Feature Selection Versus Feature Extraction for IoT Intrusion Detection Under Temporal Evaluation

Mahdia Abdessamad, El Bachir Tazi
Polydisciplinary Faculty of Taza, Sidi Mohamed Ben Abdellah University, Fez, Morocco

Abstract*—Comparative studies of feature selection (FS) and feature extraction (FE) for IoT intrusion detection rely almost universally on random train/test splits, which let temporally adjacent, highly similar records appear on both sides of the partition. This study asks whether such conclusions survive when this evaluation leakage is removed. We reproduce the detection-performance component of the framework of Li et al. on the TON-IoT network dataset — identical preprocessing, correlation-based selection versus PCA-based extraction, five classifiers, and matched dimensionalities $K \in \{9, 22, 33, 47, 77\}$ — varying only the partitioning rule across random stratified, global temporal, and per-class temporal protocols. At K = 9, the reference Decision Tree falls from 86.81% binary macro-F1 under random splitting to 58.04% under the global temporal protocol, in which unseen attack classes constitute 45.6% of the test records. In the multiclass task at K = 47, the same configuration declines from 93.14% to 14.26%, while the mean Matthews correlation coefficient (MCC) across 100 configurations falls from 0.638 to 0.196, a relative loss of approximately 69%. The family-level recommendation is directionally robust: for the single redundancy-oriented correlation selector evaluated here, PCA-based extraction outperforms selection in all three protocols (Wilcoxon over 50 matched pairs per protocol, $p < 10^{-6}$). Configuration-level orderings and the identity of the best classifier, however, are protocol-dependent. Established for the specific selector and extractor evaluated, these findings indicate that feature-reduction guidance for IoT intrusion detection should be validated under leakage-resistant protocols, with macro-averaged metrics and MCC reported alongside accuracy.

Keywords*—Internet of things; intrusion detection; feature selection; feature extraction; data leakage; temporal evaluation; concept drift; TON-IoT; machine learning

I. Introduction

The Internet of Things (IoT) connects a rapidly growing number of heterogeneous physical devices across domains such as smart homes, transportation, healthcare, and industrial automation, enabling intelligent services through interconnected sensing and communication technologies [1]. However, their limited computational resources make it difficult to implement robust security mechanisms, leaving them vulnerable to a wide range of cyberattacks. Consequently, network Intrusion Detection Systems (NIDS) have become an important line of defense for protecting IoT networks [2].

Network traffic datasets often contain numerous connection features, motivating the use of feature reduction techniques for intrusion detection [3]. Feature reduction methods are broadly categorized into feature selection (FS), which retains a subset of the original features, and feature extraction (FE), which transforms the original features into a lower-dimensional representation. Feature selection removes irrelevant and redundant features, reducing computational cost while improving detection performance and preserving the interpretability of the retained features [4]. Feature extraction produces compact representations that can combine information from all original features, at the cost of interpretability. Selecting an appropriate feature reduction strategy remains a recurring practical decision, motivating numerous empirical comparisons between feature selection and feature extraction [3], [4].

Among the existing comparisons, the study by Li et al. [3] provides a comprehensive evaluation of Feature Selection (FS) and Feature Extraction (FE) for IoT intrusion detection. Using the TON-IoT dataset, the authors compared correlation-based FS with PCA-based FE under the same reduced feature dimensions and evaluated five machine learning classifiers for both binary and multiclass classification. Their results showed that FE generally achieved better detection performance when only a small number of features were retained and was less sensitive to changes in the number of retained features, whereas FS required less training and inference time and became more competitive as the number of retained features increased. Ngo et al. [4] conducted a similar comparison on the UNSW-NB15 dataset using Pearson correlation-based feature selection and PCA-based feature extraction and reported similar findings.

Many existing studies evaluate intrusion detection models using random train/test splits rather than temporal splits. However, random splitting can place temporally related or highly similar traffic samples in both the training and test sets, leading to overly optimistic performance estimates that may not reflect real-world deployment. Pendlebury et al. demonstrated that temporal and spatial experimental biases can substantially inflate the reported performance of machine learning security models, while Arp et al. identified data snooping as one of the most prevalent pitfalls in machine learning-based security research, reporting that it was at least partly present in 73% of the reviewed studies [5], [6].

Recent studies across security and applied machine-learning domains have adopted temporal evaluation, including adversarially robust NIDS validated on official temporal partitions [7], industrial anomaly-detection benchmarks that compare random and temporal evaluation protocols [8], memory-analysis malware detection evaluated under concept drift [9], and IoT denial-of-sleep attack detection evaluated using chronological splits [10]. These studies demonstrate the increasing use of temporal evaluation for validating individual detection models.

Despite the increasing adoption of temporal evaluation, existing studies primarily evaluate the robustness of individual intrusion detection models. To the best of our knowledge, no study has investigated whether the comparative conclusions of the feature selection versus feature extraction literature remain stable when evaluation changes from random train/test splitting to temporally consistent protocols. Consequently, it remains unclear whether the relative ranking of feature reduction strategies is robust to evaluation methodology.

This study addresses that gap by investigating whether the comparative conclusions reported for feature selection and feature extraction remain valid under temporally consistent evaluation. The research question is: Do the comparative conclusions regarding feature selection and feature extraction remain valid under temporally consistent evaluation compared with conventional random train/test splitting? To answer this question, we preserve the experimental framework of Li et al. [3]including the dataset, preprocessing, feature reduction methods, classifiers, hyperparameters, and matched feature dimensionalities, while modifying only the evaluation protocol. The main contributions of this study are as follows:

- We revisit the experimental framework of Li et al. under identical preprocessing, feature reduction methods, and classifiers, with a single documented adjustment to the MLP iteration limit, establishing a random-split baseline that qualitatively reproduces their principal detection-performance findings.
- We define three evaluation protocols, including a global temporal split and a per-class temporal split, designed to reduce evaluation leakage relative to random splitting and to enable the degradation analysis presented in Section IV-D, subject to the dataset limitation identified in Section IV-F.
- We analyze how the evaluation protocol affects the relative ranking of feature selection and feature extraction methods, as well as classifier rankings, using Wilcoxon signed-rank tests and Kendall's τ rank-correlation coefficients.
- We analyze the impact of temporal evaluation on accuracy, precision, recall, macro-F1, and the Matthews correlation coefficient (MCC), showing that these metrics degrade to different extents, with MCC exhibiting the largest relative loss, and that accuracy alone can remain moderate while macro-averaged performance falls to very low absolute levels.
- We decompose the performance degradation of a fixed reference classifier into within-class temporal drift and unseen-class effects, showing that the relative weight of the two mechanisms depends on dimensionality, classifier, and reduction method, and we highlight a structural limitation of TON-IoT — whose attack classes are captured in short bursts — for per-class temporal evaluation.

II. Related Work

A. Feature Selection and Feature Extraction in Intrusion Detection

Feature selection methods for NIDS are conventionally organized into three families. Filter methods score features independently of any classifier using statistical criteria such as correlation, mutual information, information gain, or chi-square. Li et al. [3] adopt a correlation-based criterion that removes mutually redundant features. Hall's Correlation-based Feature Selection (CFS) framework [11] is the canonical relevance-and-redundancy formulation of this family, combining feature–class correlation with feature–feature inter-correlation. Wrapper methods evaluate candidate subsets by training a classifier on each, offering higher accuracy at substantially greater computational cost. Embedded methods perform selection during model training, for example through tree-based impurity importance or regularization penalties. Feature extraction methods are correspondingly organized by the nature of the transformation: linear projections such as principal component analysis, kernel-based nonlinear projections, and deep representation learning through autoencoders [12]. Because autoencoder-based extraction incurs substantially higher computational cost in both training and inference than statistical projection, PCA is commonly adopted for latency-sensitive IoT deployments [3].

Existing studies have more often evaluated individual feature reduction techniques than directly compared feature selection and feature extraction under a common experimental protocol. Ngo et al. [4] compared correlation-based selection against PCA on UNSW-NB15 across several classifiers and dimensionalities and reported that extraction was preferable at very small dimensionalities while selection became competitive and computationally cheaper as dimensionality increased. Li et al. [3] performed the most comprehensive comparison of which we are aware on Network TON-IoT, spanning five classifiers, matched dimensionalities of 9, 22, 33, 47, and 77 features, and both binary and multiclass tasks; they additionally reported feature-reduction, training, and inference times, concluding that the Decision Tree offered the best accuracy–runtime compromise for both families. Related studies have examined individual reduction techniques rather than family-level comparisons: ensemble and hybrid selection strategies [13]and stacked sparse autoencoder extraction evaluated for its effect on downstream classifier accuracy and runtime [12]. What unites this body of work, and motivates the present study, is that its conclusions are established exclusively under random partitioning.

B. Evaluation Bias and Data Leakage in Security Machine Learning

The reliability of evaluation protocols is a long-standing concern. Ambroise and McLachlan [14] showed that performing feature reduction outside a properly isolated evaluation loop yields severely biased error estimates — a principle directly relevant to comparative evaluations involving feature selection or feature extraction. Pendlebury et al. [5] introduced the TESSERACT framework, distinguishing temporal bias (incorrect time ordering between training and test data) from spatial bias (unrealistic class distributions at test time), and

demonstrated that removing both deflates malware-detection scores that had appeared near-perfect. Arp et al. [6] catalogued ten recurring pitfalls across thirty high-profile security papers, finding sampling bias and data snooping to be the most prevalent, and explicitly warned that a NIDS may learn to recognize an address range rather than generic attack behavior. A dataset-quality study documented labeling and construction artifacts in the widely used CICIDS2017 dataset [15]. For IoT specifically, Chikezie et al. [16] proposed a scientific integrity framework that established a device-disjoint, leakage-audited preprocessing substrate for the large CICIoT-DIAD 2024 corpus while explicitly deferring classifier-level conclusions to later experiments. A recent survey of deep-learning-based flow intrusion detection likewise catalogues evaluation failure modes that can inflate reported results — including temporal leakage, flawed data splitting, and weak cross-dataset generalization [17]. To the best of our knowledge, none of these studies examine whether comparative FS-versus-FE guidance survives leakage-resistant evaluation.

C. Temporal Evaluation Across Application Domains

Temporal protocols are increasingly adopted as deployment-faithful practice beyond malware analysis. Hybrid deep NIDS architectures have been validated under the official temporal partition of UNSW-NB15 to confirm that binary detection generalizes under distribution shift [7]. In industrial visual anomaly detection, a recent benchmark adopts a temporal train/test split as its primary protocol and defines the difference in performance between random and temporal splits as an explicit generalization metric, reporting that method rankings change between protocols [8]. Memory-analysis malware detection has been re-evaluated under temporal splits to capture concept drift [9], and IoT denial-of-sleep detection has been confirmed under chronological evaluation [10]. Cross-dataset studies reinforce this warning: although Random Forest achieved 95.08% accuracy on UNSW-NB15 and 99.79% on TON-IoT under same-dataset evaluation, its accuracy dropped below 40% when evaluated across datasets, demonstrating that strong within-dataset performance may substantially overestimate real-world generalisation and deployable performance [18]. Collectively, these works validate individual detectors under time-consistent protocols. The present study extends this line of enquiry in a different direction: rather than asking whether a given detector remains effective, it asks whether a comparative methodological conclusion—the FS-versus-FE ranking—remains valid, which requires holding the entire experimental pipeline fixed while varying only the partitioning rule, whose downstream effects on temporal window and class availability are themselves the object of study.

