Statistical Ranking: A Voting-Based Ensemble Approach to Feature Selection in Android Malware Detection

Anna Luiza Gomes da Silva[1], Lucas Ferreira Areias de Oliveira[1], Angelo Diniz[1], Diego Kreutz[1], Dionatan R. Schmidt[1], Rodrigo Mansilha[1], Kayuã Oleques Paim[2]

[1]AI Horizon Labs
Programa de Pós-Graduação em Engenharia de Software (PPGES)
Universidade Federal do Pampa (UNIPAMPA)

[2]Universidade Federal do Rio Grande do Sul (UFRGS)

{annaluiza,lucasareias , angelonogueira}.aluno@unipampa.edu.br
{diegokreutz, dionatanschmidt, rodrigomansilha}@unipampa.edu.br
kayua.paim@inf.ufrgs.br

Abstract. *High-dimensional Android malware datasets inflate training time and damage model generalization. We propose Statistical Ranking, an ensemble feature selection framework that enforces multi-perspective consensus: a feature is retained only when ranked in the top-k by at least two of three independent criteria (χ^2, Mutual Information, and Random Forest importance), under a per-criterion budget that scales with the original dimensionality. Across 11 public datasets under stratified 5-fold cross-validation, the method removes 89.8%–99.6% of the feature space with a median recall drop of 0.04, retaining recall ≥ 0.85 in 9 of 11 environments. An ablation over ranking criteria and voting thresholds shows that the ≥ 2-of-3 consensus maximizes the reduction, recall trade-off, and that the contribution of a third ranker depends on the redundancy between the marginal criteria, which varies with feature modality. On MH100K, 24,833 features are reduced to 93 (99.6%) at a cost of 6.5% in F1-score, and the one-off cost of selection is recovered after roughly eight retraining cycles, yielding a 27% reduction in total cost over a ten-update horizon.*

1. Introduction

Feature spaces in publicly available Android malware datasets routinely exceed 6,000 dimensions, yet our experiments demonstrate that 89.8–99.6% of these features can be removed without a measurable loss of discriminative power, while their presence inflates training time up to fivefold and undermines generalization across heterogeneous datasets. Detecting these threats is a complex task because Android applications utilize a vast array of permissions, components, and API calls, resulting in high-dimensional datasets [Rocha et al. 2026a]. In this context, feature selection (FS) has emerged as a critical step in the Machine Learning (ML) pipeline. By identifying the most informative subset of attributes, FS reduces data dimensionality, enhances model accuracy, and significantly decreases the computational costs associated with training [Venkatesh and Anuradha 2019].

Feature selection techniques are traditionally categorized into filters, wrappers, and embedded methods [da Silva and Silveira 2022]. Filters rely on statistical properties of the data, offering high computational efficiency and independence from classifiers, whereas wrappers and embedded methods interact closely with learning algorithms

to achieve higher predictive performance, though often at a higher computational cost. However, while traditional filter methods offer low complexity, they typically fail to capture complex multi-feature interactions. This structural limitation has recently motivated the exploration of hybrid and ensemble frameworks across various high-dimensional domains [Pudjihartono et al. 2022], establishing a precedent for combining diverse selection logics to mitigate individual algorithmic biases.

Despite this availability, no single selection criterion generalizes reliably across Android malware datasets: filter methods (e.g., Chi-squared) are insensitive to dependencies that are not marginal; wrappers sacrifice computational tractability; and the majority of proposals are validated on a single dataset (Table 1), which makes their claimed generalization empirically unverifiable [Rocha et al. 2026b, Rocha et al. 2026a]. These datasets are further characterized by severe class asymmetry, where benign samples significantly outnumber malicious ones, and performance variations across imbalanced data underscore the need for selection criteria that account for such asymmetries so that discriminative features are not overlooked [Rocha et al. 2026a].

Because each selection logic captures a different facet of feature relevance, statistical dependence, information-theoretic content, and model-based importance, relying on any single criterion risks discarding features that other criteria would rank as discriminative. Statistical Ranking builds on this premise by requiring cross-criterion consensus: a feature is retained only when ranked in the top-k by at least two of three independent rankers, under a per-criterion budget k that scales with the original dimensionality rather than being fixed. The method is implemented as a multi-stage automated pipeline: it first performs redundancy reduction through variance and correlation filtering, and then applies a voting mechanism that aggregates the top-ranked features from three distinct perspectives. By combining these strengths, we aim to leverage a more stable and "limitless" evaluation environment as advocated by the MH-FSF framework [Rocha et al. 2026a], producing feature sets that are less sensitive to the specificities of individual datasets. As our ablation shows, the extent to which the three perspectives are genuinely complementary is itself dataset-dependent, and quantifying that dependence is one of the contributions of this work.

This work makes four primary contributions:

- **C1, Multi-perspective consensus.** A three-ranker voting pipeline requiring agreement among ≥ 2 criteria under an adaptive per-criterion budget, eliminating single-logic bias without a labeled validation set for tuning (Section 4).
- **C2, Large-scale validation.** Evaluation across 11 public datasets under stratified 5-fold cross-validation: 89.8–99.6% reduction while preserving recall ≥ 0.85 in 9 of 11, median recall drop 0.04 (Section 6).
- **C3, Component-wise ablation.** Isolation of each ranking criterion and voting threshold $\theta \in \{1, 2, 3\}$ with a stability analysis, showing the $\geq$ 2-of-3 rule best trades reduction against recall and that a third ranker helps only where the marginal criteria disagree (Section 6.4).
- **C4, Cost accounting and generalization.** A computational budget separating one-off selection from recurring training cost, showing amortization after ≈ 8 retraining cycles, plus evidence that the reduced subsets generalize across four classifier families (Sections 6.2 and 6.3).

