Journal of Advanced Research in Natural and Applied Sciences
dergipark.org.tr/tr/pub/jarnas
Open Access

ISSN: 2757-5195 12 (3) (2026) 398-419 doi.org/10.28979/jarnas.2001271

Causal-Convolutional Architectures for IoT Intrusion Detection: A Systematic Comparison Against Classical Machine Learning Baselines and a Bidirectionality Ablation

Ömer Faruk Sarıtaş[1]

Abstract— Intrusion detection for the Internet of Things (IoT) increasingly runs on IoT gateways and enterprise sensors that need architectures accurate enough to trust without being computationally heavy. Causal convolution offers gated, dilated, and multi-scale receptive-field mechanisms at low cost compared with recurrent or attention-based models. Whether such mechanisms need a bidirectional wrapper to approach classical machine-learning classifiers on tabular network-flow data remains unexamined. This study compares three causal-convolution mechanisms, dilated gated convolution, depthwise-separable gated convolution, and multi-scale causal inception, each trained bidirectionally and unidirectionally. A three-seed comparison on Telemetry, Operating system, and Network dataset (TON_IoT) finds no significant accuracy difference between the two forms for any mechanism although the bidirectional wrapper roughly doubles the parameter count. Two feature-grouped variants split the token sequence into two or four groups before a late fusion step. All architectures are benchmarked against classical baselines, including Random Forest and Extreme Gradient Boosting, on TON_IoT. The three bidirectional mechanisms and Extreme Gradient Boosting are additionally run once, under seed 42, on Canadian Institute for Cybersecurity IoT dataset (CIC-IoT-2023) as a limited external check. Dataset structure is characterized through mutual-information and principal-component analysis. The best unidirectional architecture, dilated gated convolution with only 402,000 parameters, reaches a three-seed mean macro F1-score of 95.47 percent and a Matthews Correlation Coefficient (MCC) of 0.9798 on TON_IoT. This is 0.97 points below Extreme Gradient Boosting's 96.44 percent. A pre-specified equivalence test confirms that this architecture's bidirectional and unidirectional forms are statistically equivalent within half a macro-F1 point. Gradient-boosted trees remain strongest, but the margin is narrow enough for causal convolution to work as a lightweight alternative. A feature-ordering check across four tokenization orders, each averaged over three seeds, shows this margin depends on the specific order used. The order adopted throughout this study was the strongest of the four for two of the three mechanisms, and statistically indistinguishable from the strongest for the third. Principal-component analysis shows TON_IoT's signal concentrates in a low-dimensional subspace, a pattern mutual information corroborates through the large share of near-uninformative raw features, together explaining k-Nearest Neighbors' competitive performance there. On TON_IoT, a single-direction causal-convolution pass is a compact alternative to tree ensembles and to its own bidirectional counterpart. Measured Central Processing Unit (CPU) cost is reported with the results.

Article Info
Received: 24 Jul 2026
Accepted: 22 Sep 2026
Published: 30 Sep 2026
Research Article

Keywords — *IoT intrusion detection, causal convolution, temporal convolutional network, ablation study, tabular classification*

[1]omer@kayseri.edu.tr (Corresponding Author)
[1]Department of Computer Engineering, Faculty of Engineering and Design, Kayseri University, Kayseri, Türkiye

1. Introduction

Internet of Things (IoT) intrusion detection increasingly runs on hardware that was never built for heavy machine learning, for example industrial gateways, home routers, and small edge servers sitting between a sensor network and the wider internet. A detector meant for that hardware has to be accurate, but it has to fit in a modest memory budget and respond quickly as well. Recurrent networks and attention-based sequence models tend to be accurate at the cost of size and latency, which pushes practitioners toward lighter alternatives whenever the accuracy trade-off is acceptable.

Causal convolution is one such alternative. It was built for autoregressive audio generation [1], and dilated causal-convolution stacks of this kind are broadly known as temporal convolutional networks (TCNs) [2]. Dilation lets a small filter cover a long receptive field without adding parameters. Gating, borrowed from the same audio-generation work, adds a multiplicative nonlinearity that behaves in a manner similar to an attention mechanism without its quadratic cost. Depthwise separable convolution, popularized in image classification [3], cuts parameter count further by splitting a standard convolution into a per-channel filter and a cheap channel-mixing step. Multi-branch inception-style convolution [4] captures several receptive-field sizes in one layer rather than one at a time. None of these ideas is new on its own. What remains less explored is how well they work, in bidirectional form, directly on tabular network-flow features, where the token order fed to the model is a modeling choice rather than a real temporal signal.

This study takes three causal-convolution mechanisms, each trained in a bidirectional form and in a unidirectional ablation that drops the backward branch, plus two feature-grouped variants built on top of the bidirectional depthwise-separable gated mechanism. Table 1 shows dilated gated convolution to be the strongest bidirectional mechanism, and Table 2 shows that its unidirectional ablation is statistically equivalent to it, within a pre-specified half-point margin, at half the parameter count. The full architecture comparison, including the five classical baselines, is run on Telemetry, Operating system, and Network dataset (TON_IoT). Three bidirectional mechanisms and Extreme Gradient Boosting are additionally run once on Canadian Institute for Cybersecurity IoT dataset (CIC-IoT-2023) as a limited external check. Dataset structure is examined through mutual information and principal-component analysis to explain why the gap to classical baselines differs from one dataset to the next. The result is a practical answer to a narrow question. How close can a lightweight convolutional architecture get to gradient-boosted trees on this kind of data, at what parameter cost, and is a bidirectional wrapper worth the extra parameters.

The gap this study addresses is specific. Prior causal-convolution and classical-baseline comparisons on IoT intrusion detection rarely test more than one causal mechanism against a narrow baseline set. None of the studies reviewed in Section 2 isolates what a bidirectional wrapper contributes once the input is tabular and feature-tokenized, rather than a genuine time series. This paper addresses both gaps and adds a further extension beyond them. First, it benchmarks three distinct causal-convolution mechanisms against the same five classical baselines under an identical protocol, rather than a single mechanism against a narrower baseline set, closing the first gap. Second, it isolates the bidirectional wrapper's contribution through a matched-seed ablation with formal significance testing, rather than reporting bidirectional results alone as prior bidirectional-TCN work on network traffic does, closing the second gap. Third, beyond these two gaps, it extends the comparison to a second, much larger benchmark, CIC-IoT-2023, and explains the resulting performance gap through an explicit dataset-structure analysis rather than leaving it unexplained.

The rest of the paper is organized as follows. Section 2 reviews related work on causal convolution

and on classical tabular classifiers. Section 3 describes the datasets, the architectures, the classical baselines, and the training protocol. Section 4 reports the results and discusses them against the dataset-structure analysis and against published literature. Section 5 states the limitations of the study, and Section 6 concludes the paper.

2. Related Work

The gated activation used throughout this study traces back to the WaveNet generative model for raw audio [1], where a hyperbolic tangent branch and a sigmoid branch are multiplied together at every layer. This design was reused later across many sequence-modeling architectures outside audio. Lea et al. [2] introduced the temporal convolutional network as a dilated, causal architecture for action segmentation in video. This study, and much of the later literature, uses that name for the broader family of dilated causal-convolution designs examined here.

Depthwise separable convolution reduces parameter count by factorizing a standard convolution into an independent per-channel filter followed by a pointwise mix. Chollet [3] used this idea to build the Xception image classifier at a fraction of the cost of comparable networks. The Inception family, introduced by Szegedy et al. [4] as GoogLeNet, takes the opposite approach to efficiency. It runs several kernel sizes in parallel within one layer, so that short- and long-range patterns are captured simultaneously rather than through depth alone. Both ideas are adapted here to a causal, one-directional-then-reversed setting suited to tabular, feature-tokenized input rather than images.

On the classical side, Breiman's Random Forest [5] and Chen and Guestrin's Extreme Gradient Boosting [6] remain difficult to beat on structured tabular data. Grinsztajn, Oyallon, and Varoquaux [7] traced this pattern to inductive biases that favor axis-aligned decision boundaries over the smooth functions deep networks naturally learn. Friedman's gradient boosting machine [8] underlies Extreme Gradient Boosting and Histogram-based Gradient Boosting alike. The older nearest-neighbor rule of Cover and Hart [9] still performs competitively whenever a dataset's informative signal sits in a compact, low-dimensional subspace, a property checked directly in Section 4 through principal-component analysis.

Recent work applies causal-convolution-adjacent architectures directly to IoT intrusion detection. Nazre et al. [10] train a temporal convolutional network on the Edge-IIoTset benchmark and report an accuracy of 96.72 percent across fifteen classes. They do not report a macro F1-score or a parameter count that would allow a direct comparison here. Gombar, Topalović, and Pejić Bach [11] compare two lightweight architectures, a convolutional model and a gated recurrent model, on UNSW-NB15 [12] and CIC-IDS2017 [13] using a cost-aware focal loss. They report a Matthews Correlation Coefficient (MCC) of up to 0.93 on CIC-IDS2017. Their formulation collapses both benchmarks to binary attack detection, rather than the ten-class task posed here by TON_IoT or the fifteen-class task Nazre et al. use for Edge-IIoTset, which limits how far the comparison can be pushed. Both studies confirm that lightweight convolutional and recurrent architectures remain an active line of work on exactly the kind of benchmarks this paper targets.