III. Methodology

The design principle of this study is the strict isolation of a single experimental variable. We reproduce the experimental framework of Li et al. [3], including the preprocessing pipeline, feature-reduction methods, classifiers, and hyperparameter settings (specifically, the detection-performance component of their study; feature-reduction and inference timings are outside the present scope), with the sole exception of the implementation adjustment described in Section III-B. Within this reproduced framework, the only experimental factor deliberately varied is the data-partitioning rule; because this rule governs how records are assigned to the training and test sets, it necessarily induces the differences in temporal window and class availability documented in Section III-C, which are consequences of the partitioning rule rather than independently manipulated variables. Consequently, comparisons among the evaluation protocols are conducted under otherwise identical experimental conditions, allowing the influence of the evaluation protocol to be assessed independently. Fig. 1 summarizes the complete pipeline and identifies the stage at which the three evaluation protocols diverge.

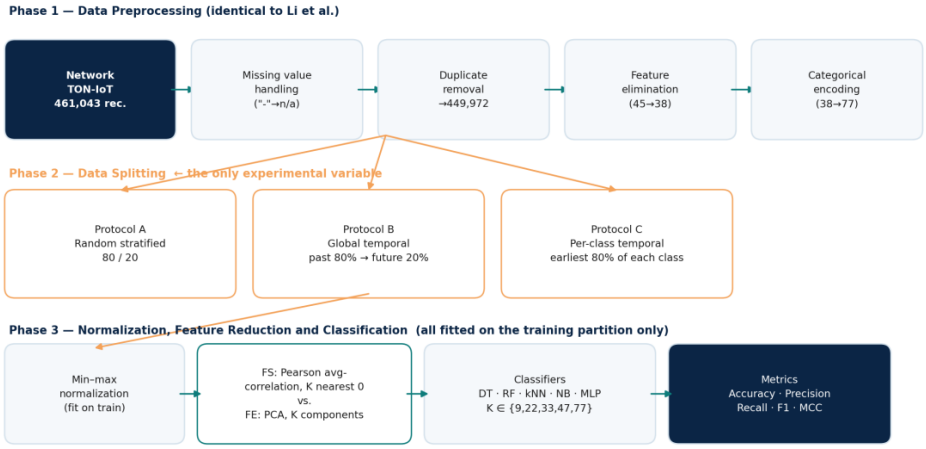

Fig. 1. Experimental pipeline. Only the data-splitting stage (Phase 2) varies across the three evaluation protocols.

A. Dataset

Experiments use the TON-IoT Network dataset [19] in its standard Train_Test_Network.csv form, a benchmark subset of the complete TON-IoT network dataset intended for fair evaluation of AI-based cybersecurity solutions. The dataset contains 461,043 labeled network flow records extracted from Zeek, organized into 45 columns: the timestamp attribute (ts), four flow identifiers (source and destination IP addresses and ports), 38 traffic features, and two labels: a binary label (normal/attack) and a ten-class type label representing normal

traffic and nine attack categories (backdoor, DDoS, DoS, injection, MITM, password, ransomware, scanning, and XSS). The timestamp attribute is retained to construct the temporal evaluation protocols but excluded from the feature matrix.

B. Preprocessing and Feature Reduction Pipeline

The preprocessing sequence, feature-reduction methods, classifier configurations, and hyperparameter settings are kept identical across all experiments; the only experimental variable is the data partitioning strategy (Phase 2 in Fig. 1). Table I summarizes each preprocessing and reduction step together with its output, and Table II lists the classifier hyperparameters. To prevent information leakage, all fitted transformations—min–max normalization, Pearson correlation ranking, and PCA—are fitted exclusively on the corresponding training partition and then applied to the associated test partition.

TABLE I. PREPROCESSING AND FEATURE REDUCTION PIPELINE (IDENTICAL ACROSS ALL PROTOCOLS)

Step	Operation	Output
1	Input: Train_Test_Network.csv	461,043 records × 45 columns
2	Missing-value handling: '-' entries replaced with an explicit 'n/a' category	—
3	Duplicate removal, applied to the raw records before identifier removal (11,071 exact duplicates)	449,972 records
4	Feature elimination: timestamp and the four flow identifiers excluded from the feature matrix	38 traffic features
5	Categorical encoding: one-hot for low-cardinality attributes (proto, service, conn_state, DNS/SSL flags, http_method, http_version); binary presence encoding for high-cardinality text attributes (dns_query, ssl_version/cipher/subject/issuer, http_uri, http_user_agent, MIME types, weird_* attributes, http_trans_depth)	77 numerical features
6	Data partitioning: Protocol A, B, or C (Section III-C)	80% training / 20% test
7	Min–max normalization, fitted on the training partition only	features scaled to [0, 1]
8	Feature reduction, fitted on the training partition only — FS: Pearson average-correlation score, K features nearest zero; FE: PCA, first K components; K ∈ {9, 22, 33, 47, 77}	K features/components
9	Classification with the five classifiers of Table II; three seeds (42, 43, 44) for stochastic models	metrics of Section III-D

TABLE II. CLASSIFIER HYPERPARAMETER SETTINGS

Classifier	Hyperparameter settings
Decision Tree (DT)	criterion: Gini; splitter: best; max_depth: none; random_state: seed
Random Forest (RF)	n_estimators: 100; criterion: Gini; max_depth: 5; random_state: seed
k-Nearest Neighbors (kNN)	n_neighbors: 3; Euclidean distance
Gaussian Naïve Bayes (NB)	default parameters
Multi-Layer Perceptron (MLP)	hidden_layer_sizes: 100; activation: ReLU; alpha: 10^{-4}; batch_size: auto; learning_rate: constant; max_iter: 500 with early stopping (adjusted from 200; see text)

Feature selection (FS) computes the Pearson correlation matrix on the training partition and retains the K features whose average pairwise correlation is closest to zero. Feature extraction (FE) applies PCA fitted on the training partition and projects both partitions onto the first K principal components. The matched dimensionalities $K \in \{9, 22, 33, 47, 77\}$, with $K = 77$ corresponding to the full encoded feature set and serving as a no-reduction control.

One implementation adjustment was required for the Multi-Layer Perceptron (MLP). The original configuration specified a maximum of 200 training iterations; however, under our implementation, the model did not consistently converge within this limit. The maximum number of iterations was therefore increased to 500, and early stopping was enabled. This adjustment was applied uniformly across all evaluation protocols to preserve a fair comparison.

C. Evaluation Protocols

The three evaluation protocols differ only in how the training and test partitions are constructed; all subsequent preprocessing, feature reduction, classifier configurations, and hyperparameters remain identical.

Protocol A — Random Stratified Split. Records are partitioned into 80% training and 20% testing using random stratified sampling on the ten-class label. This protocol serves as the baseline and enables direct comparison with the published results.

Protocol B — Global Temporal Split. To isolate the effect of temporal evaluation, the same 80/20 partition ratio is preserved while replacing random sampling with chronological ordering. Records are sorted by timestamp; the earliest 80% form the training set and the latest 20% form the test set. Class balance is intentionally not enforced. Because attack campaigns in the dataset occur during distinct time intervals, the resulting partitions are strongly imbalanced with respect to class availability, as shown in Table III. This protocol most closely reflects operational deployment, where models are trained on historical traffic and evaluated on future traffic, and naturally introduces a partial zero-day scenario (using "zero-day" in the evaluation sense common in the IDS literature, i.e., attack classes absent from training rather than previously unknown vulnerabilities).

TABLE III. CLASS AVAILABILITY UNDER PROTOCOL B (GLOBAL TEMPORAL SPLIT)

Class	In training partition	In test partition	Training records	Test records	Availability
normal	Yes	Yes	253,015	35,914	Seen
ddos	Yes	No	20,000	0	Training only
dos	Yes	No	20,000	0	Training only
injection	Yes	No	20,000	0	Training only
password	Yes	No	20,000	0	Training only
scanning	Yes	No	20,000	0	Training only
xss	Yes	Yes	6,962	13,038	Seen
backdoor	No	Yes	0	20,000	Unseen (zero-day)
mitm	No	Yes	0	1,043	Unseen (zero-day)
ransomware	No	Yes	0	20,000	Unseen (zero-day)
Total	—	—	359,977	89,995	—

a. Record counts are obtained directly from the partitioning code used in the experiments.

Record counts are obtained directly from the partitioning code used in the experiments. Five attack classes occur only in the training period and three only in the test period; the unseen classes account for 41,043 of the 89,995 test records (45.6%).

Protocol C — Per-Class Temporal Split. The same temporal principle is applied independently within each class. Records belonging to each class are ordered chronologically, and the earliest 80% are assigned to the training set while the latest 20% form the test set. Every class is therefore represented in both partitions while temporal ordering is preserved within each class.

Because Protocol A applies stratified sampling and Protocol C applies an 80/20 split within each class, both protocols produce identical class proportions in the training and test partitions; they differ only in whether the assignment within each class is random or chronological. This protocol is therefore designed to isolate within-class temporal drift while eliminating the missing-class effect, although Protocols B and C do not evaluate on identical time windows, since Protocol C draws its test partition from within each class rather than from a single terminal period.

Consequently, the difference between Protocols A and C estimates the impact of within-class temporal drift, whereas the difference between Protocols C and B estimates the additional effect of previously unseen attack classes, under the assumptions discussed in Section IV-D.

Quantitatively, the train–test class-prior divergence differs sharply across the protocols: computed directly from the partition counts (Table III for Protocol B), the total-variation distance between the training and test class distributions is approximately zero under Protocols A and C — both equalize class proportions by construction — and 0.58 under Protocol B. Because Protocol C draws each test segment from within its class's own capture span rather than from a single terminal window, Protocols B and C also differ in their boundary timestamps; the per-class temporal spans reported in Section IV-F (93 s to 8,687 s for the attack classes against 25.7 days for benign traffic) bound the temporal windows involved.

