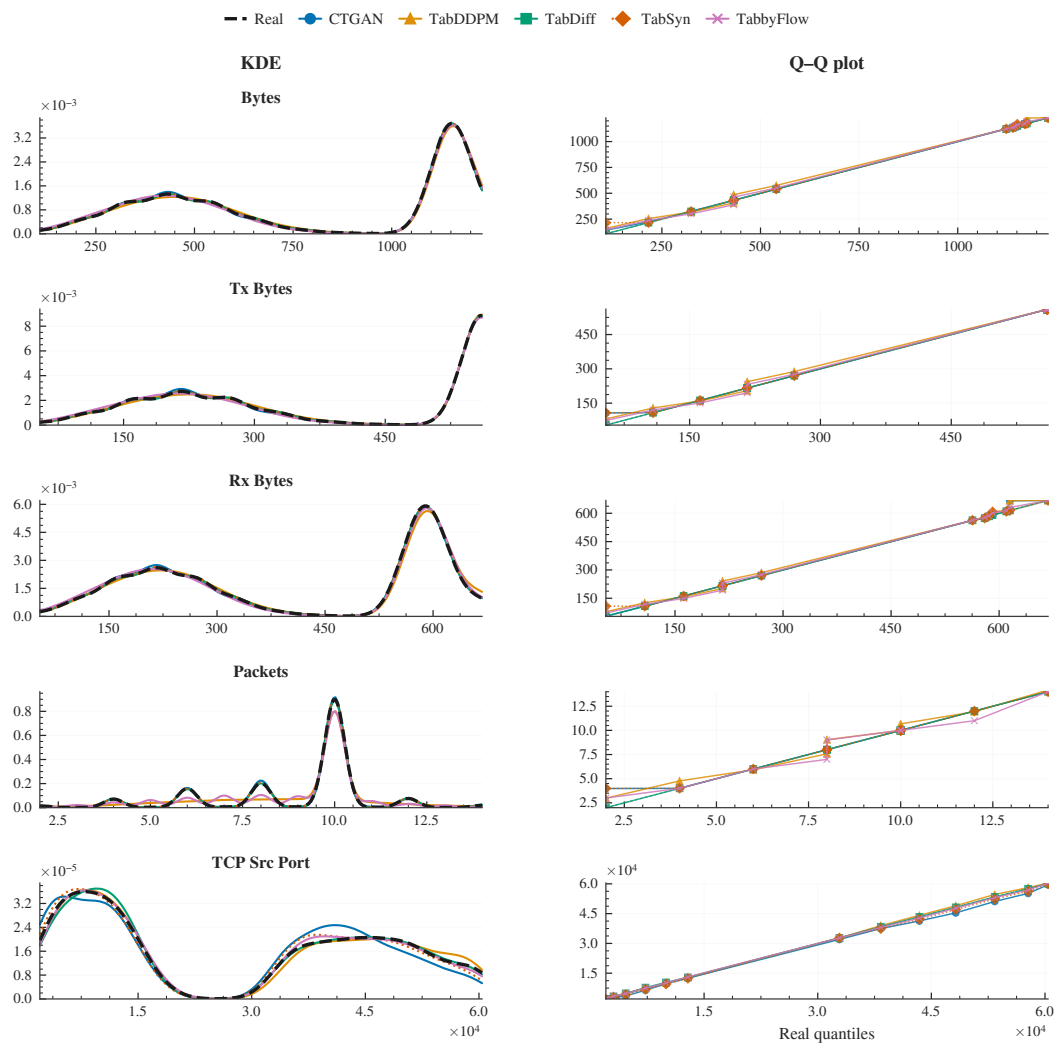

Figure 4: Feature-level fidelity comparisons for APA-DDoS. The left panels show kernel density estimates, and the right panels show quantile–quantile comparisons between real and generated samples.

For SDN, the generators obtain median normalized Wasserstein-1 distances between 0.012 and 0.021 and pairwise correlation differences between 0.043 and 0.046. Despite these moderate differences, their discriminator AUROCs range from 0.995 to 1.000. The generated samples are therefore almost perfectly distinguishable from the real samples in the joint feature space.

The same pattern appears in CICDDoS2019, CRCDDoS2022, and BCCC. A generator may reproduce the marginal distribution of one feature while shifting another, altering its tails, or changing its relationship with other features. No generator provides the closest match across all datasets and fidelity measures.

We also visualize the joint distributions using principal component analysis. For each dataset, the real data and each generated dataset are first sampled to at most 30,000 observations and then to at most 3,000 observations for PCA, using seed 42. The samples from the real data and all five generators are pooled and scaled jointly using min–max scaling. A single PCA model is fitted to the pooled samples, and the same two principal components are used to project all six sources. PCA is used only for visualization; all reported fidelity and classifier results are computed in the original feature space.