2. Background

This section outlines the definitions of the three feature selection criteria employed in our ensemble voting scheme. The Chi-Square statistic evaluates the statistical dependence between discrete features and the target class, retaining features whose association exceeds a critical significance threshold. Mutual Information (MI) quantifies the shared information between a feature and the class, capturing both linear and non-linear dependencies through probability distributions, thus providing a robust, model-agnostic evaluation of feature-class correlations [Tisoc and Beltran 2022]. Random Forest (RF) [Iranzad and Liu 2025] offers intrinsic multivariate importance scores via Mean Decrease Impurity, which measures contributions to homogeneous splits, and Mean Decrease Accuracy, which assesses feature reliance by permuting values and observing the drop in Out-of-Bag accuracy.

Relying on a single logic leads to univariate bias or sensitivity to correlations, so integrating the three through ensemble voting is intended to leverage their distinct strengths [Iranzad and Liu 2025]: χ^2 is univariate and marginal, MI is univariate and distribution-free, and RF is multivariate and model-driven, so only the last accounts for feature interactions. Section 6.4 measures how far this complementarity holds across feature modalities.

3. Related Work

In Table 1, we present the main related works in the context of feature selection methods for Android malware detection, listing the method or framework and its category, that is, whether it is a filter, wrapper, or embedded method.

Table 1. Positioning of this study in relation to recent works on feature selection for Android malware detection.

Method	Category	Domain	Feature selection logic	# DS	Metrics
Chi2-MI [Dey et al. 2022]	Hybrid (filter)	Medical	Intersection of Chi-square and Mutual Information rankings	1	Accuracy, F1
DroidRL [Wu et al. 2023]	Wrapper	Android malware	Deep RL (DDQN) + RNN sequential selection	1	Accuracy
chi2-PSOGWO [Abdo et al. 2024]	Hybrid (filter + wrapper)	General ML	Chi-square pre-filtering + swarm intelligence (PSO-GWO)	3	Accuracy, F1
CorrNetDroid [Sharma and Arora 2025]	Filter	Android malware	Feature-class and feature-feature correlation (crRelevance, NMRS)	1	Accuracy, F1
SigAPI AutoCraft [Rocha et al. 2026b]	Wrapper (auto.)	Android malware	Automated optimal feature count + SMOTE/RUS balancing	10	MCC, Recall
MH-FSF [Rocha et al. 2026a]	Eval. framework	Android malware	Benchmark of 17 FS methods across balanced and imbalanced data	10	Recall, F1, MCC
EnFeSTDroid [Jain et al. 2026]	Ensemble (filter)	Android malware	Rank-sum aggregation of six permission rankers (IG, ETC, χ^2, MTF, IDF, MTF–IDF)	1	Accuracy, F1
This work	**Hybrid (ensemble)**	**Android malware**	**θ-of-n consensus vote: Chi-squared, Mutual Information, Random Forest importance**	**11**	**Recall, F1, MCC**

Wrapper-based methods optimize feature selection through iterative interaction with learning algorithms. Wu et al. [Wu et al. 2023] introduced DroidRL, utilizing Deep Reinforcement Learning (DDQN) and RNN to reduce the feature space from 1,083 to 24

attributes while maintaining 95.6% accuracy. Similarly, Rocha et al. [Rocha et al. 2026b] developed SigAPI AutoCraft, which automates optimal feature count determination and stabilizes performance through proper data balancing (SMOTE and RUS), achieving robust MCC scores.

Filter methods offer greater computational efficiency by relying on statistical properties independently of classifiers. Sharma and Arora [Sharma and Arora 2025] proposed CorrNetDroid, employing feature-class and feature-feature correlations (crRelevance, NMRS) to achieve high detection accuracy with minimal network traffic features. The MH-FSF framework [Rocha et al. 2026a] provides a modular platform for evaluating 17 selection methods across multiple datasets, revealing that performance varies significantly between balanced and imbalanced data. Hybrid approaches overcome single-method limitations by combining complementary selection logics: Dey et al. [Dey et al. 2022] demonstrated that intersecting Chi-square and Mutual Information rankings identifies redundant features more effectively than isolated methods in medical diagnosis, and Abdo et al. [Abdo et al. 2024] showed that Chi-square pre-filtering with swarm intelligence (PSO-GWO) improves execution time and accuracy.

The closest work to ours is EnFeSTDroid [Jain et al. 2026], which aggregates six permission rankers through a rank-sum of their orderings, using Friedman and post-hoc Nemenyi tests to establish that the six differ, and reports 96.27% accuracy with 50 permissions. Two aspects of that result motivate our design. First, four of the six rankers are variants of term-frequency weighting and produce nearly identical orderings, their reported top-ten lists for MTF, IDF and MTF–IDF coincide, so the number of votes exceeds the number of perspectives. Second, the ensemble improves accuracy by 0.02 percentage points over the best single ranker (0.9627 against 0.9625 for the Extra Tree Classifier), a difference reported without a significance test. Establishing that a set of rankers is statistically *different* is not the same as establishing that combining them is *useful*, and the question of when consensus pays off is precisely what our ablation (Section 6.4) sets out to measure.

Positioning. Ensemble feature selection is not itself new; what varies is *how* rankings are combined and *how much* evidence is offered that the combination matters. Three differences separate Statistical Ranking from the closest prior work. First, the aggregation rule is a hard consensus constraint on membership, not a score fusion or rank aggregation, so it yields a subset directly, without an external cut-off or a labelled validation set for tuning. Second, the members are chosen for orthogonality of inductive bias rather than quantity: aggregating many univariate filters adds votes but not perspectives, whereas our three span marginal dependence, distribution-free information content, and multivariate importance. Third, the evidence base is broader: eleven datasets across three modalities, selection performed inside every fold, and a component-wise ablation with a stability analysis. The contribution therefore lies partly in the rule and substantially in the empirical characterisation, which is what lets us state where the method fails as well as where it works.