D. Evaluation Metrics

The same evaluation metrics are used across all three protocols: accuracy, precision, recall, F1-score, and the Matthews Correlation Coefficient (MCC). Precision, recall, and F1-score are reported as macro averages. Macro-averaging is particularly appropriate for the present study because it assigns equal weight to every class regardless of prevalence. Under the temporal protocols, the test-set class composition can diverge substantially from that of the training set—for example, under Protocol B, classes absent from training account for 45.6% of the test records (Table III)—so accuracy may remain moderate while macro-averaged performance falls to substantially lower absolute levels. Macro-averaged metrics expose this behavior, whereas accuracy or micro-averaged metrics would largely reflect the performance of the dominant classes. MCC complements these measures by summarizing the entire confusion matrix into a single coefficient that remains comparable when the test-set class composition changes across protocols, as it does here. Because Decision Trees, Random Forests, and MLPs involve randomized training procedures, each experiment is repeated using three random seeds (42, 43, and 44), and the results are reported as mean ± standard deviation. Deterministic methods (k-Nearest Neighbors, Gaussian Naïve Bayes, PCA, and correlation-based feature selection) are executed once.

E. Implementation Environment

All experiments were executed within a single implementation in which one preprocessing pipeline was applied to all three protocols, ensuring that the protocols differ in nothing but the partitioning rule. Table IV summarizes the computational environment.

TABLE IV. HARDWARE AND SOFTWARE SPECIFICATIONS OF THE IMPLEMENTATION ENVIRONMENT

Component	Description
Computing platform	Kaggle Notebooks (CPU runtime)
Processor	AMD EPYC 7B12, 4 virtual CPUs
RAM	30 GB
Operating system	Linux 6.12.90+ (x86_64)
Python	3.12.13
Core packages	Pandas 2.3.3; NumPy 2.0.2; SciPy 1.16.3; Scikit-learn 1.6.1
Random seeds	42 (global); 42, 43, 44 (repetitions)

IV. RESULTS AND DISCUSSION

The analysis addresses the research question through four complementary dimensions: whether the evaluation protocol changes measured performance (Section IV-A), whether the FS-versus-FE ranking remains stable (Section IV-B), how the individual metrics respond (Section IV-C), and how the observed degradation decomposes into distinct mechanisms (Section IV-D). Sections IV-E to IV-G then examine classifier-level behavior, a structural limitation of the dataset, and the practical implications of these findings.

Tables V and VI summarize the results using macro-F1 as the primary performance metric. For each protocol, method, and feature dimensionality, the classifier with the highest macro-F1 score is reported. Complete results for all five evaluation metrics (accuracy, precision, recall, macro-F1, and MCC) for every classifier and configuration are provided in Tables IX–XXXVIII of the Appendix.

TABLE V. BINARY CLASSIFICATION — MACRO-F1 (%) ACROSS THREE EVALUATION PROTOCOLS (BEST CLASSIFIER PER CELL)

K	Rand FE	Rand FS	Glob-Temp FE	Glob-Temp FS	PerClass FE	PerClass FS
9	86.81 (DT)	78.55 (DT)†	66.23 (kNN)	37.00 (NB)	69.83 (MLP)	61.84 (DT)†
22	94.31 (DT)	79.28 (DT)†	67.57 (DT)	61.95 (kNN)	81.88 (DT)	62.74 (DT)
33	98.10 (DT)	85.86 (DT)	83.56 (DT)	65.17 (kNN)	86.63 (DT)	70.15 (DT)
47	99.15 (DT)	94.44 (DT)	93.08 (kNN)	72.38 (DT)	95.56 (DT)	94.16 (DT)
77	99.18 (DT)	99.26 (DT)	94.53 (kNN)	92.68 (kNN)	95.59 (DT)	95.73 (DT)

b. Rand = Protocol A (random stratified); Glob-Temp = Protocol B (global temporal); PerClass = Protocol C (per-class temporal). Winning classifier in parentheses.

c. Exact tie between two or more classifiers at full precision; ties are credited to the Decision Tree here and in the dominance counts of Section IV-E.

TABLE VI. MULTICLASS CLASSIFICATION — MACRO-F1 (%) ACROSS THREE EVALUATION PROTOCOLS (BEST CLASSIFIER PER CELL)

K	Rand FE	Rand FS	Glob-Temp FE	Glob-Temp FS	PerClass FE	PerClass FS
9	46.15 (DT)	29.33 (DT)	12.96 (NB)	9.51 (kNN)	45.10 (DT)	27.77 (DT)‡
22	71.91 (DT)	32.42 (DT)	14.88 (NB)	11.09 (DT)	62.84 (DT)	30.47 (MLP)
33	81.58 (DT)	41.60 (DT)	11.70 (DT)	10.19 (MLP)	68.08 (DT)	42.15 (DT)
47	93.14 (DT)	72.16 (DT)	14.26 (DT)	11.43 (DT)	85.85 (DT)	67.30 (DT)
77	93.13 (DT)	93.65 (DT)	14.31 (DT)	14.10 (DT)	86.70 (DT)	88.06 (DT)

d. Decision Tree and MLP are tied at two decimals (27.77) in the per-class selection cell at K = 9; the Decision Tree leads at full precision (27.7720 against 27.7699) and is credited with the cell.

A. Performance Degradation Under Temporal Evaluation

Fig. 2 previews the protocol effect analyzed in this subsection. Cross-protocol comparisons require a fixed reference classifier, since the best-performing model differs across configurations (Tables V and VI). The Decision Tree is adopted as the reference throughout, for two reasons: it is the strongest and most stable classifier in this study, ranking first or tying for first in 47 of the 60 best-per-cell entries (44 outright wins and 3 ties; Section IV-E), including 25 of the 30 multiclass entries, and it was likewise identified by Li et al. as the preferred classifier within their framework. Holding the Decision Tree fixed and using PCA-based extraction at K = 9, binary performance under Protocol A reaches 87.89% accuracy, 86.84% macro-precision, 86.79% macro-recall, 86.81% macro-F1, and an MCC of 0.74. Under Protocol B, the same configuration attains 58.22%, 58.99%, 59.33%, 58.04%, and 0.18, respectively. All five measures decline together, but not proportionally: accuracy and macro-F1 lose approximately 29 percentage points, while MCC loses more than three-quarters of its value. The pattern persists at higher dimensionality — binary macro-F1 falls from 99.15% to 73.69% at K = 47 — and is far stronger in the multiclass task, where the same classifier and configuration declines from 93.14% to 14.26% macro-F1.

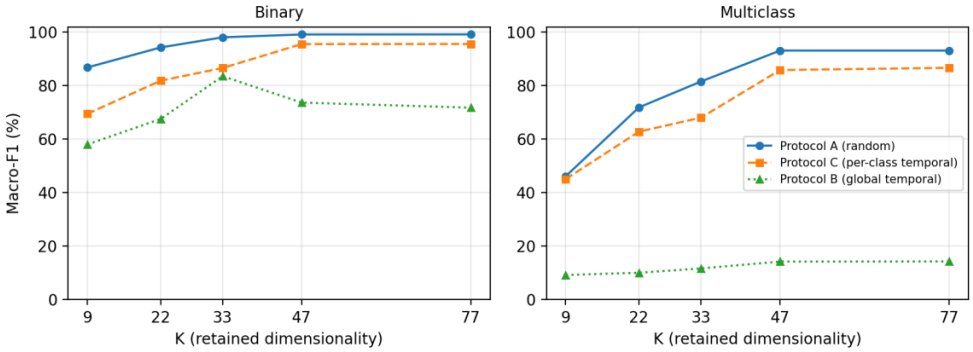

Fig. 2. Reference Decision Tree macro-F1 with PCA-based extraction as a function of the retained dimensionality K under the three evaluation protocols (left: binary task; right: multiclass task).

Because every component of the pipeline other than the data partitioning strategy is identical across protocols, these differences are attributable to the change in partitioning strategy under the controlled experimental conditions used in this study. Under Protocol A, both reduction methods approach ceiling performance at high dimensionality — 99.18% and 99.26% binary macro-F1 for extraction and selection, respectively (Table V), reproducing the near-perfect scores that characterize the random-split literature. Those scores were not attained under either temporal protocol.

Consistent with the reasoning of [5] and [6], these findings suggest that conventional random evaluation produces substantially more optimistic estimates than temporal evaluation, and that a considerable portion of the reported headroom in the FS-versus-FE literature reflects train–test similarity rather than genuine temporal generalization, at least for burst-captured corpora of the kind studied here. In such corpora, random partitioning distributes each short traffic burst across both partitions, so the classifier is evaluated on samples closely resembling those it was trained on. Under temporal partitioning, the traffic distribution is no longer assumed to match that of the training data, and the measured score reflects the deployment-relevant ability to classify traffic the model has not previously observed.

These fixed-classifier deltas deliberately upper-bound the penalty faced by a practitioner who is free to re-select the model per protocol: under Protocol B the best-available classifier recovers much of the loss, with the binary A-to-B gap under extraction at K = 77 shrinking from 27.39 points for the fixed Decision Tree to 4.53 points for k-Nearest Neighbors, the strongest model under Protocol B (99.06% to 94.53%), and from 25.46 to 5.92 points at K = 47. The Decision-Tree-based figures reported throughout should therefore be read as the penalty absent model re-selection, not as the best achievable under each protocol.

Scope of claim: the degradation magnitudes reported in this subsection are specific to TON-IoT and to the fixed reference classifier; they are not claimed to generalize to other corpora.

B. *Recommendation Stability and Ranking Instability Across Protocols*

The results in Tables V and VI operate at three distinct levels, which the analysis keeps separate: the family-level recommendation (whether extraction or selection is preferable), configuration-level rankings (the ordering of individual method–classifier–dimensionality combinations), and classifier rankings (which model performs best in a given setting). The evidence shows that the first is preserved across protocols, whereas the second and third are not, and that the magnitude of the family-level advantage itself varies with the protocol.

Under Protocol A, the pattern reported in [3] is confirmed: extraction leads selection clearly at small K, reaching 86.81% against 78.55% at K = 9 and 94.31% against 79.28% at K = 22, with the gap narrowing as dimensionality grows until selection marginally overtakes extraction at the full feature set (99.26% against 99.18%). This crossover mirrors the behavior reported by Li et al., who observed selection overtaking extraction at full dimensionality with F1-scores of 87.69% and 86.14%, respectively, under their best classifier. The absolute levels nevertheless differ: our Protocol A ceiling exceeds the values reported in [3] by roughly twelve macro-F1 points at full dimensionality. Because [3] reports aggregate rather than per-cell results for most configurations, an exhaustive cell-level reconciliation is not possible; the reproduction claim in this study is accordingly limited to the qualitative pattern — near-ceiling performance at high dimensionality and the same selection-overtakes-extraction crossover — rather than to absolute values, with the residual gap most plausibly reflecting implementation-level details (duplicate handling, encoder fitting, and the F1 averaging variant) that [3] does not fully specify. Under Protocol C, the same qualitative shape survives at reduced magnitude, with selection again edging ahead at K = 77 (95.73% against 95.59%); under Protocol B, extraction retains a small advantage at K = 77 (94.53% against 92.68%).