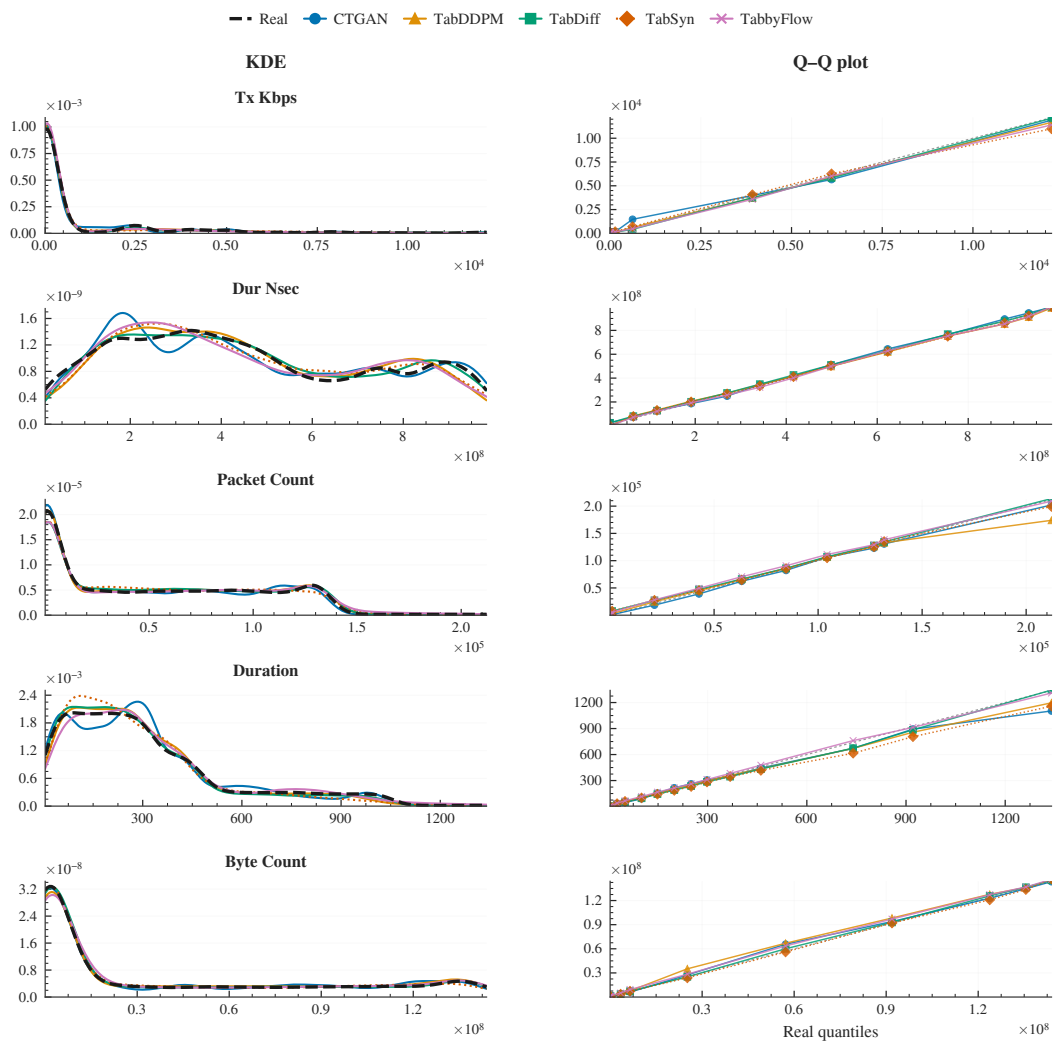

Figure 5: Feature-level fidelity comparisons for SDN. The generated distributions reproduce some marginal patterns but remain distinguishable from the real distribution.

The feature-level and PCA comparisons show why fidelity and classifier transfer must be evaluated separately. Fidelity measures statistical agreement between real and generated data, whereas TRTS measures whether a classifier trained on real data preserves its decision boundary on generated data. Similar marginal distributions therefore do not guarantee similar classifier behaviour.

A.6 Auxiliary Train-on-Synthetic, Test-on-Real Evaluation

TSTR examines whether generated traffic can be used to train a classifier that works on real traffic. This differs from the main TRTS setting, which tests a real-trained detector on generated traffic. For each dataset–generator pair, we make a stratified 80/20 split of the real data using seed 42. We train a Random Forest with 100 trees, maximum depth 10, and balanced class weights on synthetic data, then evaluate it on the 20% real test split. A Random Forest with the same configuration, trained on the corresponding real training split and evaluated on that test split, provides RealBA. This split is separate from the real-data partitions used in the main detector comparison. The gap is

$$\text{Gap} = \text{RealBA} - \text{TSTR-BA}. \tag{16}$$

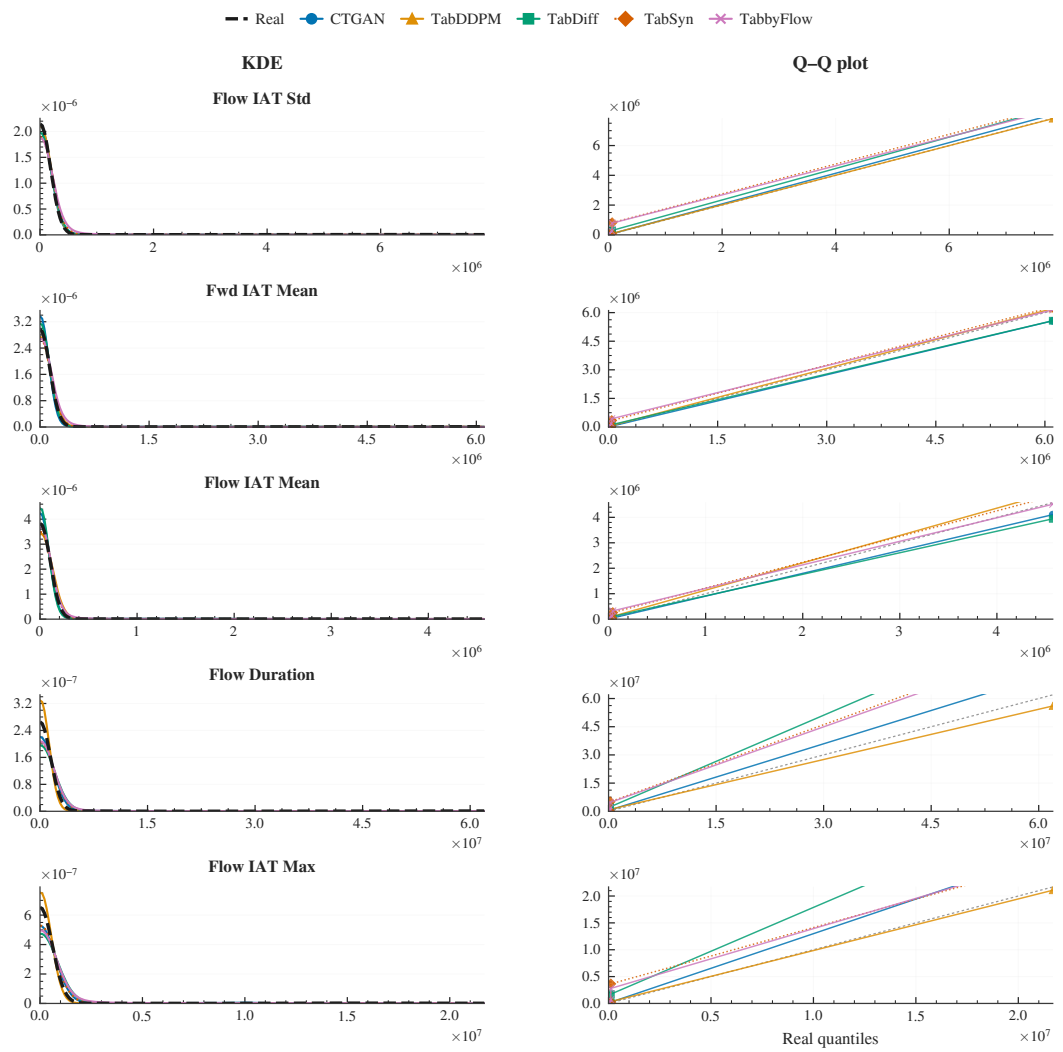

Figure 6: Feature-level fidelity comparisons for CICDDoS2019. Agreement varies across features and generators.