Bidirectional temporal convolutional networks have also been proposed directly for general network-traffic analysis, though on a different kind of input. Chen, Lv, Cai, Song, and Yin [14] combine dilated causal convolution with a bidirectional structure to detect abnormal network traffic. Akpaku et al. [15] compress a similar bidirectional design for encrypted malicious-traffic detection, and Wang, Huang, and Shi [16] add channel attention to a bidirectional temporal convolutional network for traffic-anomaly detection. All three operate on genuine temporal sequences, where packets or flow windows are ordered

in real time and a backward pass over that order can carry information the forward pass alone would miss. The tabular, feature-tokenized setting used here differs in an important way, since the order in which scalar features are tokenized is a modeling choice rather than a real temporal axis. Section 4 tests directly whether a backward pass still helps under that condition.

None of this literature reports a purely convolutional architecture benchmarked directly against classical tabular baselines on public IoT intrusion detection data with an explicit account of why the performance gap varies by dataset. None of it reports an ablation isolating what a bidirectional wrapper contributes on feature-tokenized tabular input either. Both gaps are addressed here.

3. Materials and Methods

This section describes the datasets, the feature tokenization, the causal-convolution mechanisms and their bidirectionality ablation, the feature-grouped variants, the classical baselines, the dataset-structure analysis, and the training protocol. Figure 1 summarizes the full pipeline end to end, from raw network-flow features through preprocessing and tokenization to the causal-convolution blocks and the final evaluation metrics.

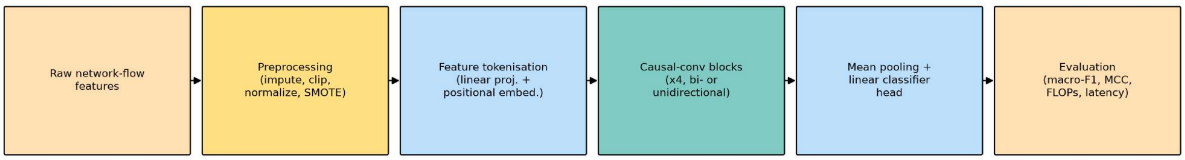


Figure 1. End-to-end experimental pipeline, from raw network-flow features to evaluation.

3.1. Datasets

Two public network-flow intrusion detection benchmarks are used. TON_IoT [17] contributes ten classes of IoT network traffic, drawn from the Train_Test_Network subset. The subset totals 211,043 raw rows, comprising 50,000 normal, 20,000 each for backdoor, distributed denial-of-service, denial-of-service, injection, password, ransomware, scanning, and cross-site scripting, and 1,043 man-in-the-middle. Source and destination host IP addresses were dropped before feature extraction. Source and destination ports were retained as numeric features, and the raw file carries no separate timestamp column. Duplicate rows were removed before splitting, 27,451 of 211,043, leaving 183,592 rows. The remaining 40 features split into 16 numeric fields and 24 categorical fields. The numeric fields include source and destination port, byte and packet counts, and Domain Name System and Hypertext Transfer Protocol fields. The categorical fields cover protocol, service, connection state, Secure Sockets Layer and Hypertext Transfer Protocol metadata, and anomaly-flag fields. These 40 features are stratified-split 70/15/15 into training, validation, and test partitions (random state fixed at 42). Because the split is performed on the raw feature values rather than on a hash, a small number of rows share identical post-processing feature values across splits despite carrying different raw connection identifiers. Out of 183,592 rows in total, this affects 16 rows between training and validation, 14 between training and test, and 3 between validation and test. This is a data-collection artifact, repeated flow records rather than leaked information, since it is measured after within-split deduplication and reflects the source dataset rather than the split procedure. It is disclosed here for transparency. CIC-IoT-2023 [18] represents a massive modern benchmark of IoT environments, comprising 33 attack classes and one benign class across 46 network features. The same 70/15/15 stratified split gives 5,491,971 training rows and

1,176,851 rows each for validation and testing, and no resampling is applied to this benchmark. The tabular network traffic is prepared for the causal-convolution layers by a fixed preprocessing sequence. Null records were replaced using the median of each respective feature, computed on the training split. Categorical fields were ordinal-encoded, with an explicit out-of-vocabulary code for categories seen only outside the training split. To mitigate the impact of extreme outliers, we restricted the feature distributions within the 0.1th and 99.9th percentiles prior to applying zero-mean unit-variance normalization. On TON_IoT only, class imbalance was then addressed in two steps applied exclusively within the training split. First, the majority class, normal traffic, was randomly undersampled to at most 10,000 rows. Second, the Synthetic Minority Oversampling Technique (SMOTE) algorithm [19] (five nearest neighbors) synthesized additional minority-class samples through interpolation up to the same 10,000-row target. All ten training classes end at exactly 10,000 rows as a result, and SMOTE is applied last in this sequence, operating on the already-clipped and normalized feature representation. Validation and test class priors are left untouched throughout, so the reported metrics reflect the original, imbalanced class distribution. Every statistic used in this sequence, the imputation median, the percentile-clipping bounds, the normalization mean and variance, the categorical encoding, and the SMOTE resampling, is fit on the training split alone. These statistics are only applied, not refit, to the validation and test splits, so no information from held-out data leaks into the transformation.

3.2. Feature Tokenization

Each numerical feature was linearly projected into a 128-dimensional embedding space, subsequently augmented with learnable position indicators to form the input sequence. This tokenization is shared by every architecture tested below, bidirectional and unidirectional alike, so any difference in results traces back to how the tokens are processed, rather than to how they are formed.

3.3. Bidirectional Causal-Convolution Mechanisms

Three mechanisms are tested, each wrapped in a bidirectional block that runs a forward pass and a reverse pass with independent weights and sums the two outputs. Every token sees both directions of the sequence this way, regardless of tokenization order. The first is dilated gated convolution, where a hyperbolic tangent branch and a sigmoid branch are multiplied together, following the WaveNet design [1]. It uses a kernel size of three with dilation doubling at each of four layers, so the receptive field grows quickly without adding parameters. The second factorizes the convolution into a depthwise filter followed by a pointwise mix, following Chollet's separable-convolution design [3]. It uses the same gating as the first mechanism, at roughly a third of the parameter count for the same receptive field. The third is adapted from the Inception family [4]. It runs three parallel causal branches with kernel sizes of three, seven, and fifteen and fuses them with a 1×1 convolution, capturing short-, medium-, and long-range feature interactions within a single layer. Every architecture stacks four such bidirectional blocks, each applying layer normalization to its input before the forward and backward convolutions and adding their sum back to the unnormalized input as a residual connection. A final layer normalization precedes mean pooling over tokens and a linear classifier head. Figure 2 details the internal structure of each of the three mechanisms individually.

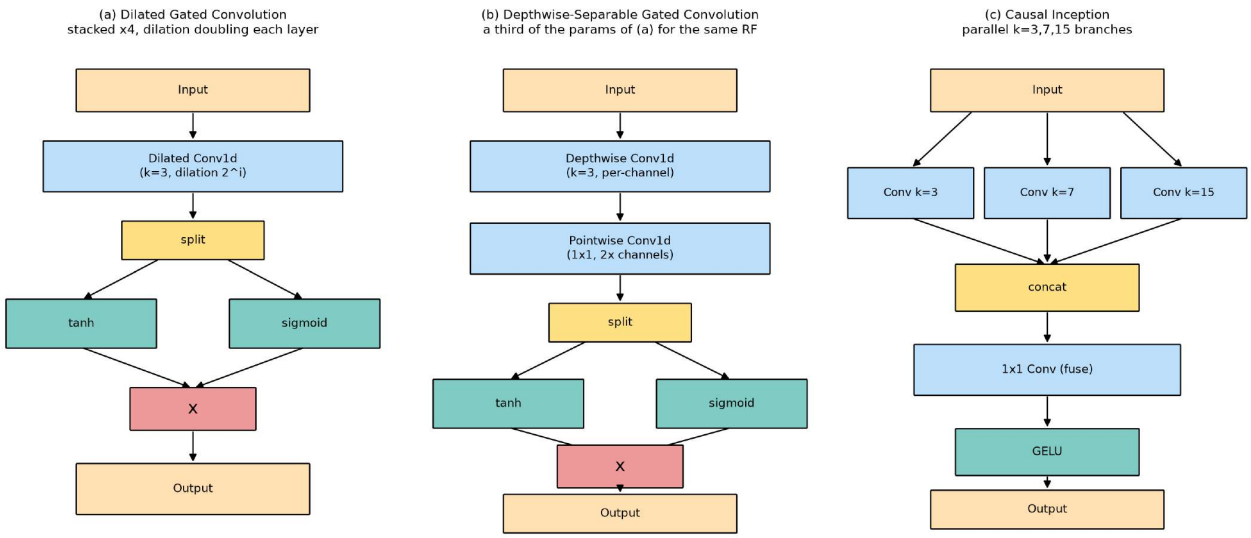


Figure 2. Internal structure of each causal-convolution mechanism, shown for its forward branch. The bidirectional block runs an independent copy of the same structure on the reversed sequence and sums the two outputs, as illustrated in Figure 3.