The Wilcoxon signed-rank tests in Table VII confirm the family-level recommendation formally: extraction outperforms selection in all three protocols across the 50 matched configurations, with median differences of +8.81, +2.75, and +13.07 macro-F1 points for Protocols A, B, and C, respectively (exact $p = 4.1 \times 10^{-9}$, 2.2×10^{-7}, and 6.6×10^{-12}). All statistics are computed on the full-precision experimental records rather than on the two-decimal values reported in the Appendix, under which no FS–FE pair is exactly tied.

TABLE VII. STATISTICAL ANALYSIS OF METHOD DIFFERENCES AND RANKING STABILITY

Test	Comparison	Statistic	p-value	Interpretation
Wilcoxon signed-rank	FE vs FS, Protocol A	median Δ = +8.81; FE wins 42/50	4.1×10^{-9}	FE superior across configurations
Wilcoxon signed-rank	FE vs FS, Protocol B	median Δ = +2.75; FE wins 40/50	2.2×10^{-7}	FE superior across configurations
Wilcoxon signed-rank	FE vs FS, Protocol C	median Δ = +13.07; FE wins 44/50	6.6×10^{-12}	FE superior across configurations
Kendall's τ	Protocol A vs C	τ = 0.751	1.8×10^{-28}	Substantial, imperfect agreement
Kendall's τ	Protocol A vs B	τ = 0.666	9.9×10^{-23}	Substantial, imperfect agreement
Kendall's τ	Protocol B vs C	τ = 0.560	1.5×10^{-16}	Moderate agreement

e. Δ denotes the difference in macro-F1 percentage points. Wilcoxon tests are paired over 50 matched FS/FE configurations per protocol; Kendall's τ (τ-b variant) is computed over 100 configuration-level macro-F1 values per protocol. All statistics use full-precision records; no FS–FE pair is exactly tied.

The advantage is therefore directionally robust, although its magnitude is protocol-dependent — the median margin under Protocol B is less than a third of that under Protocol A.

Configuration-level rankings are markedly less stable, as the Kendall's τ coefficients in Table VII show: 0.751 between Protocols A and C, 0.666 between A and B, and 0.560 between B and C. These values indicate substantial but clearly imperfect agreement — a non-trivial fraction of configuration-level orderings changes when the protocol changes. The most consequential instance concerns the classifier ranking. k-Nearest Neighbors, which is the best model in no binary cell under Protocol A, becomes the strongest classifier in six of the ten binary cells under Protocol B (Table V). A practitioner who selected both a reduction method and a classifier based on random-split evidence would therefore retain the correct method family but adopt a suboptimal classifier in the majority of binary configurations under temporally consistent evaluation — an observation that parallels the protocol-dependent method rankings reported in temporal benchmarks from other domains [8].

In summary, the family-level recommendation — prefer extraction at reduced dimensionality — survives the change of evaluation protocol, whereas the magnitude of its advantage, configuration-level orderings, and the identity of the best-performing classifier are all protocol-dependent. Comparative guidance derived from random-split evaluation should accordingly be validated under leakage-resistant protocols before informing design decisions.

Scope of claim: the directional FE-over-FS result is specific to the single redundancy-oriented selector evaluated; the ranking instability is specific to this dataset and protocol grid, though consistent with protocol-dependent rankings reported in other domains [8].

C. Differential Sensitivity of the Evaluation Metrics

The five metrics do not respond uniformly to the change of protocol, and the divergence is itself informative. Under Protocol B, accuracy can remain moderate while macro-averaged performance falls to a far lower absolute level: in the multiclass task at K = 47, the Decision Tree retains an accuracy of 48.60% while its macro-F1 falls to 14.26% — a gap of more than 34 percentage points between two measures of the same predictions.

The explanation is distributional. Accuracy is dominated by whichever classes are most frequent in the test window, so a model that continues to recognize benign traffic and one or two persistent attack families retains moderate accuracy. Macro-F1 averages per-class F1 with equal weight, so every class the model fails to detect — including the three classes that Table III shows are absent from training under Protocol B, which together account for 45.6% of the test records — contributes a value near zero and depresses the mean. The divergence between the two measures under Protocol B is therefore consistent with a loss of detection capability concentrated in the classes that accuracy weights least.

The Matthews correlation coefficient proves the most sensitive of the five measures in relative terms. Averaged over the 100 configurations under each protocol, MCC declines from 0.638 under Protocol A to 0.196 under Protocol B, a relative loss of approximately 69%, and falls below 0.30 in 76 of those configurations. The pattern holds within each task considered separately: binary MCC declines from 0.710 to 0.247 and multiclass MCC from 0.566 to 0.146. Ten Protocol B configurations — all on the selection side — yield negative MCC, reaching −0.35 for the binary K = 9 selection models; predictions there are anti-correlated with the true labels, and these cases coincide with the degenerate low-dimensionality selection subsets documented in the Appendix. The binary task at K = 9 illustrates the contrast within a single configuration: the Decision Tree's accuracy falls from 87.89% to 58.22% and its macro-F1 from 86.81% to 58.04%, while MCC drops from 0.74 to 0.18. Because MCC combines all four categories of correct and incorrect predictions, it remains responsive to imbalanced error structures that accuracy and F1 partially absorb. Precision and recall decline in parallel with F1 over the same transition, from 86.84% and 86.79% to 58.99% and 59.33%, respectively, indicating that the degradation is not confined to one side of the precision–recall trade-off but reflects a general loss of discriminative power.

These observations have a direct reporting consequence within the evaluated framework. Studies that report accuracy alone may substantially understate the effect of temporal evaluation, since accuracy is the measure least responsive to failures concentrated in infrequent or unseen attack classes. Within the evaluated framework, reporting macro-averaged metrics alongside MCC appears necessary for meaningful comparison across evaluation protocols, and the choice of

headline metric should be stated explicitly whenever protocols are compared.

Scope of claim: the metric-divergence pattern follows from macro-averaging and MCC under shifted class composition and can be expected wherever test-set composition diverges from training; the magnitudes are dataset-specific.

D. Separating Temporal Drift from Unseen-Class Effects

Contrasting the two temporal protocols suggests a decomposition of the observed degradation into two mechanisms. Because Protocol C retains every class on both sides of the boundary while Protocol B does not, the difference between Protocols A and C provides an estimate of within-class temporal drift, and the difference between Protocols C and B approximates the additional penalty attributable to the unseen attack classes documented in Table III. These components should be read as approximations rather than exact orthogonal effects: the two temporal protocols place their boundaries differently and therefore also differ in training-set composition, so class availability is not the only factor separating them. The classifier is held fixed throughout; the Decision Tree again serves as the reference model (Section IV-A). Table VIII presents the decomposition for binary classification with PCA-based extraction.

TABLE VIII. DECOMPOSITION OF PERFORMANCE DEGRADATION (BINARY, FEATURE EXTRACTION, DECISION TREE, MACRO-F1 %)

K	Protocol A (random)	Protocol C (per-class)	Protocol B (global)	A−C gap (drift est.)	C−B gap (unseen est.)
9	86.81	69.49	58.04	17.32	11.45
22	94.31	81.88	67.57	12.43	14.31
33	98.10	86.63	83.56	11.47	3.07
47	99.15	95.56	73.69	3.59	21.87
77	99.18	95.59	71.79	3.59	23.80

f. All values are Decision Tree macro-F1 taken from the Appendix tables; the classifier is held fixed so that the reported differences reflect the protocol alone. The two gaps are protocol differences interpreted as estimates of the drift and unseen-class effects under the assumptions stated in the text; they are not exactly separable causal components.

Table VIII reveals an interaction between dimensionality and degradation mechanism. The drift component declines monotonically as dimensionality grows, from 17.32 points at K = 9 to 3.59 points at K = 47 and K = 77, consistent with poor generalization to shifted traffic when few features are available even though all classes are represented in training. The unseen-class component varies non-monotonically — 14.31 points at K = 22 but only 3.07 at K = 33 — yet at K = 47 and K = 77 it reaches 21.87 and 23.80 points and clearly dominates the residual drift. For the evaluated configuration, higher dimensionality was thus associated with markedly smaller drift penalties but, at the largest dimensionalities, with substantially larger unseen-class penalties. A practitioner who increases dimensionality to improve temporal robustness could therefore be exchanging one vulnerability for a larger one, although this trade-off is established here only for the Decision Tree with PCA-based extraction on TON-IoT.

The multiclass results amplify this asymmetry. For the same Decision Tree at K = 47, macro-F1 declines from 93.14% under Protocol A to 85.85% under Protocol C but collapses to 14.26% under Protocol B, corresponding to a drift component of 7.29 points against an unseen-class component of 71.59 points. Within this experimental framework, neither of the two evaluated reduction methods enabled the classifiers to recognize genuinely novel attack categories. This limitation is inherent to closed-set classification, in which every test sample must be assigned to one of the classes observed during training; recognizing samples from previously unseen classes requires open-set formulations that provide an explicit rejection option [20]. Closed-set comparisons conducted under random partitioning cannot expose this deficiency, because the random protocol guarantees that every class appears in training.

Finally, the decomposition is itself model-dependent, which merits explicit reporting. For k-Nearest Neighbors at K = 9 (binary task, feature extraction), the ordering inverts: macro-F1 falls from 86.74% under Protocol A to 44.03% under Protocol C but recovers to 66.23% under Protocol B. In this configuration, instance-based prediction appears more sensitive to within-class drift than to a shifted class composition, so for this model the per-class protocol is the harsher test. Random Forest exhibits yet another profile in the binary task: its drift component remains between 16 and 23 points at every dimensionality while its unseen-class component contracts from roughly 12 points at K = 9–33 to about 3 points at K = 47 and 77 — the reverse of the Decision Tree's pattern. The two mechanisms are therefore not universally ordered by severity; their relative weight depends on dimensionality and the inductive bias of the classifier, and any single-figure summary of the temporal penalty should be interpreted with corresponding caution.

Scope of claim: the decomposition is approximate, classifier-dependent, and established on TON-IoT only; the dimensionality interaction is demonstrated for the Decision Tree with PCA-based extraction.