This experiment measures general synthetic-data utility and is separate from the detector comparison in the main paper.

RealBA is recomputed for each dataset–generator pair after retaining the numerical features shared by the real and generated inputs and subsampling the training data where necessary. It can therefore vary slightly across generators even when the real source data, test split, and Random Forest configuration are fixed.

RealBA is at least as high as TSTR-BA for every pair. This is expected because RealBA is an in-distribution result: the Random Forest learns from real training flows and is evaluated on real test flows collected and processed under the same conditions. In TSTR, the training set contains only the generator's approximation of that distribution. Missing feature combinations, altered dependencies, or generator-specific patterns can change the splits learned by the Random Forest. The resulting boundary may separate the synthetic classes but transfer less effectively to real traffic. The gap measures this loss in utility, although it does not identify the specific source of the mismatch.

APA-DDoS shows no loss, with all five generators matching RealBA at 1.000. SDN is also close to the real-trained reference: TSTR-BA ranges from 0.936 to 0.986 against 0.995, with gaps below

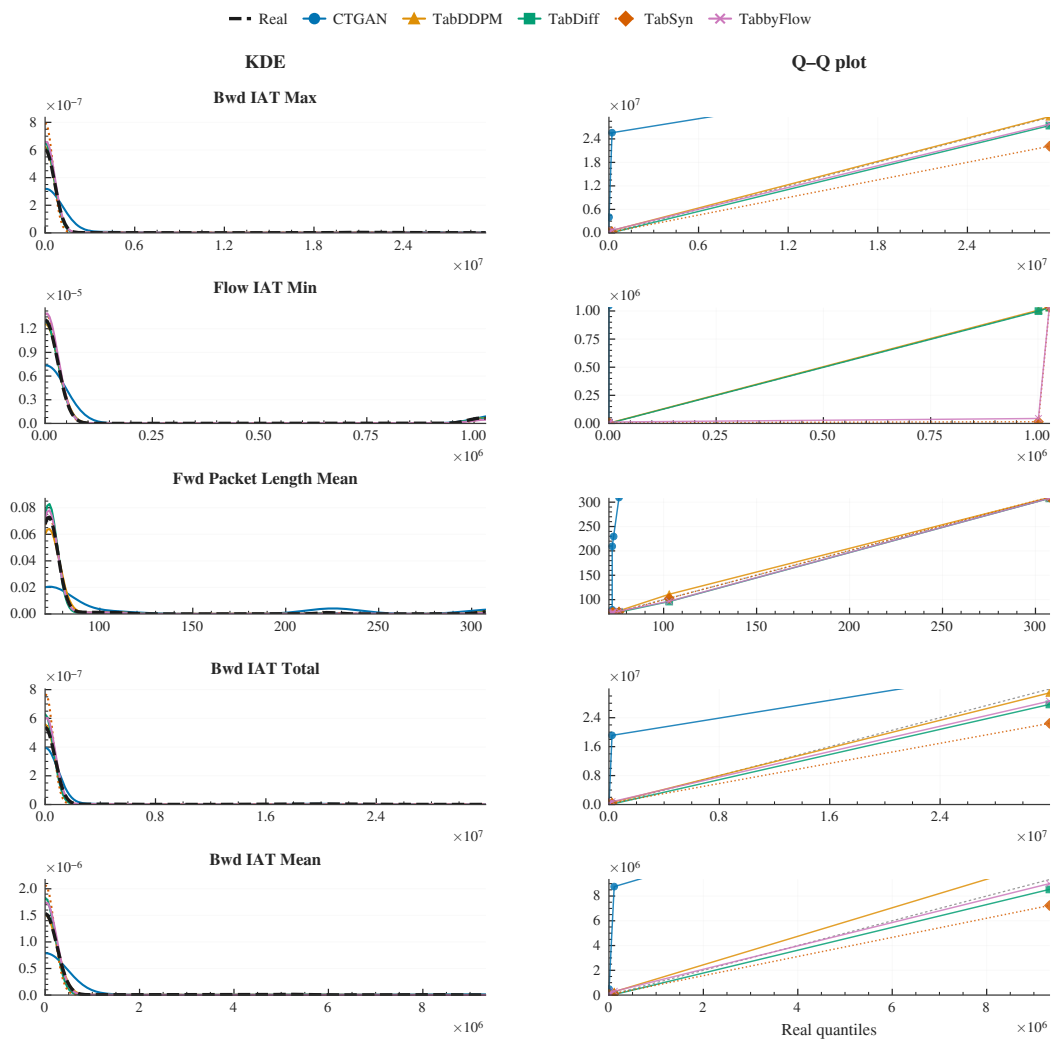

Figure 7: Feature-level fidelity comparisons for CRCDDoS2022. Differences remain visible in distribution shape, range, and tail behavior.

0.06. BCCC has a larger but narrow range of gaps, from 0.068 to 0.094. CICDDoS2019 is weaker, with TSTR-BA between 0.685 and 0.795 and a mean gap of 0.165, compared with 0.079 for BCCC.

The CRCDDoS2022 results depend strongly on the generator. TabbyFlow and TabDDPM nearly match RealBA, with gaps of 0.015 and 0.019, and CTGAN remains within 0.088. TabDiff and TabSyn fall to 0.540 and 0.548, producing gaps above 0.35. The dataset is therefore not uniformly difficult under TSTR; the loss is concentrated in these two generated training sets.

The fidelity scores do not account for this pattern. On CRCDDoS2022, TabDiff has the lowest normalized Wasserstein-1 distance (0.0095) and PCD (0.133) among the five generators, but it also has the largest TSTR gap. Discriminator AUROC is likewise unable to distinguish generators with substantially different downstream utility when its values are already close to 1.000. Marginal agreement, pairwise dependence, joint separability, and TSTR utility capture different aspects of the generated distribution and should not be treated as interchangeable.