4. The Statistical Ranking Method

This section describes the proposed method. Data flows from the original dataset through an initial cleaning stage, which applies low-variance and correlation filters to prune imme-

diate redundancy; the remaining features are then subjected to a parallel ensemble voting mechanism that aggregates statistical, information-theoretic, and model-driven perspectives before delivering the final compact subset, as illustrated in Figure 1.

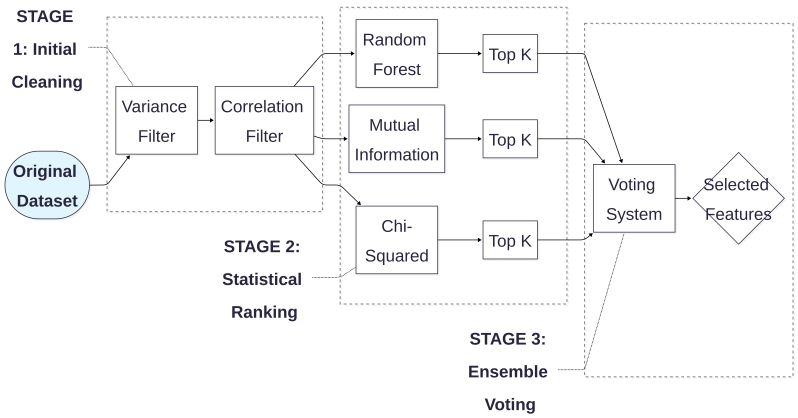

Figure 1. Statistical Ranking framework.

4.1. Pipeline

First, a low-variance filtering step is performed using a variance threshold of 0.01. Features with very small variance provide limited discriminative power because their values remain nearly constant across samples. This is followed by a correlation-based filtering step to remove redundant attributes: according to [Theng and Bhoyar 2024], a truly relevant feature set must aim to remove irrelevant and redundant features, making this a crucial step in machine learning tasks. When the absolute Pearson correlation between two features exceeds 0.95, one is discarded to reduce multicollinearity and avoid retaining features that convey nearly identical information.

After the initial filtering, each surviving feature is scored independently by the three criteria formalised in Section 2, and the top-k features of each ranking are collected. The rankers operate on the same cleaned matrix and are mutually blind: no ranker observes the output of another, so the three rankings constitute independent evidence about relevance. This follows the logic of [Abdo et al. 2024], who demonstrated that χ^2 serves as an efficient initial filter to prune the search space, and of [Alalhareth and Hong 2023], who argued that MI-based techniques are valuable in domains with sparse data or evolving attack patterns.

Adaptive budget. The number of top-ranked features collected from each criterion is not a fixed constant but a function of the original dimensionality: $k = \min(100, \max(10, \lfloor p/10 \rfloor))$, where p is the number of features in the raw dataset. A fixed budget is a poor fit for a benchmark spanning two orders of magnitude in dimensionality: on AndroCrawl ($p = 141$) a top-50 list would already cover 35% of the feature space, leaving the consensus little to discard, whereas on MH100K ($p = 24{,}833$) the same budget would force each individual criterion to be far more selective than the ensemble requires. Anchoring k at one tenth of the original dimensionality, with a floor of 10 and a ceiling of 100, yields $k \in [14, 28]$ for the permission-based datasets and $k = 100$ for the API-call-graph and large-scale ones, and introduces no dataset-specific tuning.

Consensus vote. A feature is retained if and only if it appears in the top-k of at least θ of the three rankings. We adopt $\theta = 2$ as the default; this ensemble strategy reduces dependence on any single ranking criterion, mitigating the overfitting and instability common in high-dimensional Android datasets, and is supported by [Dey et al. 2022], whose hybrid intersection model achieved higher diagnostic accuracy by merging complementary ranking criteria. Section 6.4 replaces this choice with measurements over the ten benchmark datasets and characterises the trade-off frontier that θ traverses. The entire selection process is fully automated and applied consistently across all datasets.

Two elements of the pipeline are domain-specific and one is not. The variance threshold of 0.01 is calibrated for binary presence vectors, where it excludes permissions declared by fewer than roughly 1% of applications, and would need recalibration on continuous features. The adaptive budget $k = \lfloor p/10 \rfloor$ is tuned to the dimensionality range of Android datasets (141 to 24,833 features). The voting rule itself is domain-agnostic, and we present the Android evaluation as the setting in which the method is validated rather than as the only setting in which it applies. What the domain contributes to the result is the feature modality: the finding of Section 6.4, that the value of a third ranker depends on the redundancy between the marginal criteria, is visible here precisely because Android datasets span both binary permission vectors and continuous graph centralities.

4.2. Threat Model

We assume a passive adversary with access to the same public datasets and feature-engineering techniques we use, but no knowledge of the selected subset, operating on static features (permissions, API calls) from APK manifests and bytecode, whose goal is to evade detection without runtime analysis and without altering core malicious functionality.

Consensus plausibly raises the cost of feature manipulation, because an adversary who knows one selection logic cannot infer from it which features survive the vote of the other two. This is an argument from the design, not a measured result, and it has clear limits. First, the criteria are not independent in the relevant sense: Section 6.4 reports a mean χ^2–MI overlap of 0.714 on permission-based datasets, so two of the three votes are largely redundant there and the consensus adds little resistance. Second, the pipeline is deterministic given the training data and the retained subsets are stable across folds (Kuncheva index 0.885, Table 7); an adversary with the same public datasets can reproduce the subset exactly, so Kerckhoffs' principle applies and the method offers no security through obscurity. What consensus provides is therefore narrow: it makes the subset harder to *guess* from any single logic, since a feature must be endorsed by criteria with different inductive biases. Robustness against an informed adversary would require randomising the selection, which we have not evaluated, and measuring evasion cost under partial knowledge of the subset is the security question this work leaves open.