3.4. Bidirectionality Ablation

Each of the three mechanisms above is also trained in a unidirectional form. The backward branch and its independent weights are dropped from the bidirectional block, leaving only the forward branch, the residual connection, and every other setting unchanged. This roughly halves the convolutional parameter count of the affected blocks and isolates what the backward pass contributes beyond the forward pass alone. Both forms share the same tokenization, depth, hidden dimension, and training protocol, so any difference in results traces back to the presence or absence of the backward branch.

3.5. Feature-Grouped Variants

Two further variants split the token sequence into contiguous, independent groups before applying the bidirectional depthwise-separable gated mechanism from Section 3.3 separately within each group. Table 1 ranks dilated gated convolution marginally ahead of this mechanism, so extending the same grouping strategy to dilated gated convolution instead is left for future work. A late fusion step then concatenates the group-level pooled representations before the classifier head. One variant uses two groups and the other four. Both are tested only on TON_IoT. Figure 3 illustrates the shared block layout used by every architecture, with the feature-grouped variants applying the same stack independently within each group before the late fusion step.

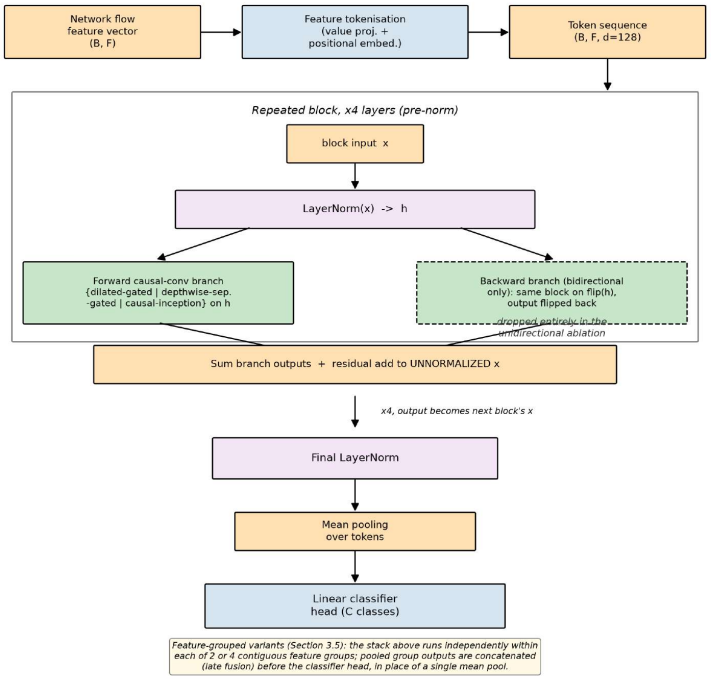


Figure 3. Block diagram of the tested causal-convolution family.

3.6. Classical Baselines

Five classical machine-learning baselines are trained on the identical preprocessed splits. Random Forest [5] uses two hundred unpruned trees with balanced-subsample class weighting. Extreme Gradient Boosting [6] runs 200 rounds at a maximum depth of eight and a learning rate of 0.1, with a Graphics Processing Unit (GPU)-accelerated histogram method. Histogram-based Gradient Boosting, built on Friedman's gradient boosting formulation [8], uses two hundred iterations at the same depth with balanced class weighting. A multinomial Logistic Regression completes the set alongside a k-Nearest Neighbors classifier with $k = 5$, following Cover and Hart's original rule [9]. The k-Nearest Neighbors classifier is fitted on a random subsample of up to 50,000 training rows rather than the full training split, since inference cost scales with the size of the fitted reference set. The reported macro F1-score for this classifier therefore likely underestimates what a full-reference-set fit could achieve, though a larger reference set is not guaranteed to raise it. Unlisted hyperparameters for all five baselines are left at their scikit-learn or Extreme Gradient Boosting library defaults.

3.7. Dataset Structure Analysis

Each dataset's feature set is characterized through mutual information between every feature and the class label, which identifies the share of near-uninformative features. A feature is counted as near-uninformative when its mutual information score falls below 0.01. The feature set is characterized through principal-component analysis as well, which reports the cumulative variance explained by the top three components as a proxy for intrinsic dimensionality. Both help explain why the gap between causal convolution and classical baselines differs across datasets. Both statistics are computed on the training split only, consistent with the leakage-safe preprocessing protocol of Section 3.1. Both are estimated on the same random subsample of 20,000 training rows for computational tractability.

3.8. Training Protocol

Every causal-convolution configuration, bidirectional, unidirectional, and feature-grouped alike, uses four blocks, a hidden dimension of 128, and a dropout rate of 0.1 within every block. These settings follow common practice in the causal-convolution literature [1, 2] and were fixed identically across every configuration rather than tuned per configuration. This choice keeps the reported comparison focused on architectural differences rather than on differences in hyperparameter-search effort. The positional embedding is initialized from a truncated normal distribution with a standard deviation of 0.02, and every other layer uses PyTorch's default initialization. Training runs for a fixed budget of 40 epochs with the AdamW optimizer [20], using a constant learning rate of 0.001 with no scheduler and a weight decay of 0.0001. Gradients are clipped to a maximum norm of 1.0, the batch size is fixed at 512, and the seed is fixed at 42 unless stated otherwise below. The training objective is an unweighted cross-entropy loss. All experiments use PyTorch 2.5.1 with Compute Unified Device Architecture (CUDA) 12.1, scikit-learn 1.9.0, XGBoost 3.2.0, and imbalanced-learn 0.14.2 for the SMOTE implementation. Neural-network training runs on a single NVIDIA L40S GPU. Central Processing Unit (CPU) inference latency (Table 6) is measured on an Intel Xeon Gold 6430 processor, pinned to a single thread or core. GPU kernel execution is not forced into a deterministic mode. Two settings were tried and both were abandoned on direct measurement rather than assumption. PyTorch's full deterministic-algorithms mode slowed every configuration by roughly a factor of ten while still warning that dense-layer matrix-multiplication operations remain non-deterministic regardless, so it provides no complete guarantee even at that cost. The CUDA Deep Neural Network (cuDNN) library's deterministic-algorithm mode was tried next on its own. For the largest-dilation configuration, a single training run exceeded seven hours without completing, against roughly ninety seconds under the library's default settings, a slowdown of about two orders of magnitude, and the run was terminated. Both settings were reverted in favor of the library's default, non-deterministic behavior used throughout. Exact reproduction of a single seed's result is therefore not guaranteed. Every comparison in this paper is reported instead as a multi-seed mean with a standard deviation or confidence interval. The individual per-seed values behind each aggregate statistic are reported directly as well (Table 3), so the paper's conclusions do not depend on any single run reproducing exactly.

The validation split is retained for protocol consistency but is not used for early stopping or hyperparameter selection. Every reported neural-network result is the epoch-40 checkpoint from the fixed 40-epoch budget. The TON_IoT multi-seed results in Tables 1–4, including the three-seed classical baselines, were obtained from a single training pipeline on one archived copy of the preprocessed data. The two feature-grouped variants remain single-run, seed-42 results under the same architecture and protocol. The CIC-IoT-2023 column of Table 5 is a separately executed single-seed check, documented as such. Random Forest, Histogram-based Gradient Boosting, and Extreme Gradient Boosting are trained with the hyperparameters given in Section 3.6. Logistic Regression uses the Limited-memory BFGS (L-BFGS) solver with a maximum of 500 iterations, and the SMOTE step in Section 3.1 uses five nearest neighbors. All causal-convolution configurations, the three plain bidirectional mechanisms of Section 3.3, their three unidirectional ablations of Section 3.4, and the two feature-grouped variants of Section 3.5, are evaluated on TON_IoT. The three plain bidirectional mechanisms, together with Extreme Gradient Boosting, are each additionally run once, under seed 42, on the massive CIC-IoT-2023 dataset. This serves as a limited external check on a much larger benchmark rather than a full replication of the TON_IoT protocol. The three plain bidirectional mechanisms and their three unidirectional ablations are each trained across multiple seeds on TON_IoT to check that the architecture ranking is not an artifact of a single training run.

4. Results and Discussion

Table 1 and Figure 4 report the full comparison. In Table 1, the three plain bidirectional causal-convolution mechanisms, dilated gated convolution, depthwise-separable gated convolution, and causal inception, are each validated across three seeds (42, 1, 2).