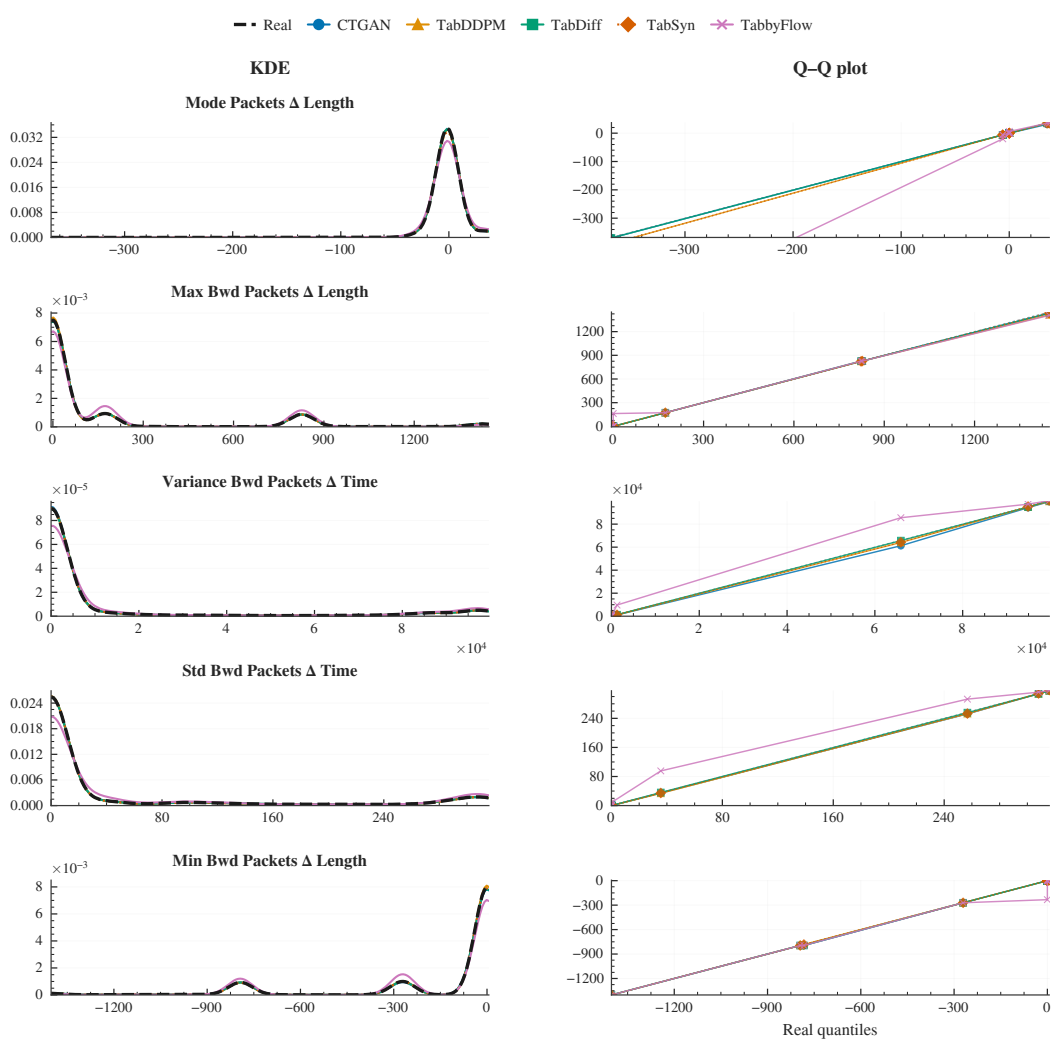

Figure 8: Feature-level fidelity comparisons for BCCC. The quality of the generated distributions depends on both the feature and the generator.

A.7 Cross-Generator Provenance Evaluation

The provenance experiment is conducted on APA-DDoS and BCCC. Let $\mathcal{G}$ denote the set of five generators. For each $g \in \mathcal{G}$, generator g is reserved for testing, while the other four generators provide synthetic attacks for training. This produces five evaluation rounds for each dataset.

Let $\mathcal{R}^A_{\text{tr}}$, $\mathcal{R}^A_{\text{val}}$, and $\mathcal{R}^A_{\text{te}}$ denote the real-attack training, validation, and test sets, respectively. Let $\mathcal{R}^B_{\text{te}}$ denote the real-benign test set, and let $\mathcal{S}^A_g$ denote the valid synthetic attacks produced by generator g. The superscripts A and B indicate attack and benign traffic, respectively. Synthetic benign flows are not used in this experiment.

Only synthetic attacks satisfying the dataset validity constraints are retained. Validity filtering is applied before sampling or evaluation. For each round, the synthetic training pool is formed from $\mathcal{G} \setminus \{g\}$. The samples are balanced across attack classes, and each observed generator contributes at most one quarter of the target synthetic training pool. Sampling is performed without replacement. The resulting synthetic attacks are class-matched to the real attacks used for training. Neither benign flows nor samples from the reserved generator are included in provenance training.