5. Experimental Setup

This section describes the datasets, the execution environment, the classifier configuration and the two evaluation protocols used throughout the paper.

5.1. Datasets

Table 2 presents the specifications of the malware datasets used in this study, sourced from the project public repository[1].

The benchmark spans three orders of magnitude in dimensionality (141 to 24,833 features) and three feature modalities: permission vectors, API-call-graph centralities, and hybrid sets. This heterogeneity is deliberate, since it is what allows the analysis of Section 6.4 to distinguish properties of the method from properties of a particular feature representation.

Table 2. Summary of the malware datasets used in this study.

Name	Samples	Features	Benigns	Malwares
Adroit	11,476	166	8,058	3,418
Android Permissions	26,864	151	9,077	17,787
Androcrawl	96,744	141	86,574	10,170
DefenseDroid PRS	11,975	2,877	5,975	6,000
Drebin215	15,031	215	9,476	5,555
KronoDroid Real Device	78,137	286	36,755	41,382
KronoDroid Emulator	63,991	276	35,246	28,745
DefenseDroid API Calls Closeness	10,476	4,274	5,222	5,254
DefenseDroid API Calls Degree	10,476	6,002	5,222	5,254
DefenseDroid API Calls Katz	10,476	6,002	5,222	5,254
MH100K	101,934	24,833	92,134	9,800

5.2. Execution Environment and Classifier Configuration

Because part of our evaluation concerns computational cost (Section 6.2), we specify the environment in which all measurements were taken: an Intel Core i7-1165G7 with 8 logical cores and 23.2 GB of RAM, running Ubuntu 24.04.4 and Python 3.12.3, with scikit-learn 1.8.0, NumPy 2.4.4, SciPy 1.17.1 and pandas 3.0.2. All timings were obtained on this single machine, without GPU acceleration, and should be read as relative rather than absolute.

The Random Forest classifier is configured with 100 estimators, Gini impurity, `max_depth` unbounded, `min_samples_split` of 2, `min_samples_leaf` of 1, `max_features` set to `sqrt`, bootstrap sampling, `random_state` 42 and `n_jobs` of -1. We state `max_features` explicitly because its default changed in scikit-learn 1.1. The same hyperparameters are used both for feature importance ranking (Stage 2 of the pipeline) and for final classification; the consequences of this dual role are examined in Sections 6.4 and 7.

5.3. Evaluation Protocols

Protocol P1: direct classification. The reduced feature set is compared against the full-dimensional one on real data with the Random Forest of Section 5.2, under stratified 5-fold cross-validation across the 11 datasets. Selection runs independently inside each training fold, so the selector never sees the test partition; this removes the optimistic bias of cross-validating an already-reduced dataset. P1 produces Table 3, extended to other inductive families in Section 6.3.

[1]Available at: `https://https://github.com/AILabs4All/datasets-mh30plus/tree/main`

Protocol P2: Train-on-Synthetic / Test-on-Real (TSTR). The reduced sets are evaluated within MalDataGen [Paim et al. 2025], which trains autoencoder and variational generators on the selected features and tests classifiers trained on the synthetic data against real data. TSTR probes whether a reduced representation preserves the data distribution, not merely its discriminative surface. Results are averaged over three classifiers (Decision Tree, Random Forest, k-NN) and both generators, and compared against Lasso, RFE and SemiDroid (Table 4, Figure 2). Recall is not comparable across protocols: on MH100K the reduced model reaches 0.7502 under P1 and 0.88 under P2, because they measure different things.

Method selection. Per dataset under P2 (Figure 2), methods are ranked lexicographically by recall, then reduction, then F1 preservation; recall leads because a missed malicious sample costs more than a false alarm.

On statistical testing. Significance in P1 uses the paired Wilcoxon signed-rank test over the five folds, whose exact null at $n = 5$ admits a minimum two-sided p-value of $2/2^5 = 0.0625 > \alpha = 0.05$: it cannot reject the null at conventional levels regardless of the differences. We report the values for completeness only; the recurring 0.0625 is the floor of the test, not a near-significant trend. The statistical claims rest instead on the Friedman test across datasets (Section 6.4), where the number of blocks suffices.

6. Results and Discussion

The evaluation answers four questions in sequence. Section 6 asks whether the reduced subsets preserve detection on real data (protocol P1, all eleven datasets). Section 6.2 asks what the pipeline itself costs and after how many retraining cycles it pays for itself. Section 6.3 asks whether the retained features are specific to the tree ensemble that ranks them. Section 6.4 asks which components of the pipeline actually contribute, and under which conditions. The comparison against Lasso, RFE and SemiDroid runs under protocol P2 (TSTR) and is reported in Section 6, whose values are not comparable with those of P1 for the reason given in Section 5.3.

Table 3 reports recall and F1-score for the original and reduced feature sets. As explained in Section 5.3, the p-values are bounded below by the design of the test and support no inference. The direction of the differences is nevertheless informative: nine of the ten benchmark datasets favour the original model, with a median recall drop of 0.04. The single exception is `android_permissions`, where reduction *improves* recall from 0.8962 to 0.9559. A model audit attributes this to overfitting mitigation: the train-test gap narrows from 0.0512 to 0.0116, cross-fold variance remains stable at ≈ 0.0081, and Gini importance concentrates on sensitive permissions instead of being diluted across the original 151 features. MH100K, whose gap is an order of magnitude larger, is discussed in Section 6.2.

6.1. Comparison Against Alternative Reduction Methods

Under Protocol P2, we compare Statistical Ranking against Lasso, RFE and SemiDroid using the multi-criteria prioritization of Section 5.3. Table 4 presents the aggregate performance of each method across the 11 datasets, separately for the autoencoder and variational generators. We define the *Trade-off Score* as Recall $\times$ Reduction$/100$, a metric that balances detection against dimensionality reduction and therefore favours methods that maintain high recall under aggressive pruning.