E. Classifier-Level Observations

The protocol effect is visible in every classifier, indicating that it is a property of the evaluation rather than of any particular model, yet individual sensitivities differ. k-Nearest Neighbors exhibits a dual character: it is the most drift-sensitive model at small K under Protocol C, reaching only 44.03% binary macro-F1 with extraction at K = 9, while simultaneously being the strongest model under Protocol B at several dimensionalities. This pattern is consistent with instance-based prediction degrading severely when within-class distributions drift, while remaining comparatively robust when the challenge is instead a shifted class composition, although the present experiments cannot establish this mechanism directly. The Decision Tree is the dominant model overall, ranking first or tying for first in 47 of the 60 best-per-cell entries across both tasks and all three

protocols (44 strict wins; the three exact ties, marked in Table V, arise in the degenerate low-dimensionality selection cells discussed in the Appendix) and in 25 of the 30 multiclass entries; four of the five multiclass exceptions occur under Protocol B, where severe degradation compresses the models into a narrow low-performance band; the fifth arises under Protocol C at K = 22 (per-class selection), where the MLP edges the Decision Tree by one-hundredth of a point (30.47 against 30.46). The MLP's multiclass macro-F1 plateaus near 41% for K = 33–77 even after the convergence adjustment described in Section III-B, suggesting a capacity or optimization limitation under the hyperparameters specified in [3]; because the Decision Tree dominates the multiclass rankings throughout, this does not affect any best-per-cell conclusion, but MLP multiclass values should be interpreted with this caveat.

Scope of claim: these classifier-level behaviours are observed on TON-IoT under the three protocols studied and for the inherited hyperparameters; they are specific to this dataset and configuration.

F. A Structural Limitation of Burst-Captured Datasets

One pattern in Table V requires a dataset-level explanation: under Protocol C, binary performance at high dimensionality recovers to near the random-split level, reaching 95.59% at K = 77. Inspection of the timestamps explains this. The attack classes of TON-IoT were generated in short scripted campaigns whose per-class time spans range from 93 seconds to 8,687 seconds, whereas benign traffic spans 25.7 days. Within a campaign lasting a few minutes, there is little temporal structure for a per-class chronological cut to exploit, so for attack classes Protocol C approaches a random split as K increases and the classifier gains capacity. The per-class temporal results therefore likely understate the within-class drift that would be observed over longer capture windows, with the genuine temporal signal carried primarily by the long-spanning benign class. This limitation suggests that future benchmark datasets would benefit from preserving device identity while also providing longer continuous capture windows to enable more rigorous temporal evaluation. Recent work has emphasized the importance of device-aware evaluation protocols and auditable evaluation substrates [16], although extending capture duration remains an open challenge.

Scope of claim: this is a structural property of TON-IoT's burst-captured attack classes; it bounds per-class temporal evaluation on this corpus and need not hold for datasets with longer capture windows.

G. Practical Implications

Three implications follow for practitioners, within the scope of the evaluated dataset, classifiers, and reduction methods. First, feature-reduction guidance derived from random-split studies should be treated as an optimistic upper estimate of achievable performance. The direction of the FS-versus-FE recommendation survived in these experiments — extraction outperformed the single redundancy-oriented correlation selector inherited from Li et al. in all three protocols — but its magnitude, and the classifier that best exploits it, changed under temporally consistent evaluation. Second, evaluation reports for IoT IDS should include macro-averaged metrics and MCC alongside accuracy and should state the partitioning protocol explicitly; the divergence of more than 34 percentage points between accuracy and macro-F1 observed in Section IV-C illustrates how much a single headline metric can conceal. Third, the interaction documented in Table VIII has a concrete design consequence: in the evaluated configuration, higher dimensionality was associated with smaller drift penalties but, at the largest dimensionalities, with greater exposure to unseen attack families (established here only for the Decision Tree with PCA-based extraction), so dimensionality should be selected with both failure modes in view rather than optimized against a random-split score alone. Because the attack classes of TON-IoT are captured in short bursts (Section IV-F), the drift components in Table VIII likely understate the within-class drift expected on corpora with longer capture windows, and this design consideration should be revalidated on such corpora. Robustness to genuinely novel attacks, in particular, was not achievable with the closed-set pipelines evaluated here — by construction, closed-set classification cannot reject samples from unobserved classes — and requires open-set or anomaly-based mechanisms [20] to be integrated into the design loop.

Scope of claim: these are practitioner-facing implications within the scope of the evaluated dataset, classifiers, and reduction methods; the methodological recommendations (validate under leakage-resistant protocols; report macro-F1 and MCC) are expected to generalise, whereas the specific magnitudes are dataset-specific.

V. Threats to Validity

Internal validity. The MLP required raising the iteration limit from the 200 specified by Li et al. [3] to 500 with early stopping to converge; this adjustment is documented in Section III-B, applied uniformly across all protocols, and does not affect best-per-cell conclusions, which are dominated by the Decision Tree and k-Nearest Neighbors. More broadly, hyperparameters are inherited from Li et al. rather than tuned per protocol. This is required by the isolation principle, and classifier rankings could differ under protocol-specific tuning; note, however, that those hyperparameters were themselves selected under random partitioning, so any residual advantage they confer accrues to Protocol A rather than to the temporal protocols, and the degradation reported here is therefore conservative.

Stochastic classifiers are averaged over three seeds, giving 3 protocols × 5 values of K × 2 tasks × 2 reduction families × 3 stochastic classifiers = 180 stochastic configurations. Across all 180, the mean seed-to-seed standard deviation of macro-F1 is 0.29 points, rising to 0.42 when restricted to the 123 configurations whose deviation is non-zero at two decimals; the maximum is 4.71 points. All of these are small relative to the protocol effects reported here, several of which exceed 25 points. Three replicates support only a coarse dispersion estimate rather than a confidence interval, and the figures above should be read accordingly. The principal ranking observation of Section IV-B is unaffected by this limitation: the k-Nearest Neighbors scores that dominate Protocol B are deterministic, and in five of the six binary cells it wins, its margin over the best stochastic competitor (7.9–22.7 macro-F1 points) far exceeds the seed-level dispersion reported above; only the selection cell at K = 33 shows a margin (3.0 points) comparable to the competing Random Forest's ±3.36 dispersion. All experiments

were run with Scikit-learn 1.6.1 on Python 3.12.13; package versions and random seeds are reported in Table IV, and the random seeds are recorded in the Appendix.

External validity. All experiments use a single dataset. This is a deliberate consequence of the isolation principle—the study revisits the framework of Li et al. on the dataset that framework used—but generalization of the specific magnitudes to other corpora is not claimed.

Construct validity. Two features of the dataset bound what the temporal protocols can be said to measure. First, TON-IoT was collected in a controlled testbed under a scripted attack schedule, so chronological order partly reflects the capture campaign rather than the natural evolution of traffic; the protocols therefore measure resistance to temporal leakage as instantiated by this capture, and the burst-capture limitation described in Section IV-F bounds per-class temporal evaluation accordingly. Second, chronological partitioning leaves some attack classes unevenly represented across the split, which affects macro-F1 directly; Table III quantifies class availability under Protocol B, ensuring that protocol effects are interpreted in the context of differences in class coverage. Tables V and VI report the best classifier per cell for readability, while all cross-protocol differences discussed in the text are computed with the classifier held fixed, and complete per-classifier results are provided in the Appendix so that no conclusion depends on the summarization choice.

The family-level claim is also bounded by the choice of selector: feature selection is represented by the single redundancy-oriented criterion inherited from Li et al., which scores inter-feature correlation without an explicit class-relevance term. Filter criteria that additionally score relevance—CFS [11], mutual-information ranking—or wrapper selectors that optimize detection-oriented objectives directly [21], albeit at substantially higher search cost, might narrow or reorder the FE–FS gap. The direction of any such shift is not obvious under temporal evaluation: a wrapper selector fits its selection to a validation split drawn from the training pool, so the selection step would itself be tuned against a leakage-prone estimate. The Wilcoxon result should therefore be read as extraction versus this selector rather than versus feature selection in general.

Conclusion validity. Findings are stated as protocol-dependent observations. The Wilcoxon and Kendall's τ analyses reported in Section IV-B are computed over configurations rather than over independent datasets, and should be read as descriptive of this experimental grid rather than as inference to a population of datasets; no correction for multiple comparisons is applied, consistent with that descriptive reading. We report point estimates with seed-level dispersion rather than bootstrap confidence intervals on individual test-set estimates; per-sample interval estimation is left to a future replication package.

VI. Conclusion and Future Work

This study asked whether the comparative feature-selection-versus-feature-extraction conclusions reported for IoT intrusion detection remain valid under leakage-resistant temporal evaluation. Holding the detection-performance component of the experimental framework of Li et al. [3] fixed and varying the partitioning rule as the sole intentionally manipulated experimental factor, we found the following. The published random-split findings were qualitatively reproduced, including near-ceiling performance at high dimensionality. Temporally consistent evaluation reduced measured performance substantially — by roughly 29 macro-F1 points in the binary task at $K = 9$ and 79 points in the multiclass task at $K = 47$, the largest observed degradation, for the fixed reference classifier — indicating that random evaluation produced markedly optimistic estimates within this framework. The five metrics did not degrade uniformly: accuracy could remain moderate while macro-averaged performance fell to very low absolute levels, and MCC exhibited the largest relative loss, declining by approximately 69% on average between the random and global temporal protocols. The family-level recommendation was directionally robust for the selector studied — extraction outperformed the inherited redundancy-oriented criterion under all three protocols, across the 50 matched configurations in each — whereas the magnitude of its advantage, configuration-level orderings, and the identity of the best-performing classifier were all protocol-dependent. Finally, a fixed-classifier decomposition across the two temporal protocols separated an estimate of within-class drift from the penalty associated with previously unseen attack classes: the drift component declined monotonically with dimensionality while the unseen-class component, although non-monotonic, dominated at the largest dimensionalities; a timestamp analysis further identified burst-captured attack campaigns as a structural limitation of per-class temporal evaluation on this corpus, implying that the reported drift components likely understate within-class drift on corpora with longer capture windows.

The principal implication is methodological. Comparative feature-engineering guidance for IoT intrusion detection should be validated under temporally consistent protocols before informing deployment decisions, and evaluation reports should state the partitioning rule explicitly and include macro-averaged metrics and MCC alongside accuracy.

Future work will extend the validation to a device-disjoint axis on corpora carrying device identity, following the split-governance protocol of Chikezie et al. [16]; that work also documents the feasibility limits of strict device-level holdout, which bound what such an extension can claim. A closely related extension will replicate the protocol contrast on an independent, timestamp-rich IoT corpus — reporting within-dataset reproduction and cross-dataset replication separately — and will construct controlled counterfactual splits, with matched class availability, matched temporal windows, and explicitly held-out attack campaigns, to decompose the observed degradation into temporal-proximity, class-shift, and campaign-shift components. A second direction is to incorporate open-set recognition mechanisms [20], recently instantiated for industrial IoT on Edge-IIoTset with incremental learning [22], so that previously unseen attack classes are detected rather than silently misclassified — noting that rejection improves operational reporting without, by itself, restoring macro-averaged performance on those classes. Finally, we will release per-sample predictions to enable bootstrap interval estimation, and examine whether deep representation-learning models exhibit the same protocol sensitivity as the shallow classifiers studied in this work.

ACKNOWLEDGMENTS AND DECLARATIONS

Funding. This research received no external funding.

COMPETING INTERESTS

The authors declare no competing interests.