Table 1. Full architecture comparison on TON_IoT (ten classes). Macro F1-score (%) / MCC / parameter count; three-seed mean±std (seeds 42, 1, 2) where available.

Method	**Macro F1 ↑**	**MCC ↑**	**Params ↓**
Logistic Regression	66.9	0.688	N/A
k-NN ($k = 5$) [9]	92.8	0.966	N/A
Depthwise-Sep.+Gated (bidirectional)	95.14±0.15	0.9777±0.0018	**276K**
Causal Inception (bidirectional)	95.14±0.17	0.9782±0.0006	1.21M
Grouped Depthwise, 2 groups	95.38	0.9785	547K
Grouped Depthwise, 4 groups	95.56	0.9795	1.09M
Dilated+Gated (bidirectional)	95.63±0.10	0.9803±0.0004	796K
Random Forest [5]	96.14±0.05	0.9853±0.0002	N/A
Histogram-based Gradient Boosting [8]	96.37±0.07	0.9867±0.0006	N/A
Extreme Gradient Boosting [6]	**96.44±0.00**	**0.9870±0.0000**	N/A

The best value in each column is shown in bold.

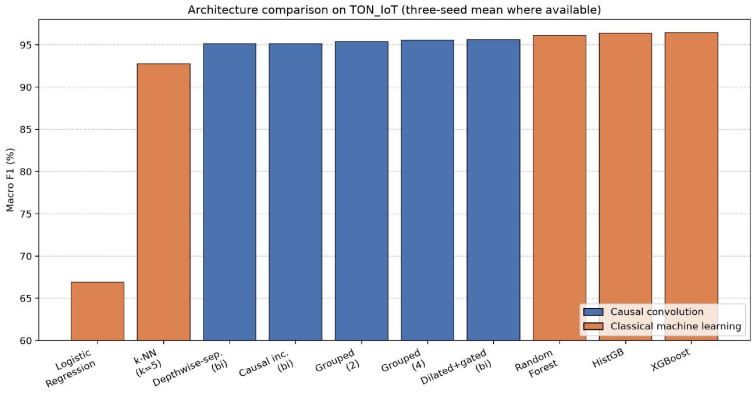


Figure 4. Macro F1-score (%) on TON_IoT across all tested methods, matching Table 1 (three-seed mean where available, seed 42 otherwise). Blue bars denote causal-convolution architectures, and orange bars denote classical machine-learning baselines.

All three are trained and evaluated together in a single regeneration run (Materials and Methods), so every mean and standard deviation reported below traces back to one consistent, archived set of runs. Random Forest, Histogram-based Gradient Boosting, and Extreme Gradient Boosting are each trained and evaluated across the same three seeds directly, in the same run. Table 5's TON_IoT column reports the same three-seed means for Extreme Gradient Boosting. The two feature-grouped variants and k-Nearest Neighbors remain single-run, seed 42 only, and Logistic Regression is deterministic given the fixed split. The dilated gated variant is the strongest of the five causal-convolution architectures, reaching a mean macro F1-score of 95.63 percent with 796K parameters. This is 0.81 points below Extreme Gradient Boosting's 96.44 percent (95 percent confidence interval 0.57 to 1.06 points, Welch t-test $p = 0.005$). The depthwise-separable variant trails it by 0.49 points (95 percent confidence interval 0.17 to 0.80 points, $p = 0.015$) while using a third of the parameters, 276K against 796K.

All three plain mechanisms show comparably low seed-to-seed variance in their bidirectional form, a standard deviation of 0.10 to 0.17 points. Causal inception's larger variance appears specifically in its unidirectional form (Table 2), not here. The two feature-grouped variants add parameters, 547K and 1.09M respectively, for a gain of 0.24 and 0.42 points over the three-seed depthwise-separable mean. This trade-off does not look worthwhile on this dataset, though the comparison remains single-seed for the grouped variants themselves. It was not subjected to a formal significance test either, so it should be read as a descriptive observation rather than a statistically confirmed conclusion. Figure 5 places these results side by side. It shows macro F1-score and MCC for the three plain bidirectional mechanisms against the three strongest classical baselines, and parameter count and Floating Point Operations (FLOPs) for the three mechanisms in both directions. Classical tree ensembles are not part of the parameter-count and FLOPs panels, since neither quantity is defined for them in a directly comparable way.

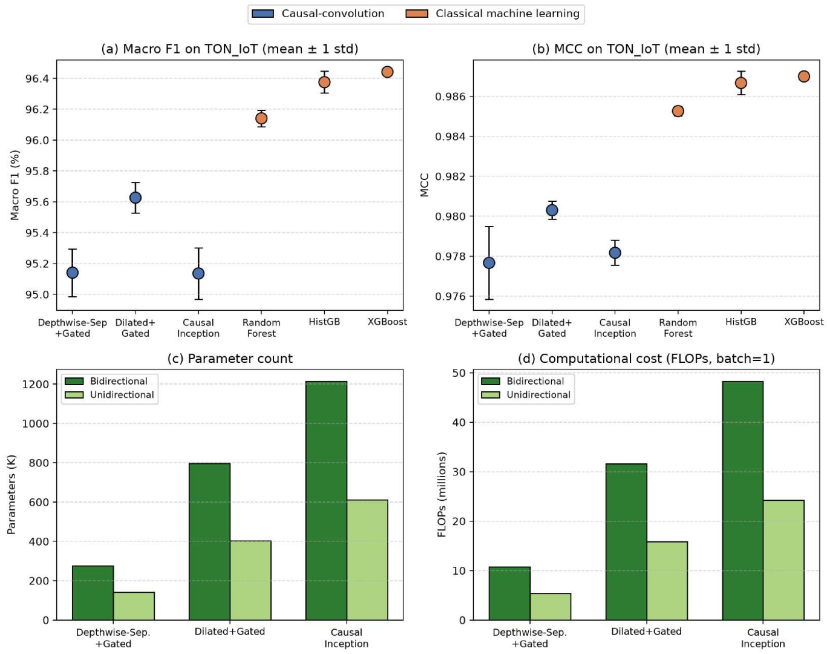


Figure 5. (a, b) Macro F1-score and MCC for the three bidirectional causal-convolution mechanisms against Random Forest, Histogram-based Gradient Boosting, and Extreme Gradient Boosting on TON_IoT. (c, d) Parameter count and FLOPs for the three causal-convolution mechanisms in both directions; classical baselines are not shown in these two panels, since neither quantity is defined for a tree ensemble in a way directly comparable to a neural network's.

Table 2. Bidirectionality ablation on TON_IoT. Macro F1-score (%) / MCC / parameter count; three-seed mean±std (sample standard deviation, seeds 42, 1, 2, trained together for both forms in one run); p from a Welch (unequal-variance) two-sample t-test between the bidirectional and unidirectional form of each mechanism.

Method	**Macro F1 ↑**	**MCC ↑**	**Params ↓**	p
Dilated+Gated (bidirectional)	**95.63±0.10**	**0.9803±0.0004**	796K	0.089
Dilated+Gated (unidirectional)	95.47±0.05	0.9798±0.0006	**402K**	
Depthwise-Sep.+Gated (bidirectional)	**95.14±0.15**	**0.9777±0.0018**	276K	0.181
Depthwise-Sep.+Gated (unidirectional)	94.88±0.23	0.9766±0.0005	**142K**	
Causal Inception (bidirectional)	**95.14±0.17**	**0.9782±0.0006**	1.21M	0.442
Causal Inception (unidirectional)	94.69±0.80	0.9765±0.0018	**611K**	

Best value per mechanism pair is shown in bold.

Table 2 reports the ablation, and Figure 6 shows the same comparison graphically with the confidence intervals discussed below. All three mechanisms are compared under the same matched-seed protocol. Three seeds (42, 1, 2) were used, with both directions trained in one consistent regeneration run, so no mechanism's comparison relies on a separately sourced or independently timed run. Sample (not population) standard deviation, computed with a divisor of $n - 1$, is used everywhere in this table and throughout the paper. To control the Family-Wise Error Rate (FWER) across the three ablation comparisons, a Bonferroni-adjusted significance threshold of $\alpha = 0.05/3 \approx 0.0167$ was applied. None of the three observed p-values cross this stricter threshold, or even the uncorrected $\alpha = 0.05$. The three values are 0.089 for dilated gated convolution, 0.181 for depthwise-separable gated convolution, and 0.442 for causal inception. The bidirectional form scores numerically higher than the unidirectional form on both macro F1-score and MCC for all three mechanisms, by 0.16, 0.26, and 0.44 points respectively. None of these differences reach significance at this sample size, however. The 95 percent confidence interval on each mean difference is reported as well, for a more transparent view of the uncertainty involved. These intervals are -0.05 to 0.36 points for dilated gated convolution, -0.20 to 0.72 points for depthwise-separable gated convolution, and -1.44 to 2.33 points for causal inception. Causal inception's markedly wider interval reflects its larger unidirectional-form standard deviation, 0.80 points, against 0.05 to 0.23 points for the other two mechanisms and forms.