Table 3. Performance comparison under Protocol P1. The p-values are bounded below by 0.0625 at $n = 5$ folds and are reported for completeness only (Section 5.3).

Dataset	Orig. Recall	Δ Recall (Red−Orig)	Red. Recall	Orig. F1-Score	p-value (recall)	Red. F1-Score
DD API Katz	0.9233	-0.0116	0.9117	0.9355	0.0625	0.9253
AndroCrawl	0.9163	-0.0462	0.8701	0.9331	0.0625	0.9138
KD Real Device	0.9731	-0.0102	0.9629	0.9771	0.0625	0.9642
DD PRS	0.9052	-0.0155	0.8897	0.9246	0.1250	0.9169
DD API Closeness	0.9252	-0.0143	0.9109	0.9386	0.1250	0.9244
Adroit	0.7876	-0.0506	0.7370	0.8437	0.0625	0.8160
KD Emulator	0.9669	-0.0406	0.9263	0.9691	0.0625	0.9385
Android Permissions	0.8962	+0.0597	0.9559	0.7813	0.0625	0.7917
DREBIN-215	0.9759	-0.0479	0.9280	0.9843	0.0625	0.9364
DD API Degree	0.9233	-0.0116	0.9117	0.9355	0.0625	0.9253
Large-Scale Validation						
MH100K	0.8867	-0.1365	0.7502	0.8727	—	0.8159

The product form is deliberate but not neutral. It treats a one-point loss of recall as interchangeable with a one-point gain in reduction, which is defensible only where both costs are comparable, that is, in high-throughput triage where the alternative to an aggressive reduction is not analysing the sample at all. Where a missed sample is expensive, the correct operating point is not the maximum of the product: a deployment that weights recall at w times reduction should compare methods by $\text{Recall}^w \times \text{Reduction}$, and for $w \geq 3$ the ordering in Table 6 changes, with $\theta = 1$ overtaking $\theta = 2$. We report the unweighted product because it makes the comparison against prior work reproducible without assuming a cost model, and we report recall and reduction separately in every table so that a reader with a different cost model can re-rank the configurations.

Table 4. Aggregate comparison of feature selection methods across the 11 datasets, under protocol P2 (TSTR). Trade-off = Recall × Reduction/100.

Method	Autoencoder				Variational			
	Recall	F1	Red (%)	Trade-off	Recall	F1	Red (%)	Trade-off
Lasso	0.707	0.707	43.3	0.306	0.715	0.730	43.3	0.310
RFE	0.811	0.795	78.4	0.636	0.760	0.729	78.4	0.596
SemiDroid	0.780	0.775	78.1	0.609	0.745	0.716	78.1	0.582
Statistical (Ours)	**0.809**	**0.805**	**94.3**	**0.763**	**0.778**	**0.746**	**94.3**	**0.734**

Statistical Ranking achieves the highest trade-off score under both generators (0.763 and 0.734), outperforming RFE by 20.0–23.0% and SemiDroid by 25.3–25.9%, and is stable across generator types (recall difference of 0.031). It was not, however, optimal everywhere: it was selected as best on 4 of 11 datasets (`android_permissions`, `kronodroid_emulator`, `kronodroid_real_device`, MH100K), combining recall above 0.85 with 90–98% reduction; RFE was optimal where recall could be raised sharply (e.g. `defensedroid_apicalls_closeness`, 0.45 to 0.85 at 95% reduction); SemiDroid led on API-call-dominated datasets; and Lasso was never selected, its 43.3% reduction being too limited. This heterogeneity is not explained by the permission/API dichotomy alone: on MH100K, 83.9% of retained features and 87.8% of Gini importance are API calls with only three permissions surviving, so what varies with modality is the redundancy among the marginal criteria (Section 6.4), not the applicability

of the method.

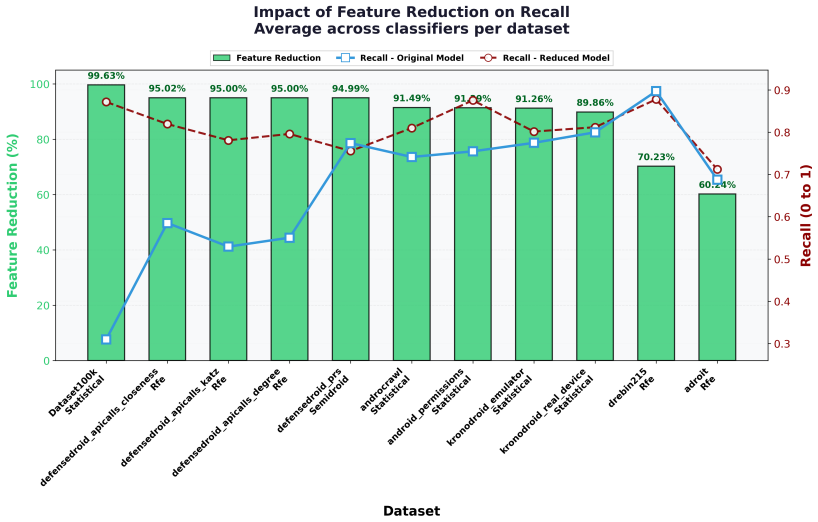

Figure 2. Feature reduction and recall preservation across datasets using the best-performing method per dataset. Green bars: feature reduction percentage. Lines: original (blue) vs. reduced (red) recall.

Figure 2 pairs each dataset with its best method: reduction ranges from roughly 60% to above 95%, and the reduced recall usually matches or exceeds the original, confirming that a recall-first selection criterion preserves detection.

6.2. Computational Cost of the Framework

An efficiency claim is only meaningful if it accounts for the cost of the selection itself. We measured the full budget across the ten benchmark datasets, separating the one-off cost of the selection pipeline from the recurring cost of training a detector.