DATA AVAILABILITY STATEMENT

The TON-IoT Network dataset used in this study is publicly available from its original source. All preprocessing procedures, evaluation protocols, experimental settings, and implementation details necessary to reproduce the reported experiments are described in this manuscript and the Appendix. No additional datasets or derived datasets are provided.

DECLARATION ON GENERATIVE AI

Generative AI tools were used solely to assist with language refinement, grammar improvement, and editorial polishing during the preparation of this manuscript. All scientific ideas, experimental design, implementation, analyses, interpretation of the results, and conclusions were developed and verified by the authors. The authors reviewed and edited all AI-assisted content and take full responsibility for the accuracy, originality, and integrity of the manuscript.

REFERENCES

[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," IEEE Communications Surveys & Tutorials, vol. 17, no. 4, pp. 2347–2376, 2015, doi: 10.1109/COMST.2015.2444095.

[2] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network Intrusion Detection for IoT Security Based on Learning Techniques," IEEE Communications Surveys & Tutorials, vol. 21, no. 3, pp. 2671–2701, 2019, doi: 10.1109/COMST.2019.2896380.

[3] J. Li, M. S. Othman, H. Chen, and L. M. Yusuf, "Optimizing IoT intrusion detection system: feature selection versus feature extraction in machine learning," J Big Data, vol. 11, no. 1, p. 36, Feb. 2024, doi: 10.1186/s40537-024-00892-y.

[4] V.-D. Ngo, T.-C. Vuong, T. Van Luong, and H. Tran, "Machine learning-based intrusion detection: feature selection versus feature extraction," Cluster Comput, vol. 27, no. 3, pp. 2365–2379, Jun. 2024, doi: 10.1007/s10586-023-04089-5.

[5] F. Pendlebury, F. Pierazzi, R. Jordaney, J. Kinder, and L. Cavallaro, "TESSERACT: Eliminating Experimental Bias in Malware Classification across Space and Time," presented at the 28th USENIX Security Symposium (USENIX Security 19), 2019, pp. 729–746. Accessed: Jul. 20, 2026. [Online]. Available: https://www.usenix.org/conference/usenixsecurity19/presentation/pendlebury

[6] D. Arp et al., "Dos and Don'ts of Machine Learning in Computer Security," presented at the 31st USENIX Security Symposium (USENIX Security 22), 2022, pp. 3971–3988. Accessed: Jul. 20, 2026. [Online]. Available: https://www.usenix.org/conference/usenixsecurity22/presentation/arp

[7] J. A. González-Ramos, E. J. Abril, P. Fernández, J. Prieto, and P. Chamoso, "Adversarial robustness evaluation of hybrid CNN-LSTM-transformer NIDS on evolving threats," Journal of Information Security and Applications, vol. 100, p. 104467, Jul. 2026, doi: 10.1016/j.jisa.2026.104467.

[8] T. Kim, "Beyond Random Splits: A Temporal Robustness Benchmark for Industrial Food Anomaly Detection," IEEE Access, vol. 14, pp. 72331–72345, 2026, doi: 10.1109/ACCESS.2026.3691935.

[9] P. Maniriho, A. N. Mahmood, and M. J. M. Chowdhury, "MeMalDet: A memory analysis-based malware detection framework using deep autoencoders and stacked ensemble under temporal evaluations," Computers & Security, vol. 142, p. 103864, Jul. 2024, doi: 10.1016/j.cose.2024.103864.

[10] I. Dissanayake, A. Welhenge, and H. D. Weerasinghe, "A Machine Learning Approach to Detect Denial of Sleep Attacks in Internet of Things (IoT)," IoT, vol. 6, no. 4, p. 71, Dec. 2025, doi: 10.3390/iot6040071.

[11] M. A. Hall, "Correlation-based feature selection for machine learning," Ph.D. dissertation, University of Waikato, 1999.

[12] B. Yan and G. Han, "Effective Feature Extraction via Stacked Sparse Autoencoder to Improve Intrusion Detection System," IEEE Access, vol. 6, pp. 41238–41248, 2018, doi: 10.1109/ACCESS.2018.2858277.

[13] B. A. Tama, M. Comuzzi, and K.-H. Rhee, "TSE-IDS: A Two-Stage Classifier Ensemble for Intelligent Anomaly-Based Intrusion Detection System," IEEE Access, vol. 7, pp. 94497–94507, 2019, doi: 10.1109/ACCESS.2019.2928048.

[14] C. Ambroise and G. J. McLachlan, "Selection bias in gene extraction on the basis of microarray gene-expression data," Proceedings of the National Academy of Sciences, vol. 99, no. 10, pp. 6562–6566, May 2002, doi: 10.1073/pnas.102102699.

[15] G. Engelen, V. Rimmer, and W. Joosen, "Troubleshooting an intrusion detection dataset: the CICIDS2017 case study," in Proc. IEEE Security and Privacy Workshops (SPW), 2021, pp. 7–12, doi: 10.1109/SPW53761.2021.00009.

[16] C. I. Chikezie, A. U. Usman, M. David, S. Zubair, H. O. Ohize, and J. Ojeniyi, "A Scientific Integrity Framework for Open-Set IoT Intrusion Detection with Device-Disjoint Splits," Future Internet, vol. 18, no. 6, p. 287, Jun. 2026, doi: 10.3390/fi18060287.

[17] A. E. Mahdaouy, I. A. Yahia, S. Oualil, and I. Berrada, "Deep Learning for Contextualized NetFlow-Based Network Intrusion Detection: Methods, Data, Evaluation and Deployment," Mar. 03, 2026, arXiv: arXiv:2602.05594. doi: 10.48550/arXiv.2602.05594.

[18] M. Z. Hossain, M. A. R. Khan, M. R. Islam, S. M. S. Islam, and T. Gedeon, "Assessing Generalisation Capability of Machine Learning Models for Intrusion Detection," May 06, 2026, arXiv: arXiv:2605.04407. doi: 10.48550/arXiv.2605.04407.

[19] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets," Sustainable Cities and Society, vol. 72, p. 102994, Sep. 2021, doi: 10.1016/j.scs.2021.102994.

[20] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, "Toward Open Set Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 7, pp. 1757–1772, Jul. 2013, doi: 10.1109/TPAMI.2012.256.

[21] Z.-H. Cheng, H. Shang, and C. Qian, "Detection-Rate-Emphasized Multi-objective Evolutionary Feature Selection for Network Intrusion Detection," Jun. 13, 2024, arXiv: arXiv:2406.09180. doi: 10.48550/arXiv.2406.09180.

[22] W. Lian and A. Guerra-Manzanares, "MI^{2}DAS: A Multi-Layer Intrusion Detection Framework with Incremental Learning for Securing Industrial IoT Networks," Feb. 27, 2026, arXiv: arXiv:2602.23846. doi: 10.48550/arXiv.2602.23846.

APPENDIX: COMPLETE PER-CLASSIFIER RESULTS

Tables IX–XXXVIII report, for every protocol, task, and dimensionality K, all five classifiers under both reduction methods with accuracy, precision, recall, macro-F1 (± standard deviation over three seeds for stochastic models), and MCC. All values derive from the single unified experimental run described in Section III-E.

It may be noted that, in several low-dimensionality selection configurations (K = 9 and K = 22), two or three classifiers report identical values across all five metrics—for example, the Decision Tree, MLP, and k-Nearest Neighbors under Protocol A at K = 9. These coincidences are genuine outputs of the replicated pipeline rather than transcription errors.

A. Protocol A (Random)

TABLE IX. PROTOCOL A (RANDOM) — BINARY, K = 9

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	80.84	79.64	77.83	78.55	0.5745
FS	RF	79.17	78.67	74.66	75.85	0.5318
FS	kNN	80.84	79.64	77.83	78.55	0.5745
FS	NB	78.95	82.76	71.89	73.52	0.5356
FS	MLP	80.84	79.64	77.83	78.55	0.5745
FE	DT	87.89	86.84	86.79	86.81	0.7363
FE	RF	87.13	85.85	86.55	86.15 ±0.31	0.7240
FE	kNN	87.81	86.73	86.75	86.74	0.7348
FE	NB	83.45	81.86	82.79	82.26	0.6464
FE	MLP	86.51	85.08	86.43	85.62	0.7150

TABLE X. PROTOCOL A (RANDOM) — BINARY, K = 22

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	81.40	80.16	78.67	79.28	0.5881
FS	RF	77.67	84.75	69.28	70.53 ±0.10	0.5177
FS	kNN	79.87	79.32	75.67	76.82	0.5487
FS	NB	78.29	84.87	70.20	71.65	0.5309
FS	MLP	81.40	80.16	78.67	79.28	0.5881
FE	DT	94.69	93.77	94.99	94.31	0.8875
FE	RF	89.87	88.64	89.87	89.16 ±0.57	0.7850
FE	kNN	86.68	86.16	84.47	85.19	0.7061
FE	NB	84.57	83.11	84.97	83.73	0.6805
FE	MLP	86.72	85.30	86.57	85.82	0.7185

TABLE XI. PROTOCOL A (RANDOM) — BINARY, K = 33

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	86.75	85.33	86.64	85.86	0.7195
FS	RF	86.00	84.55	86.32	85.19 ±0.01	0.7085
FS	kNN	81.76	81.05	78.35	79.33	0.5934
FS	NB	79.87	85.58	72.48	74.29	0.5656
FS	MLP	86.62	85.19	86.49	85.72	0.7167
FE	DT	98.25	97.90	98.32	98.10	0.9622
FE	RF	92.61	91.39	93.74	92.21 ±0.07	0.8511
FE	kNN	88.87	89.53	86.09	87.39	0.7554
FE	NB	83.47	83.69	79.66	81.00	0.6322
FE	MLP	86.72	85.30	86.57	85.83	0.7186

TABLE XII. PROTOCOL A (RANDOM) — BINARY, K = 47

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	94.91	94.58	94.32	94.44	0.8889
FS	RF	88.19	86.84	88.63	87.49 ±0.68	0.7544
FS	kNN	89.32	92.42	85.27	87.46	0.7736
FS	NB	81.15	84.06	75.11	76.97	0.5849
FS	MLP	86.66	85.24	86.53	85.77	0.7175
FE	DT	99.22	99.07	99.24	99.15 ±0.01	0.9831
FE	RF	95.21	94.26	95.68	94.87 ±0.01	0.8993
FE	kNN	99.08	98.90	99.11	99.00	0.9801
FE	NB	69.87	70.81	60.13	59.07	0.2904
FE	MLP	86.72	85.30	86.56	85.82	0.7185