A non-significant t-test does not by itself establish equivalence. A two one-sided tests (TOST) equivalence analysis [21] was therefore run for each mechanism with a margin fixed in advance at 0.5 macro-F1 points. This is a difference this study's deployment framing treats as practically negligible, computed directly from the same three-seed runs reported above rather than requiring additional seeds. For dilated gated convolution, TOST establishes statistical equivalence within this margin ($p = 0.007$). For depthwise-separable gated convolution and causal inception, equivalence is not established at this sample size ($p = 0.108$ and $p = 0.456$ respectively). For these two mechanisms, the data cannot yet distinguish a practically negligible difference from a larger one. The absence of a significant t-test result should not be read as evidence of equivalence on its own. Across all three mechanisms, the backward branch that defines the bidirectional wrapper consistently doubles the convolutional parameter count on this feature-tokenized tabular input, for a small, numerically consistent, but not always statistically confirmed accuracy benefit. Dilated gated convolution in its unidirectional form is therefore the architecture carried forward as the recommended lightweight configuration. It is the strongest unidirectional mechanism outright (Table 3). It is also the one mechanism for which the bidirectional wrapper's benefit is both small and, within the pre-specified practical margin above, statistically confirmed to be negligible rather than merely undetected. Against Extreme Gradient Boosting, this recommended configuration trails by 0.97 macro-F1 points on the same three seeds (Welch two-sample t-test, $t = -34.0$, $p < 0.001$, 95 percent confidence interval 0.85 to 1.10 points on the gap). This gap is small in absolute terms but, unlike the bidirectional-versus-unidirectional comparisons above, statistically significant. Extreme Gradient Boosting shows essentially zero seed-to-seed variance (Table 1) and so contributes almost no uncertainty of its own to the comparison.

Table 3. Individual per-seed macro F1-score (%) values underlying the three-seed means of Tables 1 and 2. Both directions of all three mechanisms come from the same run (Materials and Methods): every mean and standard deviation in Tables 1 and 2 is computed directly from the values shown here.

Method	Direction	Seed 42	Seed 1	Seed 2
Dilated+Gated	Bidirectional	95.53	95.73	95.62
	Unidirectional	95.51	95.41	95.48
Depthwise-Sep.+Gated	Bidirectional	95.26	95.20	94.97
	Unidirectional	95.14	94.73	94.77
Causal Inception	Bidirectional	95.18	94.95	95.27
	Unidirectional	95.33	93.79	94.96

Table 3 reports the individual seed values behind the aggregated statistics above, for full transparency beyond the mean and standard deviation. Every row is taken from the same training pipeline, so the means in Tables 1 and 2 are the averages of the values shown here. Causal inception's unidirectional seed 1 value, 93.79 percent, is the clear outlier behind that form's larger standard deviation (0.80 points, Table 2). The other two seeds for the same mechanism and direction, 95.33 and 94.96 percent, are close to the other mechanisms' typical range. Exact bit-for-bit reproduction of these values from the same seed is not guaranteed, since GPU kernel execution was not forced into a fully deterministic mode for this study (Materials and Methods). The per-seed values shown here are the ones actually obtained, and are what every aggregate statistic in this study is computed from.

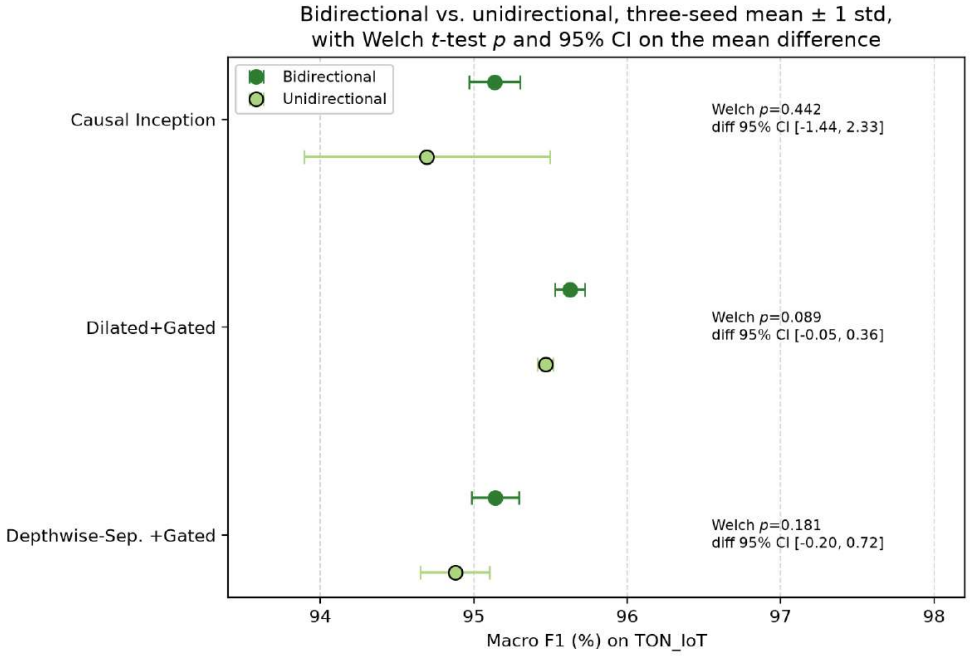


Figure 6. Bidirectional versus unidirectional macro F1-score for each mechanism on TON_IoT. Points and error bars show the three-seed mean ± 1 sample standard deviation; annotations give the Welch two-sample t-test p-value and the 95 percent confidence interval on the mean difference, from Table 2.

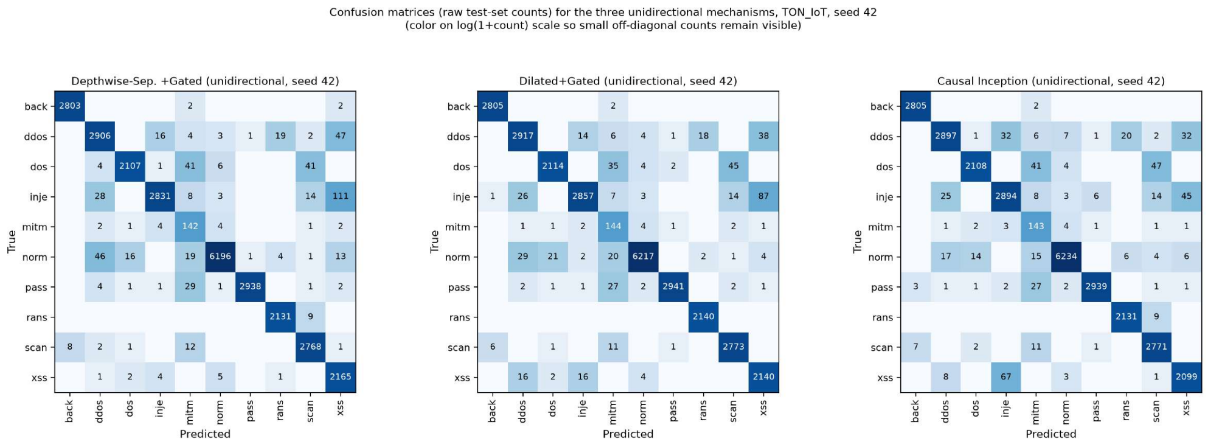


Figure 7. Confusion matrices for the three unidirectional mechanisms on the TON_IoT test set, seed 42 (Table 3), shown as raw counts rather than row-normalized, since the row-normalized view alone hides the precision loss discussed below.

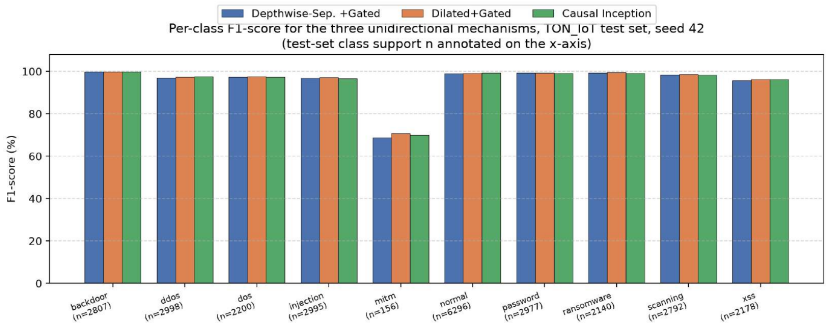


Figure 8. Per-class F1-score for the three unidirectional mechanisms on the TON_IoT test set, seed 42 (Table 3); test-set class support is annotated on the x-axis for every class.