The cleaning filters account for only 2.4% of the selection time (0.33 s), the ranking stage for the remaining 97.6% (13.28 s); the asymmetry is driven by sample count, not dimensionality, since the mutual-information estimator and the Random Forest ranker scale with samples (AndroCrawl's 96,744 samples cost 16.7 s to rank, against 4.7 s for the twenty-times-wider but eight-times-smaller DefenseDroid PRS). Filter ordering is also a memory constraint: correlation filtering needs a dense $p^2 \times 8$-byte matrix (4.93 GB for MH100K), so applying the variance filter first ($v1$) shrinks p beforehand and cut MH100K's execution from 1,957 s to 400 s (a 79.6% reduction) with no measurable loss in recall. We adopt $v1$ throughout.

One-off cost against recurring savings. Training on the reduced subsets is on average $8.1\times$ faster (0.28 s against 2.25 s). Since selection is paid once per dataset and training recurs on every model update, the break-even point is reached after a median of 7.7 retraining cycles, ranging from 1.4 on DefenseDroid PRS to 72.6 on DREBIN-215, the dataset with the smallest reduction and hence the smallest saving per update. Aggregated over the ten datasets, a single training run costs $6.2\times$ more with the pipeline than without it, whereas ten updates cost 27.1% less and fifty updates 75.6% less. This delimits the applicability of the method precisely: Statistical Ranking is not economical for one-off analyses, but it is economical for deployed detectors subjected to periodic

retraining, which is the regime Android malware detection operates in, since concept drift makes infrequent retraining a liability. These figures exclude the constant baseline cost of APK feature extraction in production, which our method does not affect. On MH100K the exchange is starker: the 99.6% reduction and the 79.6% preprocessing-time saving come with the study's largest detection gap (discussed in Section 8). We attach no significance value to that gap, since at $n = 5$ folds the paired Wilcoxon test has a minimum attainable p-value of 0.0625 and cannot separate it from the much smaller gaps elsewhere in Table 3.

6.3. Generalisation Across Classifiers

Protocol P1 evaluates the reduced subsets with Random Forest, which also supplies one of the three rankings. A reasonable concern is that the retained features are those a tree ensemble happens to favour, rather than features that are discriminative in general. We therefore evaluated the reduced MH100K subset, the most aggressive reduction in the study, with four classifiers drawn from distinct inductive families: a single decision tree, a random forest, a distance-based classifier (k-NN) and a linear model. Features were standardised for the two classifiers that require it, with the scaler fitted on training data only.

Table 5. Detection performance on the reduced MH100K subset (93 features) across four classifier families, stratified 5-fold cross-validation.

Classifier	Recall	SD	F1	MCC	Train (s)
Decision Tree	**0.8618**	0.0060	0.8568	0.8415	0.89
k-Nearest Neighbours	0.8615	0.0123	0.8442	0.8275	0.004
Random Forest	0.8587	0.0083	**0.8603**	**0.8455**	4.47
Logistic Regression	0.8399	0.0054	0.8486	0.8328	3.44
Range	**0.0219**	—	**0.0161**	**0.0180**	—

Recall spans 0.8399 to 0.8618 and Matthews correlation 0.8275 to 0.8455, ranges of 0.022 and 0.018 respectively (Table 5). The subset therefore carries signal that is not specific to tree ensembles: a k-NN classifier, which has no notion of feature importance and depends only on distances in the retained space, performs within 0.0003 recall of the decision tree. The linear model is the weakest of the four, consistent with the presence of feature interactions that a linear boundary cannot express and with the ablation finding (Section 6.4) that the multivariate ranker is the indispensable member of the ensemble.

Two limitations delimit this evidence. First, all four classifiers belong to the classical tabular family; the result does not transfer automatically to graph-, sequence-, or image-based architectures, in which an independently scored tabular feature has no direct counterpart (Section 7). Second, these figures come from cross-validating a subset selected once over the whole dataset rather than inside each fold as in Protocol P1. The comparison between classifiers is unaffected, since all four share the same subset, but the absolute values are optimistically biased: the same measurement under in-fold selection yields 0.7502 for Random Forest (Table 3) against 0.8587 here, a gap of 0.109 that evaluations on pre-reduced datasets would silently incur.

6.4. Ablation Study

The pipeline combines two cleaning filters, three ranking criteria, an adaptive per-criterion budget, and one voting threshold. We isolated the contribution of each component across

the ten benchmark datasets, excluding MH100K because the component-wise design evaluates sixteen configurations per fold, which is not tractable at that scale within our computational budget. Since the ablation compares configurations against one another rather than establishing absolute performance, this exclusion does not affect its conclusions. Significance is assessed with the Friedman test across the ten datasets, which, unlike the per-dataset Wilcoxon test, has enough blocks to be informative.

Adaptive budget and voting threshold. Table 6 compares the adaptive rule against fixed budgets of 50 and 100 features per criterion. At $\theta = 2$ the adaptive rule attains a trade-off score of 0.843 against 0.774 and 0.709, and prevails on nine of the ten datasets (Wilcoxon $p = 0.0059$). Within the adaptive budget, θ trades recall against compression monotonically: recall falls from 0.9135 to 0.9006 and 0.8795 as θ goes from 1 to 3, while reduction rises from 90.41% to 93.65% and 95.54%. Neither extreme dominates, since the union rule maximises recall and the unanimity rule maximises reduction, but the product of the two objectives is maximised at $\theta = 2$, which prevails over the union rule on nine of ten datasets ($p = 0.0605$). Requiring agreement from two of three criteria is therefore not a compromise between two extremes but the point at which both objectives are jointly best served.

Table 6. Voting threshold θ against the top-k budget, averaged over the ten benchmark datasets and five folds.