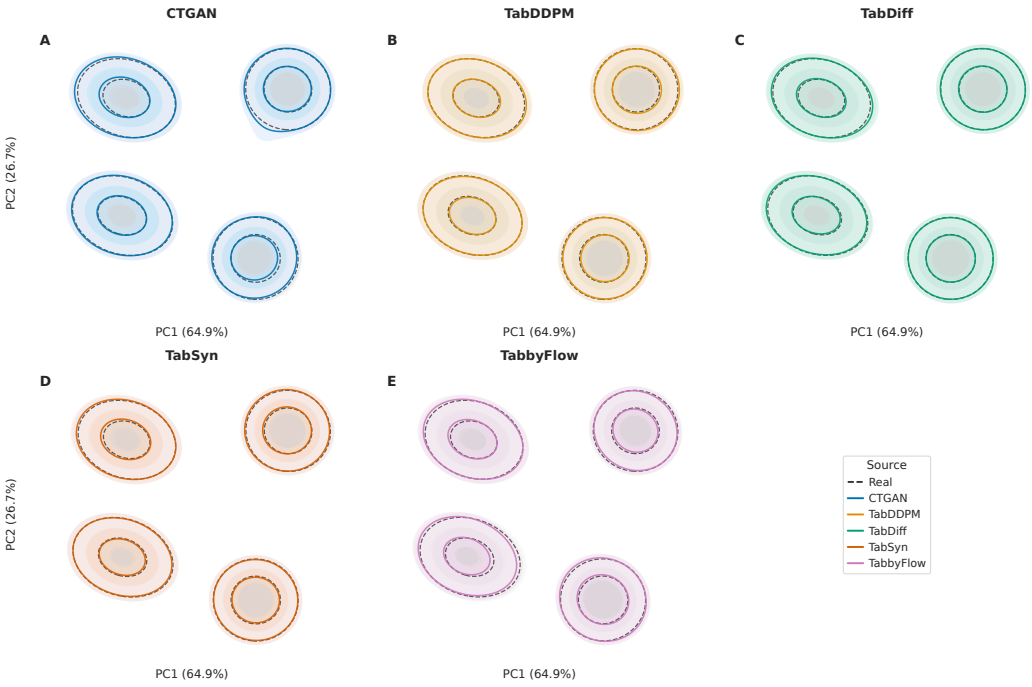

Figure 9: PCA projection of real and generated samples for APA-DDoS. A single projection is fitted jointly to samples from the real data and all five generators.

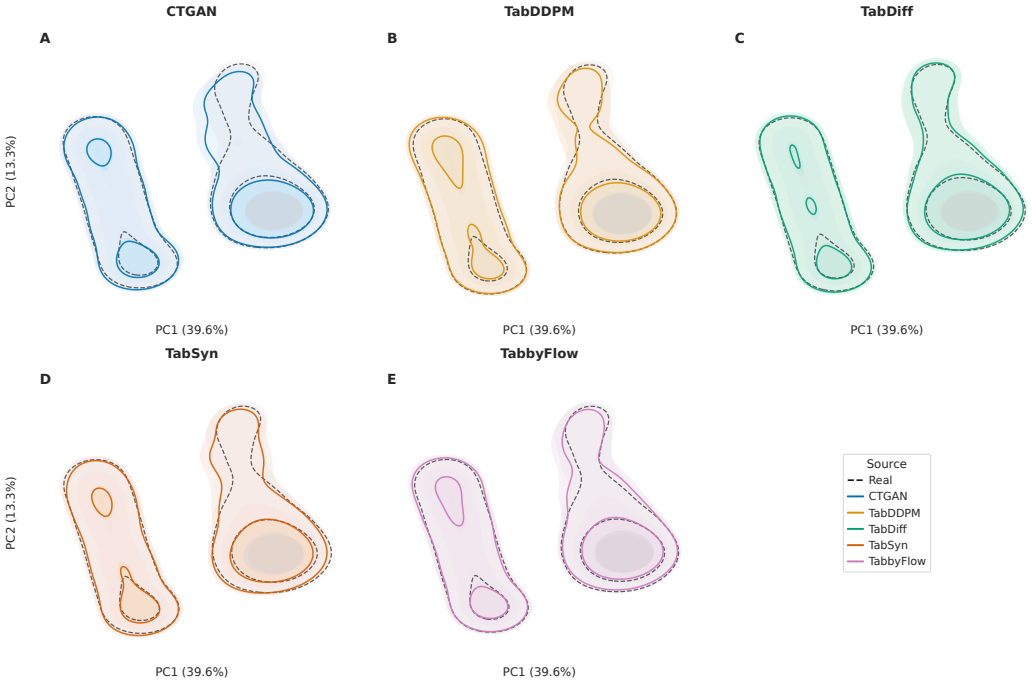

Figure 10: PCA projection of real and generated samples for SDN. A single projection is fitted jointly to samples from the real data and all five generators.

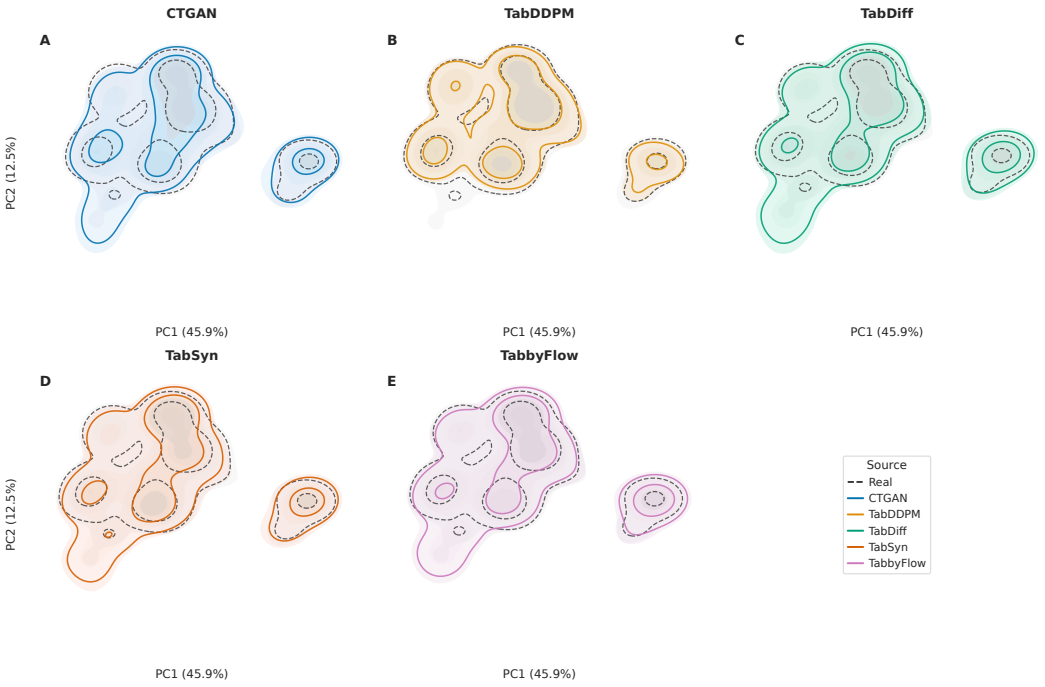

Figure 11: PCA projection of real and generated samples for CICDDoS2019. A single projection is fitted jointly to samples from the real data and all five generators.