Figures 7 and 8 break the aggregate macro F1-score down by class for the three unidirectional mechanisms, seed 42, the same run reported in Table 3. Every class reaches an F1-score above 0.95 except man-in-the-middle traffic, which sits at 0.69 to 0.71 across the three mechanisms, out of only 156 test-set examples of this class, 0.6 percent of the test set. Recall for man-in-the-middle traffic is high, 0.91 to 0.92, so most of the F1-score loss comes from precision, 0.55 to 0.57. Other traffic is being mislabeled as man-in-the-middle considerably more often than man-in-the-middle traffic itself is being missed, a pattern a row-normalized confusion matrix would hide. The dominant sources of these false positives are visible in the raw confusion counts of Figure 7. They are denial-of-service traffic (35 to 41 of each mechanism's false man-in-the-middle predictions), password traffic (27 to 29), and normal traffic (15 to 20). Man-in-the-middle's own errors, by contrast, spread thinly across several other classes, with at most 4 misclassified test examples landing in any single one of them. This class is small relative to denial-of-service and normal traffic in the test set. SMOTE's oversampling of it within the training split (Section 3.1) may not fully compensate for the classifier's difficulty separating it from higher-volume classes it resembles in feature space. This looks like a class-specific and data-volume issue more than a general architectural weakness, and is a natural target for future feature engineering or class-specific handling. Since macro F1-score weights every class equally, this one difficult class pulls the aggregate score down more than its small share of the test set would otherwise suggest.

Table 4. Feature-ordering sensitivity for the three unidirectional mechanisms on TON_IoT. Macro F1-score (%), three-seed mean±std (seeds 42, 1, 2) for every ordering; the Original column reuses the unidirectional values already reported in Tables 2 and 3, so no additional runs were needed for it.

Method	Original	Reversed	Random Perm. A	Random Perm. B
Depthwise-Sep.+Gated	**94.88±0.23**	94.07±1.08	94.84±0.06	94.82±0.64
Dilated+Gated	**95.47±0.05**	95.35±0.16	94.95±0.51	95.23±0.19
Causal Inception	94.69±0.80	**94.70±0.77**	94.50±0.40	93.98±0.92

Best ordering per row is shown in bold.

Table 4 reports macro F1-score for the three unidirectional mechanisms under four feature orderings, each now averaged over the same three seeds (42, 1, 2) used throughout this study rather than a single run. These are the original order used throughout this paper, a fully reversed order, and two independent random permutations, with every other setting unchanged. Feature order has a measurable effect on accuracy. The spread across the four ordering means reaches 0.81 points for depthwise-separable gated convolution and 0.52 points for dilated gated convolution. Both values sit well above each mechanism's own unidirectional seed-to-seed standard deviation reported in Table 2 (0.23 and 0.05 points respectively). Causal inception's spread, 0.71 points, is smaller than its own unidirectional standard deviation, 0.80 points, so for this mechanism specifically the ordering effect is not clearly distinguishable from ordinary seed-to-seed noise. The original order used throughout this paper is the strongest of the four for depthwise-separable gated convolution and for dilated gated convolution. For causal inception, the reversed order is marginally ahead, 94.70 against 94.69 percent, a gap far smaller than either ordering's own standard deviation and best read as a tie rather than a genuine ranking. The headline results should therefore be read as representative of a favorable tokenization for two of the three mechanisms, and as indistinguishable from the best available tokenization for the third. They should not be read as guaranteed optimal for any of them. The token order is indeed a modeling choice rather than a genuine temporal signal, as argued in Section 1, but this experiment shows that choice is not performance-neutral. A learned positional embedding paired with local convolutional kernels is sensitive to which features end up adjacent in the sequence. The causal-convolution family's margin against Extreme Gradient Boosting on TON_IoT is itself under one point for the recommended configuration (Table 2). An ordering effect of comparable size means part of that reported gap could plausibly close or widen under a different, equally arbitrary tokenization. Deterministic GPU kernels were not enforced for any experiment in this study (Materials and Methods). An earlier attempt to do so made training prohibitively slow for some of these architectures without providing a bit-exact guarantee in return. Even a fixed seed and a fixed ordering are therefore not expected to reproduce exactly across separate runs. This is itself part of why every comparison in this paper is reported as a multi-seed mean with a standard deviation or confidence interval rather than as a single number.

Table 5. Baseline comparison across datasets. All three causal-convolution rows are the bidirectional form. The TON_IoT column reports three-seed means (seeds 42, 1, 2); the CIC-IoT-2023 column reports single runs under seed 42, a narrower check rather than a like-for-like replication (Section 4). Every method reports a macro F1-score row and an MCC row.

Method	Metric	TON_IoT	CIC-IoT-2023
Depthwise-Sep.+Gated (bidirectional)	Macro F1 ↑	95.14	68.91
	MCC ↑	0.9777	0.9869
Dilated+Gated (bidirectional)	Macro F1 ↑	95.63	70.36
	MCC ↑	0.9803	0.9874
Causal Inception (bidirectional)	Macro F1 ↑	95.14	69.40
	MCC ↑	0.9782	0.9847
Extreme Gradient Boosting [6]	Macro F1 ↑	**96.44**	**75.51**
	MCC ↑	**0.9870**	**0.9922**

Best value per column is shown in bold.

Table 5 reports results on both datasets, but the two columns are not evaluated with the same breadth. Only the three plain bidirectional causal-convolution mechanisms and Extreme Gradient Boosting were run on CIC-IoT-2023, once each under seed 42. The CIC-IoT-2023 column is therefore a limited external check rather than a full replay of the five-baseline, three-seed suite used for TON_IoT in Table 1. Extreme Gradient Boosting remains the strongest baseline on TON_IoT. The CIC-IoT-2023 run indicates how the causal-convolution mechanisms behave on a massive 34-class problem with nearly 5.5 million training rows, the natural 70 percent split of the full dataset. No resampling was applied to this benchmark, unlike TON_IoT. Dilated gated convolution again leads the causal-convolution family there, reaching a macro F1-score of 70.36 percent against Extreme Gradient Boosting's 75.51 percent. This 5.15-point gap is considerably wider than the 0.81-point gap the same bidirectional mechanism shows on TON_IoT, a difference examined further through the dataset-structure analysis below. Unlike the TON_IoT figures, the CIC-IoT-2023 results reflect a single training run under seed 42 rather than a three-seed mean, and should be read with that caveat.

Table 6. Computational cost at batch size one, on TON_IoT's feature count. Parameter count (on-disk model size in megabytes for the classical baselines, which have no directly comparable parameter count), FLOPs (thop profiler for PyTorch), and single-sample CPU inference latency (mean±std over 200 repeats after 20 warm-up calls, pinned to a single thread/core of an Intel Xeon Gold 6430).

Method	Params ↓	FLOPs ↓	CPU Latency (ms) ↓
Depthwise-Sep.+Gated (bidirectional)	276K	10.7M	2.47±0.07
Depthwise-Sep.+Gated (unidirectional)	**142K**	**5.4M**	**1.32±0.04**
Dilated+Gated (bidirectional)	796K	31.6M	6.33±0.10
Dilated+Gated (unidirectional)	402K	15.8M	3.26±0.04
Causal Inception (bidirectional)	1.21M	48.3M	3.88±0.07
Causal Inception (unidirectional)	611K	24.2M	2.02±0.07
Random Forest [5]	112 MB	N/A	14.40±0.13
Histogram-based Gradient Boosting [8]	**3.4 MB**	N/A	6.43±0.08
Extreme Gradient Boosting [6]	6.6 MB	N/A	**2.08±0.03**

Best value per column within each block (causal-convolution, classical) is shown in bold.

Table 6 reports parameter count, FLOPs, and single-sample CPU inference latency for every causal-

convolution architecture, and on-disk model size and CPU inference latency for the three classical baselines with a directly comparable cost. Random Forest's 200 unpruned trees serialize to 112 megabytes, roughly an order of magnitude larger than Extreme Gradient Boosting (6.6 megabytes) and Histogram-based Gradient Boosting (3.4 megabytes). This is also between one and a half and two and a half orders of magnitude larger than any causal-convolution model's raw parameter footprint. That footprint ranges from 0.5 to 4.6 megabytes as a 32-bit buffer for 142K to 1.21M parameters. On latency, though, the ranking is not as one-sided. Extreme Gradient Boosting's 2.08-millisecond single-sample CPU latency is faster than every bidirectional causal-convolution mechanism and faster than unidirectional dilated gated convolution, but slower than the unidirectional depthwise-separable and causal-inception forms. A lightweight neural alternative's latency advantage over gradient-boosted trees therefore depends on which configuration is deployed, not on size alone. FLOPs were measured with the thop profiler for PyTorch at a batch size of one. Latency is a mean and standard deviation over 200 repeated single-sample forward passes, or inference calls for the classical models, following 20 warm-up calls. All measurements are pinned to a single thread or core of an Intel Xeon Gold 6430 processor, to reflect a resource-constrained, single-threaded deployment setting rather than a fully parallel server. FLOPs and latency both roughly halve from the bidirectional to the unidirectional form of every mechanism, consistent with dropping exactly one of the two convolutional branches. Parameter count, FLOPs, and latency do not all rank the three causal-convolution mechanisms identically. Causal inception carries the most parameters and FLOPs of the three in both directions, 1.21M parameters and 48.3M FLOPs bidirectional against dilated gated convolution's 796K parameters and 31.6M FLOPs. It runs faster in wall-clock terms nonetheless, 3.88 ms against 6.33 ms bidirectional. Its parallel, narrower branches map more efficiently onto this CPU than the single wide dilated convolution does. Parameter count alone would therefore have ranked these two mechanisms in the wrong order for actual deployment latency. Depthwise-separable gated convolution is the cheapest mechanism on every measure in both directions, and its unidirectional form is the cheapest configuration overall, at 142K parameters, 5.4M FLOPs, and 1.32 ms. Against this, unidirectional dilated gated convolution, the configuration recommended in Section 4 on accuracy grounds, costs 402K parameters, 15.8M FLOPs, and 3.26 ms, roughly 2.5 to 2.9 times more on every measure. This buys an accuracy gain of 0.59 macro-F1 points over unidirectional depthwise-separable gated convolution (Table 3). This trade-off is stated here explicitly. Where minimizing cost matters more than a gain of this size, unidirectional depthwise-separable gated convolution, not dilated gated convolution, is the more defensible lightweight recommendation. This remains a CPU proxy for both model families, rather than a measurement on an actual embedded IoT device or on the NVIDIA L40S GPU used for training (Materials and Methods).