Budget	θ	#Feat.	Red. (%)	Recall	F1	Trade-off
adaptive	1	85.32	90.41	0.9135	0.9163	0.8259
adaptive	**2**	**44.24**	**93.65**	**0.9006**	**0.9052**	**0.8434**
adaptive	3	26.74	95.54	0.8795	0.8854	0.8403
$k = 50$	1	68.68	81.59	0.9164	0.9196	0.7476
$k = 50$	2	43.38	85.34	0.9073	0.9120	0.7742
$k = 50$	3	31.64	88.36	0.8953	0.8971	0.7911
$k = 100$	1	120.70	74.05	0.9190	0.9218	0.6805
$k = 100$	2	77.58	77.55	0.9142	0.9171	0.7090
$k = 100$	3	55.76	80.87	0.9065	0.9083	0.7330

Ranking criteria. Table 7 reports single-criterion selection and the three leave-one-out variants (Friedman $\chi^2 = 24.79$, $p = 0.00037$ on trade-off score). Random Forest is the indispensable member: it attains the best mean rank, the two pairs that retain it rank second and third, and the pair that omits it (C4) falls to fifth with the lowest recall of the study (0.8822). Dropping either marginal criterion from the full ensemble, by contrast, is not statistically detectable ($p = 0.883$ for MI, $p = 0.281$ for χ^2), while no single criterion is an adequate substitute for the ensemble ($p = 0.0098$ and $p = 0.0039$ for C1 and C2 against C7).

That negative result becomes interpretable once read against feature modality. The mean pairwise Jaccard overlap between the top-k lists of χ^2 and MI is 0.714 on the permission-based datasets but only 0.445 on the API-call-graph ones. The explanation is structural: for a binary feature and a binary class, both statistics are computed from the same 2×2 contingency table and induce similar orderings, whereas on continuous graph centralities they diverge. The per-dataset outcome follows this structure exactly: the full ensemble attains the best trade-off on precisely the three datasets with the lowest χ^2–MI overlap (0.339–0.359, all API-call-graph) and on no others. The value of a third ranker

Table 7. Criterion ablation with the cleaning filters and the adaptive budget held fixed; $\theta = 2$ for the two- and three-criterion configurations. KI is the Kuncheva stability index across folds.

Configuration	#Feat.	Red. (%)	Recall	F1	Trade-off	KI
C1 χ^2 only	52.10	93.20	0.8940	0.9022	0.8332	**0.981**
C2 MI only	52.10	93.20	0.8966	0.8992	0.8356	0.862
C3 RF only	52.10	93.20	**0.9108**	**0.9142**	0.8489	0.860
C4 χ^2 + MI ($-$RF)	34.02	94.58	0.8822	0.8900	0.8344	0.914
C5 χ^2 + RF ($-$MI)	29.76	95.08	0.8908	0.8979	0.8470	0.893
C6 MI + RF ($-\chi^2$)	33.94	95.06	0.8947	0.8955	**0.8505**	0.851
C7 all three (ours)	44.24	93.65	0.9006	0.9052	0.8434	0.885

is thus not a constant property of the method but a function of the redundancy among its marginal members. We retain all three criteria because the modality is not known a priori at deployment time, and because the marginal criteria account for a small fraction of the pipeline cost.

Selection stability. Stability is the property an ensemble is expected to improve, yet it is seldom reported in the Android malware feature selection literature. We measure it with the Kuncheva index between the five per-fold subsets, which corrects for the overlap expected by chance. The values do not support the strongest possible claim: the consensus subset reaches 0.885, below the 0.981 of χ^2 alone. What the ensemble does is absorb the instability of its least stable member, raising Random Forest from 0.860 to 0.885 while retaining the multivariate perspective the ablation identifies as indispensable. The trade-off is deliberate, since the most stable criterion in isolation also has the lowest trade-off score of the seven configurations: stability alone is not evidence of a good selection.

7. Threats to Validity

Conclusion validity. The paired Wilcoxon test applied within each dataset uses $n = 5$ folds, and its minimum attainable two-sided p-value at that sample size is 0.0625, above $\alpha = 0.05$. The per-dataset tests are therefore uninformative about significance, and we do not treat them as such. The aggregate analysis of Section 6.4, which applies the Friedman test over ten datasets, does not share this limitation and detects a small but consistent recall reduction that the per-dataset tests could not resolve. The direction of this bias is conservative with respect to our claims: the method's cost in detection performance is real, and we report it rather than relying on the absence of significance.

Thresholds. Both cleaning thresholds are conservative by design: a variance of 0.01 on binary features excludes only those present in fewer than roughly 1% of samples, and a Pearson correlation of 0.95 removes only near-duplicates. Section 6.4 shows that this stage removes 66.95% of the feature space without measurable loss of recall, which indicates that the values are not on the aggressive side of the trade-off. We did not sweep them, however, and cannot exclude that different values would shift the outcome.

Random Forest in a dual role. Random Forest serves both as one of the three rankers and as the classifier in Protocol P1, which could bias selection toward features that tree ensembles favour. Three pieces of evidence bound this concern: χ^2 and MI are classifier-agnostic and the $\geq$ 2-of-3 rule prevents Random Forest from unilaterally admitting a feature; Section 6.3 finds a recall spread of only 0.022 across four inductive

families; and, against our own interpretation, the ablation identifies Random Forest as the single most valuable ranker. We therefore cannot rule out that part of its measured value derives from the shared inductive bias with the evaluator.

External validity. All eleven datasets describe applications through static artefacts, and the datasets are temporal snapshots evaluated with random folds. The reported figures compare the reduced against the full-dimensional model under identical conditions and should not be read as estimates of field performance under concept drift.