TABLE XIII. PROTOCOL A (RANDOM) — BINARY, K = 77

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	99.32	99.18	99.35	99.26	0.9853
FS	RF	91.49	90.29	92.93	91.09 ±0.59	0.8318
FS	kNN	99.18	99.00	99.23	99.11	0.9823
FS	NB	59.66	71.22	67.86	59.29	0.3893
FS	MLP	86.73	85.31	86.58	85.83	0.7187
FE	DT	99.25	99.11	99.25	99.18	0.9836
FE	RF	95.42	94.41	96.12	95.12 ±0.19	0.9051
FE	kNN	99.13	99.01	99.10	99.06	0.9811
FE	NB	69.87	70.80	60.13	59.07	0.2903
FE	MLP	86.72	85.30	86.57	85.83	0.7186

TABLE XIV. PROTOCOL A (RANDOM) — MULTICLASS, K = 9

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	72.62	38.16	33.33	29.33	0.4548
FS	RF	71.38	31.27	27.70	24.09	0.4128
FS	kNN	68.65	38.99	32.76	25.74	0.4134
FS	NB	19.49	24.03	34.30	19.19	0.1965
FS	MLP	72.62	37.86	33.32	29.31 ±0.01	0.4547
FE	DT	78.20	74.61	48.54	46.15 ±0.02	0.5935
FE	RF	76.50	52.45	43.90	37.93 ±0.04	0.5591
FE	kNN	75.90	71.49	50.70	44.68	0.5850
FE	NB	52.35	30.65	50.30	32.56	0.4098
FE	MLP	76.80	51.07	44.95	39.53 ±0.41	0.5674

TABLE XV. PROTOCOL A (RANDOM) — MULTICLASS, K = 22

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	73.46	58.30	35.41	32.42	0.4767
FS	RF	70.21	31.69	24.58	20.98 ±2.63	0.3768
FS	kNN	69.24	58.23	34.15	28.29	0.4267
FS	NB	19.33	36.64	41.07	17.76	0.2063

FS	MLP	73.45	58.40	35.40	32.40	0.4766
FE	DT	87.68	79.88	72.50	71.91 ±0.10	0.7879
FE	RF	78.54	61.49	46.96	43.95 ±0.22	0.5963
FE	kNN	77.43	65.48	55.67	49.26	0.5966
FE	NB	56.00	37.60	63.67	41.61	0.4635
FE	MLP	77.36	66.99	45.85	41.27	0.5779

TABLE XVI. PROTOCOL A (RANDOM) — MULTICLASS, K = 33

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	77.39	67.09	46.04	41.60	0.5789
FS	RF	70.19	35.39	23.64	20.29 ±1.05	0.3749
FS	kNN	70.72	57.15	37.51	30.36	0.4648
FS	NB	32.54	31.79	45.52	24.09	0.2756
FS	MLP	77.22	65.91	45.71	41.09 ±0.01	0.5755
FE	DT	93.06	84.32	84.15	81.58 ±0.02	0.8813
FE	RF	82.52	61.31	57.79	56.07 ±0.42	0.6846
FE	kNN	79.10	73.98	52.69	52.08	0.6205
FE	NB	45.18	37.19	62.61	39.64	0.3925
FE	MLP	77.36	67.02	45.86	41.27 ±0.01	0.5779

TABLE XVII. PROTOCOL A (RANDOM) — MULTICLASS, K = 47

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	87.64	81.91	74.15	72.16 ±0.04	0.7865
FS	RF	72.06	51.03	27.77	25.54 ±2.00	0.4276
FS	kNN	82.33	79.89	56.10	56.42	0.6739
FS	NB	29.94	24.26	42.30	20.66	0.2476
FS	MLP	77.33	66.94	45.84	41.25 ±0.06	0.5775
FE	DT	98.08	93.03	93.34	93.14 ±0.07	0.9666
FE	RF	88.36	75.92	67.67	68.04 ±1.94	0.7897
FE	kNN	97.87	92.65	93.33	92.94	0.9629
FE	NB	35.06	39.30	56.67	32.27	0.3210
FE	MLP	77.37	68.76	45.88	41.32 ±0.04	0.5781

TABLE XVIII. PROTOCOL A (RANDOM) — MULTICLASS, K = 77

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	98.24	93.72	93.67	93.65 ±0.05	0.9694
FS	RF	78.05	56.89	41.26	41.18 ±4.71	0.5765
FS	kNN	98.09	93.37	93.60	93.44	0.9669
FS	NB	20.48	30.32	42.12	20.62	0.2023
FS	MLP	77.36	67.11	45.85	41.27	0.5779
FE	DT	98.14	93.16	93.19	93.13 ±0.14	0.9676
FE	RF	91.53	80.55	73.47	76.16 ±1.10	0.8480
FE	kNN	97.80	93.14	92.70	92.91	0.9614
FE	NB	26.44	30.59	47.90	20.18	0.2433
FE	MLP	77.36	67.07	45.85	41.27 ±0.01	0.5780

PROTOCOL B (GLOBAL TEMPORAL)

TABLE XIX. PROTOCOL B (GLOBAL TEMPORAL) — BINARY, K = 9

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	30.34	27.38	36.32	26.69	-0.3519
FS	RF	30.34	27.38	36.32	26.69	-0.3519
FS	kNN	39.91	69.95	50.01	28.54	0.0072
FS	NB	38.18	40.93	42.69	37.00	-0.1629
FS	MLP	30.34	27.38	36.32	26.69	-0.3519
FE	DT	58.22	58.99	59.33	58.04 ±0.02	0.1831
FE	RF	57.92	58.62	58.95	57.73 ±0.20	0.1756
FE	kNN	66.75	76.66	72.19	66.23	0.4865
FE	NB	58.91	59.18	59.56	58.59	0.1873
FE	MLP	57.90	58.68	59.00	57.72	0.1769

TABLE XX. PROTOCOL B (GLOBAL TEMPORAL) — BINARY, K = 22

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	54.05	55.91	55.98	54.04	0.1189
FS	RF	52.37	54.63	54.63	52.37 ±0.01	0.0926
FS	kNN	63.00	75.79	69.19	61.95	0.4449
FS	NB	62.23	73.18	52.88	44.03	0.1633
FS	MLP	54.05	55.91	55.98	54.04	0.1189
FE	DT	67.71	68.50	69.19	67.57 ±0.07	0.3769
FE	RF	58.05	58.83	59.16	57.88 ±0.02	0.1798
FE	kNN	62.27	68.31	66.59	62.02	0.3486
FE	NB	66.05	64.85	65.11	64.94	0.2996
FE	MLP	57.91	58.73	59.05	57.74 ±0.01	0.1778

TABLE XXI. PROTOCOL B (GLOBAL TEMPORAL) — BINARY, K = 33

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	57.70	58.58	58.88	57.55	0.1746
FS	RF	62.81	62.35	62.73	62.14 ±3.36	0.2508
FS	kNN	65.80	76.42	71.43	65.17	0.4758
FS	NB	49.82	43.13	44.60	42.88	-0.1218
FS	MLP	57.63	58.52	58.82	57.48 ±0.02	0.1733
FE	DT	83.77	83.56	84.96	83.56 ±0.11	0.6850
FE	RF	58.69	59.33	59.69	58.48 ±0.55	0.1902
FE	kNN	67.77	68.05	68.80	67.52	0.3685
FE	NB	66.06	64.85	65.11	64.94	0.2996
FE	MLP	57.84	58.70	59.01	57.68 ±0.01	0.1771

TABLE XXII. PROTOCOL B (GLOBAL TEMPORAL) — BINARY, K = 47

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	72.48	78.73	76.83	72.38 ±0.01	0.5553
FS	RF	59.57	59.84	60.16	59.17 ±3.92	0.2000
FS	kNN	61.34	67.07	65.56	61.12	0.3260

FS	NB	48.91	42.96	44.14	42.86	-0.1285
FS	MLP	57.67	58.56	58.87	57.52 ±0.02	0.1743
FE	DT	73.69	77.18	76.99	73.69 ±0.07	0.5417
FE	RF	69.45	68.97	69.72	68.95 ±0.19	0.3868
FE	kNN	93.33	92.89	93.29	93.08	0.8618
FE	NB	65.89	64.70	64.98	64.79	0.2968
FE	MLP	57.83	58.69	59.00	57.67 ±0.01	0.1769

TABLE XXIII. PROTOCOL B (GLOBAL TEMPORAL) — BINARY, K = 77

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	73.07	76.99	76.55	73.06 ±0.01	0.5354
FS	RF	66.53	65.58	66.01	65.65 ±0.39	0.3158
FS	kNN	92.94	92.46	92.95	92.68	0.8541
FS	NB	49.56	42.45	44.18	42.25	-0.1326
FS	MLP	57.68	58.57	58.87	57.53 ±0.02	0.1744
FE	DT	71.83	76.71	75.69	71.79 ±0.43	0.5239
FE	RF	69.69	69.18	69.93	69.17 ±0.76	0.3910
FE	kNN	94.76	94.57	94.50	94.53	0.8907
FE	NB	64.83	72.55	56.51	51.36	0.2423
FE	MLP	57.83	58.69	59.00	57.67 ±0.01	0.1769

TABLE XXIV. PROTOCOL B (GLOBAL TEMPORAL) — MULTICLASS, K = 9

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	26.32	4.56	9.42	6.15	-0.1435
FS	RF	26.32	4.56	9.42	6.15	-0.1435
FS	kNN	39.91	6.65	16.67	9.51	0.0029
FS	NB	7.23	17.86	5.94	7.57	0.0959
FS	MLP	26.32	4.56	9.42	6.15	-0.1435
FE	DT	29.53	11.69	9.02	9.25	0.1099
FE	RF	25.98	14.60	6.51	5.39 ±0.01	0.0521
FE	kNN	41.28	15.28	11.10	9.16	0.2191
FE	NB	32.44	13.22	12.81	12.96	0.1831
FE	MLP	29.23	14.26	8.86	9.36 ±0.01	0.1086

TABLE XXV. PROTOCOL B (GLOBAL TEMPORAL) — MULTICLASS, K = 22

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	29.63	14.22	11.18	11.09	0.0861
FS	RF	41.14	18.89	13.68	10.79 ±1.52	0.2076
FS	kNN	39.84	18.99	12.48	8.54	0.1803
FS	NB	8.99	15.81	5.30	6.54	0.1041
FS	MLP	29.63	13.69	10.77	10.68 ±0.58	0.0861
FE	DT	35.29	10.36	10.59	10.09 ±0.03	0.1822
FE	RF	30.07	14.69	8.40	8.00 ±0.41	0.0909
FE	kNN	38.02	14.93	10.80	9.82	0.1766
FE	NB	35.55	14.60	15.19	14.88	0.2253
FE	MLP	29.52	11.56	9.02	9.20 ±0.02	0.1067