Table 7. Dataset structure analysis for the IoT datasets.

Dataset	Near-zero MI features	Top-3 PCA variance
TON_IoT	19/40 (47.5%)	99.9%
CIC-IoT-2023	12/46 (26.1%)	40.9%

Table 7 summarizes this structure. Both the mutual-information and the principal-component statistics are computed on the already-standardized training features (Section 3.1), so the concentration reported below is not an artifact of unscaled feature magnitudes. TON_IoT's variance concentrates heavily in a three-dimensional subspace, 99.9 percent of variance in the top three principal components, with nearly half its raw features carrying close to no mutual information about the label. That concentration

is a plausible reason why k-Nearest Neighbors alone reaches a macro F1-score of 92.8 percent there, and why the causal-convolution architectures close most of the gap to gradient-boosted trees on that particular dataset. The dataset structural analysis for CIC-IoT-2023 reveals a much more distributed signal, with only 40.9 percent of the variance captured by the top three components and roughly a quarter of its features being near-uninformative. This broader, harder-to-compress signal distribution, combined with its massive scale, is a plausible reason why lightweight causal-convolution designs lack the representational capacity that this dataset needs. This is consistent with the wide gap to Extreme Gradient Boosting reported above rather than with the narrow one seen on TON_IoT.

Two recent studies report results on the same TON_IoT benchmark under different problem formulations, and the comparison belongs here rather than left for a reader to find separately. Benaddi et al. [22] apply knowledge distillation and Shapley Additive Explanations (SHAP)-guided pruning to a binary attack-versus-benign version of TON_IoT. They reach a macro F1-score of 0.9863 to 0.9867 with a student network of only 3,042 parameters. That is about 47 times smaller, closer to one and a half orders of magnitude, than the smallest causal-convolution architecture tested here. Binary detection is a substantially easier task than the ten-class formulation used throughout this paper, so the two results are not directly comparable. The finding does show that extreme parameter efficiency is reachable once the classification target is simplified. Bahgat, Farrag, and Rashed [23] stay within the same ten-class train_test_network subset used here. They report up to 99.59 percent accuracy and a macro F1-score of 0.996 from an ensemble of fifteen classical classifiers combined through hybrid feature selection and Bayesian-optimized weighted soft voting. That figure exceeds this study's best classical result, Extreme Gradient Boosting's 96.44 percent macro F1-score, by close to 3.2 percentage points. Their source paper does not document feature-selection budget, hyperparameter search cost, or the exact train and test split fully, so a fully controlled comparison is not possible. The gap suggests still that heavier feature engineering and ensemble tuning have room to close on this dataset, beyond what the classical baselines in this study attempt. Neither external result changes this paper's central finding, which concerns the gap between the causal-convolution family and Extreme Gradient Boosting. Both place that finding inside a wider ceiling on TON_IoT that remains open rather than closed.

5. Limitations

A few points limit how far these results should be pushed. The three plain bidirectional mechanisms and three of the five classical baselines carry three-seed means and standard deviations. Section 4 reports significance tests for the bidirectionality ablation and for the recommended unidirectional architecture against Extreme Gradient Boosting, but not for every pairwise comparison among the five architectures in Table 1. The two feature-grouped variants, together with Logistic Regression and k-Nearest Neighbors, remain single-seed without a significance test at all. The small percentage-point differences between the causal-convolution architectures should not be treated as final, therefore, until checked across more seeds and tested for statistical significance where a test is not yet reported. Every architecture is trained for a fixed epoch budget without validation-based early stopping, which keeps the comparison focused on architecture rather than on model-selection effort. It means as well, however, that the reported numbers may not reflect each architecture's best achievable result.

The feature-grouped variants were tried only on TON_IoT, so it is not known whether the same modest, cost-heavy gain pattern would hold on CIC-IoT-2023. The CIC-IoT-2023 evaluation itself is narrower than the TON_IoT evaluation on two further counts. Its causal-convolution and Extreme Gradient Boosting figures come from a single training run under seed 42, unlike the three-seed protocol behind Table 1 and Table 2. Its classical-baseline comparison is limited to Extreme Gradient Boosting as

well, since Random Forest, Histogram-based Gradient Boosting, Logistic Regression, and k-Nearest Neighbors were not evaluated on this larger benchmark. The bidirectionality ablation of Table 2 is likewise limited to TON_IoT and to the three plain mechanisms. The feature-grouped variants were not tested in unidirectional form, and whether a real temporal signal in a different tokenization scheme would restore a bidirectional advantage was not tested here. The feature-ordering check of Table 4 is limited to four orderings out of the factorially many possible arrangements, though each is now averaged over the same three seeds used elsewhere in this study. It shows that the specific order used throughout this study is the strongest of the four tested for two of the three mechanisms, and statistically indistinguishable from the strongest for the third. The reported margin against Extreme Gradient Boosting should therefore be read as achievable under a reasonably favorable tokenization, not as guaranteed under an arbitrary one. Exact bit-for-bit reproduction of any single seed's result is not guaranteed either, since deterministic GPU kernels were not enforced (Materials and Methods). Every comparison in this study is accordingly reported as a multi-seed statistic rather than relied upon as a single-run figure.

This study stops at classification performance on known attack categories as well. It does not examine unsupervised anomaly detection, concept drift, or previously unseen attack types. These are all practical failure modes once a tabular machine-learning-based intrusion detection system faces adversaries who adapt their traffic patterns over time. This is documented for network intrusion detection specifically by INSOMNIA's concept-drift-robust retraining framework [24] and by recent surveys and empirical studies of adversarial robustness under drift [25, 26]. A production deployment would need periodic retraining or drift detection alongside the accuracy results reported here. A different balance between convolutional receptive field and classifier capacity might apply as well, once anomaly detection, rather than classification among known classes, becomes the goal. No adversarial-robustness evaluation is included either, which would matter for an architecture pitched explicitly at resource-constrained deployment. This is left to future work, together with a temporally ordered evaluation, training on network captures from one period and testing on a later one. Such an evaluation would turn the drift concern raised here into a concrete, testable measurement rather than a general caveat. Table 6 reports FLOPs and CPU latency, but only as a proxy measurement on a general-purpose processor rather than on an actual embedded IoT gateway chip. The deployment claim is therefore supported by a measured CPU proxy, though not yet confirmed on target hardware.

The classical baselines in this study use default or lightly tuned settings as well. Published work on the same TON_IoT subset has used heavier feature selection and ensemble tuning instead, to reach higher accuracy. The reported gap between causal convolution and classical machine learning should be read, therefore, as a floor on the classical side, not a ceiling.