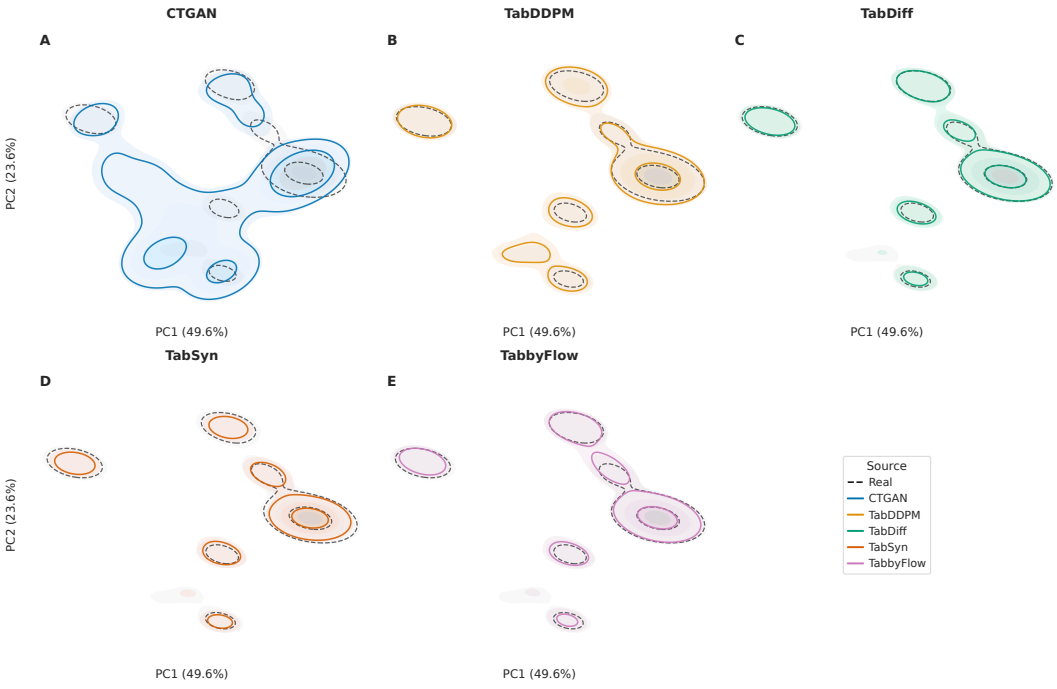

Figure 12: PCA projection of real and generated samples for CRCDDoS2022. A single projection is fitted jointly to samples from the real data and all five generators.

Table 8: TSTR balanced accuracy for all 25 dataset–generator pairs. TSTR-BA is obtained by training a Random Forest on generated traffic and testing it on held-out real traffic. RealBA is the corresponding real-trained TRTR result, and Gap is RealBA minus TSTR-BA. Mean rows summarize each dataset.

Dataset	Generator	TSTR-BA ↑	RealBA (TRTR) ↑	Gap ↓
APA-DDoS	CTGAN	1.000	1.000	0.000
	TabDDPM	1.000	1.000	0.000
	TabDiff	1.000	1.000	0.000
	TabSyn	1.000	1.000	0.000
	TabbyFlow	1.000	1.000	0.000
	Mean	**1.000**	**1.000**	**0.000**
SDN	CTGAN	0.952	0.995	0.043
	TabDDPM	0.972	0.995	0.023
	TabDiff	0.986	0.995	**0.009**
	TabSyn	0.936	0.995	0.059
	TabbyFlow	0.967	0.995	0.028
	Mean	**0.963**	**0.995**	**0.032**
CICDDoS2019	CTGAN	0.739	0.908	0.169
	TabDDPM	0.723	0.904	0.181
	TabDiff	0.772	0.908	0.136
	TabSyn	0.685	0.908	0.224
	TabbyFlow	0.795	0.908	**0.113**
	Mean	**0.743**	**0.907**	**0.165**
CRCDDoS2022	CTGAN	0.816	0.904	0.088
	TabDDPM	0.886	0.904	0.019
	TabDiff	0.540	0.904	0.364
	TabSyn	0.548	0.904	0.357
	TabbyFlow	0.889	0.904	**0.015**
	Mean	**0.736**	**0.904**	**0.169**
BCCC	CTGAN	0.716	0.806	0.090
	TabDDPM	0.738	0.806	**0.068**
	TabDiff	0.734	0.806	0.072
	TabSyn	0.712	0.806	0.094
	TabbyFlow	0.735	0.805	0.070
	Mean	**0.727**	**0.806**	**0.079**

The provenance classifier is trained to distinguish real attacks from synthetic attacks. Synthetic provenance is treated as the positive class. Let

$$s(\mathbf{x}) \in [0, 1] \tag{17}$$

denote the predicted probability that flow $\mathbf{x}$ is synthetic.

Threshold calibration. The rejection threshold is selected using only the real-attack validation set. Among all candidate thresholds, we choose the lowest threshold for which no more than 1% of the validation attacks are classified as synthetic:

$$\tau^* = \min \left\{ \tau : \frac{1}{|\mathcal{R}^A_{\text{val}}|} \sum_{\mathbf{x} \in \mathcal{R}^A_{\text{val}}} \mathbb{I}[s(\mathbf{x}) \geq \tau] \leq 0.01 \right\}. \tag{18}$$

Selecting the lowest admissible threshold maximizes synthetic-attack coverage subject to the calibration constraint. The threshold remains fixed during testing and is not adjusted for the reserved generator.

Provenance metrics. AUROC is computed using the real attacks in $\mathcal{R}^A_{\text{te}}$ as the negative class and the unseen synthetic attacks in $\mathcal{S}^A_g$ as the positive class. The catch rate for attacks from generator g is

$$\text{Catch}_g = \frac{1}{|\mathcal{S}^A_g|} \sum_{\mathbf{x} \in \mathcal{S}^A_g} \mathbb{I}[s(\mathbf{x}) \geq \tau^*]. \tag{19}$$

The observed rejection rate on the separate real-attack test set is

$$\mathrm{RR}_A = \frac{1}{|\mathcal{R}^A_{\mathrm{te}}|} \sum_{\mathbf{x} \in \mathcal{R}^A_{\mathrm{te}}} \mathbb{I}[s(\mathbf{x}) \geq \tau^*] \,. \tag{20}$$

Because this rate is measured on a test set rather than the calibration set, it need not equal the 1% calibration target exactly.