TABLE XXVI. PROTOCOL B (GLOBAL TEMPORAL) — MULTICLASS, K = 33

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	29.58	11.64	9.05	9.25	0.1064
FS	RF	39.74	18.99	12.45	8.54 ±0.07	0.1791
FS	kNN	39.74	16.98	11.12	7.93	0.1912
FS	NB	11.35	11.80	5.99	5.07	0.0727
FS	MLP	29.47	12.88	9.99	10.19	0.1051
FE	DT	40.48	11.92	12.00	11.70 ±0.07	0.2776
FE	RF	30.23	14.76	8.26	7.66 ±0.38	0.0927
FE	kNN	33.38	11.95	10.07	10.20	0.1742
FE	NB	16.93	9.64	8.05	8.72	0.0613
FE	MLP	29.50	11.59	8.99	9.17	0.1054

TABLE XXVII. PROTOCOL B (GLOBAL TEMPORAL) — MULTICLASS, K = 47

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	43.32	13.03	12.65	11.43	0.2661
FS	RF	38.54	17.45	11.19	7.71 ±0.53	0.1586
FS	kNN	36.75	15.07	10.20	9.10	0.1583
FS	NB	13.63	12.29	7.56	5.84	0.0990
FS	MLP	29.51	12.45	9.66	9.86 ±0.48	0.1054
FE	DT	48.60	12.24	17.15	14.26 ±0.07	0.3348
FE	RF	39.76	14.24	13.85	13.68 ±0.48	0.2485
FE	kNN	41.64	13.50	12.41	12.83	0.3296
FE	NB	14.23	14.49	7.40	8.53	0.1465
FE	MLP	29.50	12.45	9.66	9.85 ±0.48	0.1053

TABLE XXVIII. PROTOCOL B (GLOBAL TEMPORAL) — MULTICLASS, K = 77

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	49.02	11.93	17.34	14.10 ±0.17	0.3317
FS	RF	36.13	9.39	10.06	7.27 ±0.25	0.1412
FS	kNN	41.50	13.32	12.32	12.70	0.3264
FS	NB	15.09	11.95	8.56	5.62	0.1042
FS	MLP	29.51	12.33	9.66	9.84 ±0.48	0.1056
FE	DT	49.40	12.08	17.61	14.31 ±0.04	0.3404
FE	RF	40.30	14.52	13.97	13.79 ±0.58	0.2529
FE	kNN	41.78	14.03	12.48	13.07	0.3331
FE	NB	13.52	14.41	7.28	8.22	0.1415
FE	MLP	29.50	12.39	9.66	9.84 ±0.48	0.1053

PROTOCOL C (PER-CLASS TEMPORAL)

TABLE XXIX. PROTOCOL C (PER-CLASS TEMPORAL) — BINARY, K = 9

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	63.12	62.03	62.99	61.84	0.2500
FS	RF	62.02	60.52	61.29	60.43 ±0.67	0.2179
FS	kNN	35.79	17.89	50.00	26.36	0.0000
FS	NB	63.12	62.03	62.99	61.84	0.2500
FS	MLP	63.12	62.03	62.99	61.84	0.2500
FE	DT	69.78	71.24	72.92	69.49	0.4413
FE	RF	69.93	71.51	73.18	69.67 ±0.26	0.4465
FE	kNN	47.27	69.70	58.87	44.03	0.2643
FE	NB	66.87	67.58	69.08	66.40	0.3662
FE	MLP	70.11	71.59	73.29	69.83	0.4485

TABLE XXX. PROTOCOL C (PER-CLASS TEMPORAL) — BINARY, K = 22

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	63.87	63.01	64.10	62.74	0.2709
FS	RF	61.01	59.17	59.78	59.15 ±0.90	0.1894
FS	kNN	43.23	69.21	55.78	38.29	0.2108
FS	NB	62.75	61.43	62.30	61.31	0.2371
FS	MLP	63.85	62.99	64.07	62.72 ±0.02	0.2703
FE	DT	82.18	82.55	85.39	81.88 ±0.03	0.6788
FE	RF	69.99	72.61	73.98	69.82 ±0.53	0.4657
FE	kNN	54.79	71.56	64.67	53.55	0.3557
FE	NB	70.22	72.25	73.86	70.02	0.4608
FE	MLP	70.09	71.55	73.25	69.80 ±0.33	0.4477

TABLE XXXI. PROTOCOL C (PER-CLASS TEMPORAL) — BINARY, K = 33

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	70.46	71.77	73.52	70.15	0.4526
FS	RF	69.99	71.85	73.47	69.76 ±0.04	0.4529
FS	kNN	44.17	69.40	56.51	39.65	0.2248
FS	NB	64.52	63.68	64.83	63.42	0.2848
FS	MLP	70.44	71.75	73.50	70.13 ±0.01	0.4521
FE	DT	87.02	86.32	89.38	86.63 ±0.05	0.7564
FE	RF	70.17	73.04	74.37	70.03 ±0.52	0.4739
FE	kNN	59.33	71.11	67.61	58.93	0.3856
FE	NB	69.43	71.91	73.35	69.28	0.4524
FE	MLP	69.71	71.32	72.97	69.45 ±0.10	0.4426

TABLE XXXII. PROTOCOL C (PER-CLASS TEMPORAL) — BINARY, K = 47

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	94.65	94.31	94.02	94.16	0.8832
FS	RF	70.26	72.79	74.27	70.11 ±0.08	0.4703
FS	kNN	69.12	76.56	75.83	69.11	0.5239

FS	NB	65.34	65.09	66.42	64.53	0.3148
FS	MLP	70.33	71.72	73.45	70.04 ±0.11	0.4513
FE	DT	95.84	94.86	96.54	95.56 ±0.01	0.9138
FE	RF	72.21	77.02	77.78	72.19 ±0.09	0.5479
FE	kNN	93.99	92.94	94.46	93.58	0.8739
FE	NB	81.40	86.03	74.78	76.80	0.5976
FE	MLP	70.01	71.52	73.20	69.73 ±0.26	0.4469

TABLE XXXIII. PROTOCOL C (PER-CLASS TEMPORAL) — BINARY, K = 77

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	96.01	95.12	96.52	95.73 ±0.09	0.9163
FS	RF	72.70	77.22	78.13	72.67 ±0.06	0.5534
FS	kNN	95.04	94.13	95.38	94.68	0.8950
FS	NB	43.65	61.16	54.97	40.14	0.1490
FS	MLP	69.81	71.39	73.05	69.54 ±0.25	0.4441
FE	DT	95.87	94.89	96.57	95.59 ±0.10	0.9144
FE	RF	72.78	77.66	78.42	72.76 ±0.07	0.5608
FE	kNN	95.79	94.89	96.25	95.49	0.9114
FE	NB	81.42	86.05	74.80	76.82	0.5981
FE	MLP	70.02	71.53	73.22	69.75 ±0.27	0.4472

TABLE XXXIV. PROTOCOL C (PER-CLASS TEMPORAL) — MULTICLASS, K = 9

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	52.84	36.28	33.56	27.77	0.2727
FS	RF	51.87	29.55	30.55	24.56	0.2381
FS	kNN	6.17	21.75	15.51	5.36	0.0357
FS	NB	14.94	21.41	40.98	17.39	0.1630
FS	MLP	52.84	36.28	33.55	27.77	0.2727
FE	DT	59.36	54.73	51.18	45.10	0.4335
FE	RF	57.95	56.81	45.75	38.44 ±0.39	0.4020
FE	kNN	23.15	45.92	29.74	20.03	0.1875
FE	NB	51.76	30.15	46.41	31.88	0.4004
FE	MLP	57.88	44.49	46.68	39.71 ±0.01	0.4181

TABLE XXXV. PROTOCOL C (PER-CLASS TEMPORAL) — MULTICLASS, K = 22

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	53.49	53.98	35.19	30.46	0.2907
FS	RF	55.09	19.32	21.68	16.47 ±3.36	0.1838
FS	kNN	15.47	52.89	20.82	10.35	0.1310
FS	NB	23.62	41.89	48.12	25.25	0.2473
FS	MLP	53.48	53.56	35.19	30.47 ±0.01	0.2907
FE	DT	74.68	63.82	72.05	62.84 ±0.03	0.6394
FE	RF	61.21	57.03	53.41	47.55 ±0.96	0.4505
FE	kNN	33.02	60.35	36.42	27.45	0.2632
FE	NB	56.76	48.18	58.98	44.37	0.4628
FE	MLP	58.44	59.43	48.32	41.88 ±0.11	0.4283

TABLE XXXVI. PROTOCOL C (PER-CLASS TEMPORAL) — MULTICLASS, K = 33

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	59.01	66.56	48.42	42.15	0.4332
FS	RF	51.80	30.28	30.24	24.47 ±0.09	0.2304
FS	kNN	16.60	47.58	21.37	12.19	0.1167
FS	NB	24.02	37.22	44.19	24.61	0.2270
FS	MLP	59.00	62.67	48.31	41.97 ±0.04	0.4329
FE	DT	80.51	68.30	76.47	68.08 ±0.01	0.7100
FE	RF	61.76	58.83	54.51	49.46 ±0.88	0.4606
FE	kNN	40.42	54.91	41.70	32.47	0.3125
FE	NB	56.62	46.41	56.35	43.13	0.4638
FE	MLP	59.02	63.55	48.25	41.90 ±0.10	0.4327

TABLE XXXVII. PROTOCOL C (PER-CLASS TEMPORAL) — MULTICLASS, K = 47

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	86.38	72.77	69.62	67.30	0.7636
FS	RF	54.29	47.10	35.82	31.18 ±1.01	0.2708
FS	kNN	51.97	47.53	50.96	39.40	0.4201
FS	NB	23.61	35.10	43.06	22.63	0.2267
FS	MLP	58.86	62.17	48.30	41.92 ±0.07	0.4317
FE	DT	93.54	84.57	88.27	85.85 ±0.08	0.8929
FE	RF	66.01	62.61	68.15	62.35 ±1.08	0.5559
FE	kNN	89.74	78.12	80.93	78.59	0.8285
FE	NB	29.83	43.27	51.13	36.72	0.3250
FE	MLP	58.54	60.50	48.92	42.32 ±0.85	0.4292

TABLE XXXVIII. PROTOCOL C (PER-CLASS TEMPORAL) — MULTICLASS, K = 77

Method	Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	MCC
FS	DT	94.65	87.52	89.81	88.06 ±0.08	0.9112
FS	RF	60.13	57.20	49.16	43.01 ±1.53	0.3800
FS	kNN	90.72	80.45	81.74	79.92	0.8436
FS	NB	20.55	35.09	42.25	21.41	0.2033
FS	MLP	58.66	60.74	48.36	41.96 ±0.12	0.4300
FE	DT	93.86	85.20	89.37	86.70 ±0.08	0.8988
FE	RF	66.92	64.09	69.60	63.70 ±1.26	0.5646
FE	kNN	92.29	82.37	85.04	83.05	0.8702
FE	NB	29.12	49.28	53.93	37.05	0.3207
FE	MLP	58.46	59.34	48.36	41.91 ±0.12	0.4285