6. Conclusion

This study compared three causal-convolution mechanisms, each in a bidirectional and a unidirectional form, plus two feature-grouped variants, all evaluated on TON_IoT. These were compared against five classical machine-learning baselines on TON_IoT, and against Extreme Gradient Boosting on the additional CIC-IoT-2023 benchmark. None of the three mechanisms showed a statistically significant accuracy difference between its bidirectional and unidirectional form on TON_IoT at the sample size tested. This holds even though the bidirectional wrapper roughly doubles the parameter count and scores numerically higher on both metrics for every mechanism. This result contrasts with bidirectional temporal convolutional networks built for genuine time-ordered network traffic. The feature-tokenized input used here carries no real temporal order for a backward pass to exploit, since the token order is

a modeling choice rather than a genuine temporal signal. This outcome is therefore close to what one would predict beforehand rather than a surprising discovery. The contribution of the ablation is to confirm that expectation empirically, with matched-seed statistical testing and, for one mechanism, a pre-specified equivalence test, rather than to overturn an open question. For dilated gated convolution specifically, a two one-sided tests equivalence check with a margin fixed in advance at 0.5 macro-F1 points establishes statistical equivalence between its two forms ($p = 0.007$, Section 4). The same check does not establish equivalence for the other two mechanisms at this sample size, where the confidence interval on the mean difference remains too wide to rule out a practically larger gap. The strongest configuration overall, dilated gated convolution in its unidirectional form, reached within 0.97 macro-F1 points of Extreme Gradient Boosting on TON_IoT. This was achieved at only 402K parameters (95 percent confidence interval 0.85 to 1.10 points, Welch t-test $p < 0.001$). This gap reflects a three-seed mean with a standard deviation of 0.05 points. Gradient-boosted trees won outright on TON_IoT still. A feature-ordering check, now averaged over three seeds per ordering rather than a single run, shows this margin is not fully order-independent. The tokenization order used throughout this study was the strongest of four orderings tested for two of the three mechanisms, and statistically indistinguishable from the strongest for the third. The reported gap should therefore be read as achievable rather than guaranteed under an arbitrary ordering. The gap to Extreme Gradient Boosting widened considerably on CIC-IoT-2023. There, the same dilated gated mechanism in its bidirectional form trailed by 5.15 macro-F1 points, against 0.81 points on TON_IoT, consistent with the more diffuse feature signal reported in Table 7. Whether the bidirectionality ablation and the feature-grouped variants would show the same pattern on CIC-IoT-2023 was not tested here, and is left to future work. For deployments where a small accuracy gap against tree ensembles is an acceptable price for a compact, convolution-only architecture, the unidirectional dilated gated design is the practical recommendation from this study. Its accuracy is statistically equivalent, within the pre-specified half-point margin above, to its bidirectional counterpart, at half the parameters. A cheaper alternative, unidirectional depthwise-separable gated convolution, is available at a further cost of 0.59 macro-F1 points where minimizing parameter count, FLOPs, and latency matters more than that margin (Table 6).

Future studies should extend the bidirectionality ablation and the feature-grouped variants to CIC-IoT-2023, add multi-seed confirmation with confidence intervals for the remaining single-seed configurations, and test robustness against adversarially perturbed inputs. Testing the feature-grouped variants in unidirectional form would also show whether the bidirectionality finding of Table 2 extends beyond the three plain mechanisms. The feature-ordering check of Table 4 should be extended to more than four orderings, to establish how much of the reported margin against Extreme Gradient Boosting depends on the specific tokenization used here. Later work should as well pursue heavier feature selection and ensemble tuning of the classical baselines themselves, given that published results on the same TON_IoT subset exceed the classical figures reported here. Extending the CPU latency measurement of Table 6 to real embedded or edge-gateway hardware would also confirm whether the ranking observed here transfers to the resource-constrained devices this architecture targets.

Author Contributions

Ömer Faruk Sarıtaş conceived the study and was responsible for the methodology, software, formal analysis, investigation, data curation, and writing (original draft, review and editing). The author has read and approved the final version of the study.

Declaration

This paper is not derived from any thesis or dissertation.

Conflict of Interest

The author declares no conflict of interest.

Ethical Review and Approval

No approval from the Board of Ethics is required, since this study relies exclusively on publicly available, previously anonymized benchmark datasets and involves no human or animal subjects.

Artificial Intelligence (AI) Use Statement

AI tools were used only for improving language, readability, or reference formatting. The author is solely responsible for the content, accuracy, and integrity of the work.

Funding

This research received no external funding.

Data Availability Statement

The TON_IoT [17] and CIC-IoT-2023 [18] datasets analyzed in this study are publicly available from their original publishers. The processed data, trained models, and code that support the findings of this study are available from the corresponding author upon reasonable request.

References

[1] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, K. Kavukcuoglu, *WaveNet: A generative model for raw audio*, 9th ISCA Speech Synthesis Workshop, Sunnyvale, California, 2016, pp. 125. [link]

[2] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, G. D. Hager, *Temporal convolutional networks for action segmentation and detection*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017) 1003–1012. [link]

[3] F. Chollet, *Xception: Deep learning with depthwise separable convolutions*, IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, Hawaii, 2017, pp. 1800-1807. [link]

[4] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, *Going deeper with convolutions*, IEEE Conference on Computer Vision and Pattern Recognition, Boston, Massachusetts, 2015, pp. 1–9. [link]

[5] L. Breiman, *Random forests*, Machine Learning 45 (1) (2001) 5–32. [link]

[6] T. Chen, C. Guestrin, *XGBoost: A scalable tree boosting system*, Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, New York, 2016, pp. 785–794. [link]

[7] L. Grinsztajn, E. Oyallon, G. Varoquaux, *Why do tree-based models still outperform deep learning on typical tabular data?*, Proceedings of the 36th International Conference on Neural Information Processing Systems, New Orleans, Louisiana, 2022, pp. 507–520. [link]

[8] J. H. Friedman, *Greedy function approximation: A gradient boosting machine*, The Annals of Statistics 29 (5) (2001) 1189–1232. [link]

[9] T. Cover, P. Hart, *Nearest neighbor pattern classification*, IEEE Transactions on Information Theory 13 (1) (1967) 21–27. [link]

[10] R. Nazre, R. Budke, O. Oak, S. Sawant, A. Joshi, *A temporal convolutional network-based approach for network intrusion detection*, International Conference on Integrated Intelligence and Communication Systems, Kalaburagi, 2024, pp. 1–6. [link]

[11] M. Gombar, A. Topalović, M. Pejić Bach, *Cost-aware lightweight deep learning for intrusion detection: A comparative study on UNSW-NB15 and CIC-IDS2017*, Electronics 15 (8) (2026) 1603. [link]

[12] N. Moustafa, J. Slay, *UNSW-NB15: A comprehensive data set for network intrusion detection systems*, Military Communications and Information Systems Conference, Canberra, 2015, pp. 1–6. [link]

[13] I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani, *Toward generating a new intrusion detection dataset and intrusion traffic characterization*, Proceedings of the 4th International Conference on Information Systems Security and Privacy, Funchal, 2018, pp. 108–116. [link]

[14] J. Chen, T. Lv, S. Cai, L. Song, S. Yin, *A novel detection model for abnormal network traffic based on bidirectional temporal convolutional network*, Information and Software Technology 157 (2023) 107166. [link]

[15] E. Akpaku, J. Chen, M. Ahmed, R. N. A. Sosu, F. K. Agbenyegah, D. K. Louis, *eBiTCN: Efficient bidirectional temporal convolution network for encrypted malicious network traffic detection*, Journal of Computer Security 33 (3) (2025) 180–211. [link]

[16] F. Wang, Y. Huang, Y. Shi, *Bidirectional temporal attention convolutional networks for high-performance network traffic anomaly detection*, Information 17 (1) (2026) 61. [link]

[17] A. Alsaedi, N. Moustafa, Z. Tari, A. Mahmood, A. Anwar, *TON_IoT telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems*, IEEE Access 8 (2020) 165130–165150. [link]

[18] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, A. A. Ghorbani, *CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment*, Sensors 23 (13) (2023) 5941. [link]

[19] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, *SMOTE: Synthetic minority over-sampling technique*, Journal of Artificial Intelligence Research 16 (2002) 321–357. [link]

[20] I. Loshchilov, F. Hutter, *Decoupled weight decay regularization*, International Conference on Learning Representations (ICLR), New Orleans, Louisiana, 2019. [link]

[21] D. Lakens, *Equivalence tests: A practical primer for t tests, correlations, and meta-analyses*, Social Psychological and Personality Science 8 (4) (2017) 355–362. [link]

[22] H. Benaddi, M. Jouhari, N. Laamech, A. Motii, K. Ibrahimi, *Lightweight intrusion detection in IoT via SHAP-guided feature pruning and knowledge-distilled Kronecker networks*, International Conference on Advanced Communication Technologies and Networking, Rabat, 2025, pp. 1–6. [link]

[23] W. M. Bahgat, T. A. Farrag, A. E. E. Rashed, *An integrated three-phase framework with feature selection and Bayesian optimization for multiclass intrusion detection on ToN_IoT datasets*, Neural Computing and Applications 38 (10) (2026) 363. [link]

[24] G. Andresini, F. Pendlebury, F. Pierazzi, C. Loglisci, A. Appice, L. Cavallaro, *INSOMNIA: Towards concept-drift robustness in network intrusion detection*, Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security, Online, 2021, pp. 111–122. [link]

[25] G. Apruzzese, A. Fass, F. Pierazzi, *When adversarial perturbations meet concept drift: An exploratory analysis on ML-NIDS*, Proceedings of the 2024 Workshop on Artificial Intelligence and Security, Salt Lake City, Utah, 2024, pp. 149–160. [link]

[26] K. He, D. D. Kim, M. R. Asghar, *Adversarial machine learning for network intrusion detection systems: A comprehensive survey*, IEEE Communications Surveys & Tutorials 25 (1) (2023) 538–566. [link]