8. Final Considerations

Statistical Ranking removes 89.8–99.6% of the features in public Android malware datasets while the detector remains operational: training is $8.1\times$ faster on average and, on Android Permissions, the train–test gap narrows from 0.0512 to 0.0116. Unlike single-criterion approaches, it requires agreement between at least two of three complementary rankers (χ^2, Mutual Information, and Random Forest importance), which the ablation of Section 6.4 identifies as the point that jointly maximises reduction and recall. Across the ten benchmark datasets recall remains ≥ 0.85 in nine, with a median drop of 0.04; the paired Wilcoxon test detects no degradation, but at $n = 5$ folds its floor of 0.0625 makes this an absence of evidence rather than evidence of equivalence, and nine of the ten differences favour the full set. MH100K is the exception and reads as an operational trade-off: the study's largest gap (recall from 0.8867 to 0.7502) buys a 99.6% reduction and a 79.6% preprocessing-time saving, favourable where throughput binds and unfavourable where every missed sample is costly.

Three limitations delimit these conclusions (Section 7): the evaluation is entirely static; all four classifiers are classical tabular models, leaving graph- and sequence-based architectures open; and the datasets are temporal snapshots, so the stability of the selected subsets under concept drift is the immediate next step, measurable through the timestamps in KronoDroid. Beyond that, we intend to replace the fixed voting threshold with one learned per dataset through meta-learning.

As future work, we intend to evaluate the stability of the selected subsets under concept drift, exploiting the timestamps of KronoDroid and of longitudinal datasets such as MH-1M [Bragança et al. 2026], and to replace the fixed voting threshold with a value learned per dataset through meta-learning. We further plan to decouple Random Forest from its dual role by adopting a multivariate ranker distinct from the evaluating classifier, to sweep the variance and correlation thresholds over continuous features, and to extend the validation to graph- and sequence-based architectures, as well as to domains outside Android malware detection, since the consensus rule is domain-agnostic. Finally, measuring evasion cost under partial knowledge of the subset, possibly with randomised selection, remains the open security question left by our threat model.

Acknowledgments. This work was partially supported by Capes, CNPq (Grant No. 409743/2025-9), and FAPERGS (grants no. 24/2551-0001368-7, 24/2551-0000726-1 and 25/2551-0002572-9) and FAPESP (grants no. 2020/05183-0 and 2023/00816-2). The authors also acknowledge the use of Generative AI as a supporting tool for reviewing and improving the code, manuscript, and figures.

References

Abdo, A., Mostafa, R., and Abdel-Hamid, L. (2024). An optimized hybrid approach for feature selection based on chi-square and particle swarm optimization algorithms. *Data*, 9(2).

Alalhareth, M. and Hong, S.-C. (2023). An improved mutual information feature selection technique for intrusion detection systems in the internet of medical things. *Sensors*, 23(10).

Bragança, H., Kreutz, D., Rocha, V., Assolin, J., and Feitosa, E. (2026). MH-1M: A 1.34 million-sample multi-feature android malware dataset with rich metadata. *Scientific Data*, 13:153.

da Silva, A. R. and Silveira, C. G. (2022). Handcrafted feature selection techniques for pattern recognition: A survey.

Dey, S. K., Uddin, K. M. M., Babu, H. M. H., Rahman, M. M., Howlader, A., and Uddin, K. A. (2022). Chi2-mi: A hybrid feature selection based machine learning approach in diagnosis of chronic kidney disease. *Intelligent Systems with Applications*, 16:200144.

Iranzad, R. and Liu, X. (2025). A review of random forest-based feature selection methods for data science education and applications. *International Journal of Data Science and Analytics*, 20(2):197–211.

Jain, S., Goyal, H., Arora, A., and Kumar, D. (2026). Enfestdroid: Ensembled feature selection techniques based android malware detection. *Computers and Electrical Engineering*, 129:110763.

Paim, K., Nogueira, A., Kreutz, D., Cordeiro, W., and Mansilha, R. (2025). Maldatagen: A modular framework for synthetic tabular data generation in malware detection. In *Anais Estendidos do XXV Simpósio Brasileiro de Cibersegurança*, pages 38–47, Porto Alegre, RS, Brasil. SBC.

Pudjihartono, N., Fadason, T., Kempa-Liehr, A. W., and O'Sullivan, J. M. (2022). A review of feature selection methods for machine learning-based disease risk prediction. *Frontiers in Bioinformatics*, 2.

Rocha, V., Kreutz, D., Bragança, H., and Feitosa, E. (2026a). Limitless feature selection: Revolutionizing evaluation with mh-fsf. *Journal of the Brazilian Computer Society*, 32(1):73–84.

Rocha, V., Tschiedel, L., Kreutz, D., Bragança, H., Assolin, J., Mansilha, R. B., Quincozes, S. E., and Nogueira, A. G. D. (2026b). Generalizing feature selection in android malware detection: The sigapi autocraft approach. *Journal of the Brazilian Computer Society*, 32(1):250–263.

Sharma, Y. and Arora, A. (2025). Cornetdroid: Android malware detector leveraging a correlation-based feature selection for network traffic features.

Theng, D. and Bhoyar, K. K. (2024). Feature selection techniques for machine learning: a survey of more than two decades of research. *Knowledge and Information Systems*, 66(3):1575–1637.

Tisoc, M. and Beltran, J. (2022). Mutual information and correlations. *Revista de la Facultad de Ciencias, Universidad Nacional de Ingenieria*. Received on February 14, 2022; Revised on May 25, 2022; Accepted on July 11, 2022.

Venkatesh, B. and Anuradha, J. (2019). A review of feature selection and its methods. *Cybernetics and information technologies*, 19(1):3–26.

Wu, Y., Li, M., Zeng, Q., Yang, T., Wang, J., Fang, Z., and Cheng, L. (2023). Droidrl: Feature selection for android malware detection with reinforcement learning. *Computers and Security*, 128:103126.