The same threshold is applied to held-out real benign flows:

$$\mathrm{RR}_B = \frac{1}{|\mathcal{R}^B_{\mathrm{te}}|} \sum_{\mathbf{x} \in \mathcal{R}^B_{\mathrm{te}}} \mathbb{I}[s(\mathbf{x}) \geq \tau^*] \,. \tag{21}$$

This quantity is reported separately because benign flows are not used to train or calibrate the provenance classifier.

Behavioural comparison. For generator g, let $\mathcal{S}^A_g$ denote the constraint-valid test flows whose ground-truth labels are attack classes. No synthetic benign flows are included in $\mathcal{S}^A_g$. The behavioural detectors are evaluated on this same unseen-attack set. Let $\mathcal{C}_A$ denote the set of attack classes and let $f(\mathbf{x})$ be the class predicted by a behavioural detector. Its attack-detection rate is

$$\mathrm{ADR}_g(f) = \frac{1}{|\mathcal{S}^A_g|} \sum_{\mathbf{x} \in \mathcal{S}^A_g} \mathbb{I}[f(\mathbf{x}) \in \mathcal{C}_A] \,. \tag{22}$$

Thus, an unseen synthetic attack is counted as detected whenever the model assigns it to any attack class. Predicting an incorrect attack subtype still counts as detection, whereas predicting the benign class counts as a miss.

The behavioural detectors retain the parameters and decision rules selected using real training and validation data. They are not trained on outputs from any generator. The provenance classifier and the behavioural detectors are therefore evaluated without retraining, threshold adjustment, or generator-specific calibration.

A.8 Additional TRTR and TRTS Detection Metrics

We report macro-F1, attack detection rate (ADR), and benign false-positive rate (B-FPR) alongside the balanced accuracy results in Table 3. Macro-F1 is the unweighted mean of the class-wise F1 scores. ADR is the proportion of attack samples assigned to any attack class, while B-FPR is the proportion of benign samples incorrectly assigned to an attack class.

The metrics are calculated from the same predictions used for balanced accuracy. The checkpoints, preprocessing states, decision rules, and test sets remain unchanged. No retraining, test-time adaptation, or generator-specific calibration is performed. Mean TRTS is the unweighted mean across the five generators.

The additional metrics reveal errors that are not visible from balanced accuracy alone. On CICDDoS2019, TinySEED reaches a mean TRTS ADR of 0.952, but its mean macro-F1 is 0.539 and its B-FPR is 0.247. It recognizes most attack samples as attacks, but it also confuses attack types and incorrectly classifies many benign samples as attacks. DistilBERT and TinySEED show a similar pattern on CRCDDoS2022, where high ADR is accompanied by high B-FPR.

D^2Former obtains the highest mean macro-F1 and ADR on BCCC, where the largest transfer losses occur. It does not obtain the lowest B-FPR on that dataset. No detector performs best across all metrics, datasets, and generators. Balanced accuracy is therefore used as the primary metric, while macro-F1, ADR, and B-FPR provide further information about the types of errors made by each detector.

Table 9: Macro-F1 under TRTR and TRTS evaluation. Bold indicates the highest result within each dataset and column.

Dataset	Detector	TRTR ↑	CTGAN ↑	TabDDPM ↑	TabDiff ↑	TabSyn ↑	TabbyFlow ↑	Mean TRTS ↑
APA-DDoS	D^2Former	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	MLP	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	TabTransformer	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	DistilBERT	0.556	0.556	0.546	0.534	0.535	0.539	0.542
	TinySEED	0.556	0.556	0.144	0.200	0.198	0.161	0.252
SDN	D^2Former	**0.999**	**0.923**	0.956	**0.975**	0.900	0.929	0.937
	MLP	0.989	0.843	**0.963**	0.970	**0.931**	**0.938**	0.929
	TabTransformer	0.990	**0.923**	**0.963**	0.963	0.918	0.937	**0.941**
	DistilBERT	0.992	0.576	0.546	0.627	0.529	0.628	0.581
	TinySEED	0.993	0.539	0.574	0.520	0.576	0.623	0.566
CICDDoS2019	D^2Former	**0.745**	**0.822**	0.777	0.811	**0.821**	**0.790**	**0.804**
	MLP	0.698	0.733	**0.811**	**0.814**	0.797	0.789	0.789
	TabTransformer	0.610	0.696	0.719	0.714	0.725	0.711	0.713
	DistilBERT	0.739	0.612	0.600	0.638	0.640	0.604	0.619
	TinySEED	0.678	0.489	0.495	0.568	0.603	0.542	0.539
CRCDDoS2022	D^2Former	0.999	0.960	0.996	0.996	0.994	0.993	0.988
	MLP	**1.000**	0.941	0.969	0.993	0.993	0.921	0.963
	TabTransformer	**1.000**	**0.991**	**1.000**	**1.000**	**1.000**	**1.000**	**0.998**
	DistilBERT	0.887	0.540	0.499	0.526	0.513	0.509	0.517
	TinySEED	0.871	0.496	0.520	0.621	0.617	0.562	0.563
BCCC	D^2Former	0.951	**0.714**	**0.844**	**0.651**	0.509	**0.730**	**0.690**
	MLP	0.955	0.372	0.429	0.369	0.336	0.363	0.374
	TabTransformer	**0.968**	0.690	0.833	0.647	0.486	0.707	0.672
	DistilBERT	0.831	0.589	0.548	0.548	**0.534**	0.552	0.554
	TinySEED	0.829	0.355	0.358	0.354	0.348	0.351	0.353

Table 10: Attack detection rate under TRTR and TRTS evaluation. Bold indicates the highest result within each dataset and column.

Dataset	Detector	TRTR ↑	CTGAN ↑	TabDDPM ↑	TabDiff ↑	TabSyn ↑	TabbyFlow ↑	Mean TRTS ↑
APA-DDoS	D^2Former	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	MLP	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	TabTransformer	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	DistilBERT	**1.000**	**1.000**	0.958	0.903	0.905	0.928	0.939
	TinySEED	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
SDN	D^2Former	**0.999**	**0.965**	0.956	**0.984**	0.952	**0.961**	**0.964**
	MLP	0.990	0.885	**0.979**	0.979	**0.958**	0.958	0.952
	TabTransformer	0.991	0.961	0.976	0.973	0.939	0.950	0.960
	DistilBERT	0.991	0.673	0.662	0.638	0.620	0.619	0.642
	TinySEED	0.993	0.742	0.674	0.685	0.666	0.615	0.676
CICDDoS2019	D^2Former	**1.000**	0.982	0.990	0.996	0.996	0.979	0.988
	MLP	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	TabTransformer	**1.000**	0.999	0.997	**1.000**	**1.000**	0.999	0.999
	DistilBERT	**1.000**	0.995	0.994	0.997	0.998	0.999	0.997
	TinySEED	**1.000**	0.970	0.858	0.997	0.996	0.937	0.952
CRCDDoS2022	D^2Former	0.997	0.999	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	MLP	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	TabTransformer	**1.000**	0.999	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	DistilBERT	0.958	0.456	0.953	0.996	0.997	0.994	0.879
	TinySEED	0.956	0.742	0.995	0.997	0.998	0.999	0.946
BCCC	D^2Former	0.987	**0.932**	**0.911**	**0.948**	**0.987**	**0.946**	**0.945**
	MLP	0.986	0.419	0.480	0.465	0.409	0.474	0.449
	TabTransformer	**0.989**	0.865	0.895	0.905	0.901	0.887	0.891
	DistilBERT	0.958	0.495	0.505	0.508	0.466	0.500	0.495
	TinySEED	0.945	0.264	0.277	0.282	0.286	0.293	0.280

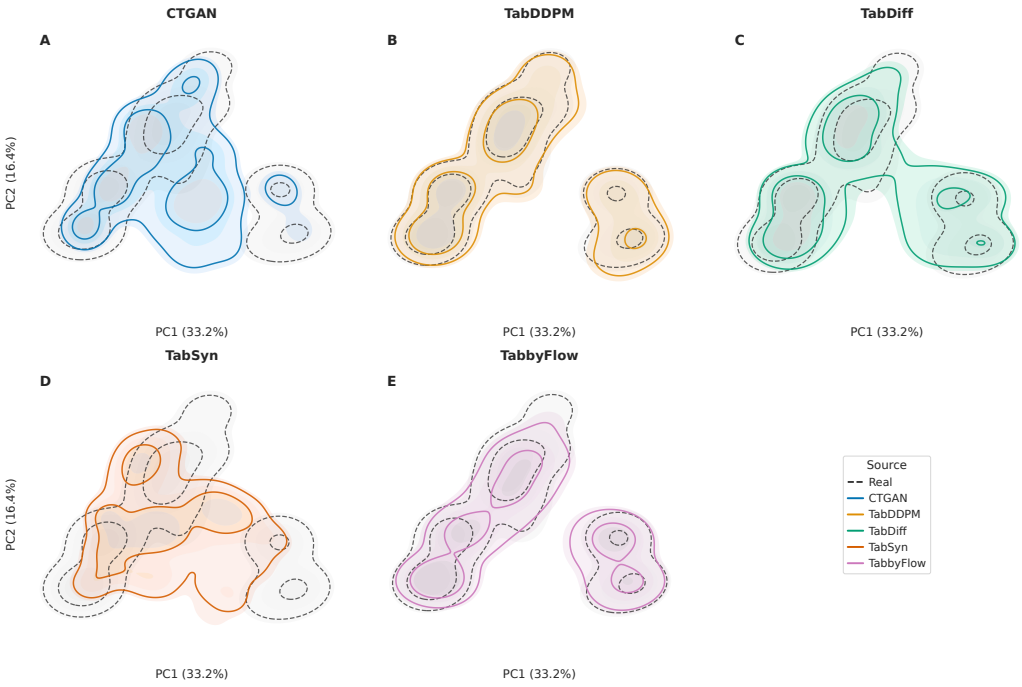

Figure 13: PCA projection of real and generated samples for BCCC. A single projection is fitted jointly to samples from the real data and all five generators.

Table 11: Benign false-positive rate under TRTR and TRTS evaluation. Bold indicates the lowest result within each dataset and column.

Dataset	Detector	TRTR ↓	CTGAN ↓	TabDDPM ↓	TabDiff ↓	TabSyn ↓	TabbyFlow ↓	Mean TRTS ↓
APA-DDoS	D^2Former	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
	MLP	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
	TabTransformer	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
	DistilBERT	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
	TinySEED	**0.000**	**0.000**	0.985	0.900	0.902	0.961	0.750
SDN	D^2Former	**0.001**	**0.044**	**0.023**	**0.016**	**0.053**	**0.037**	**0.034**
	MLP	0.013	0.177	0.044	0.034	0.082	0.071	0.082
	TabTransformer	0.014	0.097	0.043	0.041	0.090	0.067	0.068
	DistilBERT	0.006	0.484	0.528	0.369	0.528	0.352	0.452
	TinySEED	0.011	0.590	0.488	0.586	0.479	0.357	0.500
CICDDoS2019	D^2Former	**0.000**	0.117	0.077	0.070	0.081	0.036	0.076
	MLP	0.003	0.066	0.030	**0.011**	**0.016**	**0.033**	**0.030**
	TabTransformer	0.003	**0.045**	**0.017**	**0.011**	0.038	0.045	0.031
	DistilBERT	0.087	0.348	0.408	0.366	0.404	0.436	0.392
	TinySEED	0.107	0.304	0.176	0.349	0.215	0.191	0.247
CRCDDoS2022	D^2Former	0.005	0.147	0.018	0.018	0.025	0.028	0.047
	MLP	**0.000**	0.212	0.118	0.029	0.029	0.271	0.132
	TabTransformer	**0.000**	**0.035**	**0.000**	**0.000**	**0.000**	**0.000**	**0.007**
	DistilBERT	0.084	0.375	0.950	0.829	0.929	0.912	0.799
	TinySEED	0.089	0.776	0.839	0.625	0.621	0.763	0.725
BCCC	D^2Former	0.044	0.215	0.117	0.510	0.182	0.143	0.233
	MLP	0.040	0.171	0.213	0.159	0.218	0.187	0.190
	TabTransformer	0.029	0.187	0.115	0.462	0.184	0.140	0.218
	DistilBERT	0.025	**0.043**	**0.068**	**0.057**	**0.055**	**0.056**	**0.056**
	TinySEED	**0.018**	0.120	0.105	0.132	0.125	0.132	0.123