Uma Análise Compreensiva e Exaustiva de Métodos de Seleção de Características para Detecção de Malware Android

Vanderson Rocha[1], Diego Kreutz[2], Hendrio Bragança[1], Joner Assolin[1], Nicolas Pinto[1], Eduardo Feitosa[1]

[1]Instituto de Computação – Universidade Federal do Amazonas (UFAM)

[2]LEA, PPGES – Universidade Federal do Pampa (UNIPAMPA)

vanderson@ufam.edu.br,diegokreutz@unipampa.edu.br
{hendrio.luis,joner.assolin,efeitosa,nicolas.neves}@icomp.ufam.edu.br

Abstract. *Feature selection methods are essential for optimizing prediction accuracy and resource usage in machine learning models. In this study, we evaluate 17 selection methods using 10 distinct datasets, including 11 classical methods and 6 specific to Android malware detection. Surprisingly, classical methods like LASSO outperformed the specific ones, demonstrating excellent performance and generalization capability across most of the tested datasets.*

Resumo. *Os métodos de seleção de características são fundamentais para otimizar a predição e o uso de recursos em modelos de aprendizado de máquina. Neste estudo, avaliamos 17 métodos de seleção utilizando 10 conjuntos de dados distintos, sendo 11 métodos clássicos e 6 específicos para a detecção de malware em dispositivos Android. Surpreendentemente, métodos clássicos como o LASSO superaram os específicos, demonstrando excelente desempenho e capacidade de generalização na maioria dos conjuntos de dados testados.*

1. Introdução

A detecção de *malware* em dispositivos Android enfrenta desafios significativos devido à constante evolução e sofisticação das ameaças digitais. A seleção adequada de características é crucial nesse contexto, pois determina quais aspectos dos aplicativos são analisados pelos modelos de aprendizado de máquina para identificar comportamentos maliciosos [Dhal and Azad, 2022].

Um dos principais desafios para se detectar aplicações maliciosas é o fato de estarem continuamente se adaptando e evitando métodos tradicionais de detecção. Isso significa que as características que indicam comportamento malicioso podem mudar com o tempo, exigindo métodos de seleção de características dinâmicos, adaptáveis e com boa capacidade de generalização. Adicionalmente, o volume e a diversidade dos dados coletados de aplicativos Android são imensos: selecionar as características mais relevantes e informativas dentre milhares de permissões, chamadas de APIs, intenções, chamadas de sistema e outros dados é uma tarefa complexa. No entanto, uma escolha inadequada de características pode resultar em modelos de detecção menos eficazes, aumentando a taxa de falsos negativos, isto é, falha em detectar *malware* legítimo.

Outro desafio na construção de soluções para detecção de *malware*, que não pode ser negligenciado, está relacionado à eficiência computacional. Com recursos computacionais limitados, ou para economizar recursos financeiros (e.g., menos horas de

computação e/ou VMs mais baratas na nuvem), a seleção de um grande número de características pode aumentar significativamente o tempo necessário para treinar e validar modelos de detecção de *malware*. Isso pode ser especialmente problemático em cenários de detecção em tempo real, onde a latência é um fator crítico, e cenários com recursos computacionais limitados.

Resumidamente, uma seleção adequada de características pode levar a modelos com melhores taxas de predição e maior capacidade de generalização, além de reduzir significativamente os custos (i.e., consumo de recursos computacionais) durante o treinamento e a evolução dos modelos [Dhal and Azad, 2022]. Ademais, investigar métodos de seleção de características visa também tornar esses modelos mais eficientes e adaptáveis às mudanças das ameaças digitais. Tais esforços são essenciais para garantir a segurança digital contínua dos usuários, em um ambiente cada vez mais hostil.

Na literatura podemos encontrar uma variedade significativa de métodos de seleção de características, desde aqueles considerados clássicos (e.g., LASSO, PCA, IG, LR) até os mais sofisticados e específicos ao domínio, como MT, RFG, JOWNDroid e SemiDroid [Şahin et al., 2023b, Islam et al., 2023, Dhal and Azad, 2022, Meijin et al., 2022]. Apesar da farta literatura sobre seleção de características, os métodos são majoritariamente utilizados de forma pontual e isolada. A absoluta maioria dos trabalhos demonstram o resultado positivo da seleção de características para um número limitado de *datasets* (e.g., tipicamente um ou dois) e consideram apenas alguns poucos métodos clássicos, como ganho de informação e regressão linear, de seleção de características na comparação.

Neste trabalho, avaliamos de forma empírica, extensiva e exploratória dois tipos de métodos de seleção de características (clássicos e específicos do domínio) no contexto de detecção de *malware* Android. Para a avaliação, reproduzimos e implementamos detalhadamente 17 métodos, sendo 11 clássicos e 6 especializados no domínio e que contam com uma gama significativa de recursos avançados, como a combinação de múltiplos métodos clássicos, etapas de correlação de características e modelos de aprendizado de máquina. Nosso objetivo é observar o comportamento dos métodos quanto à qualidade do resultado de predição dos modelos aplicados ao conjunto reduzido de características selecionadas e a capacidade generalização frente a dez conjuntos de dados heterogêneos.

Como contribuições do trabalho podemos destacar: **(a)** um arcabouço de software com a implementação detalhada dos 17 métodos (11 clássicos e 6 sofisticados); **(b)** uma extensiva análise dos métodos seleção de características clássicos e específicos do domínio, aplicados em dez conjuntos de dados frequentemente utilizados no contexto de detecção de *malware* Android; **(c)** *insights* contra-intuitivos que podem determinar o rumo de pesquisas futuras de seleção de características para diferentes domínios.

O restante deste trabalho está organizado da seguinte forma: nas Seções 2 e 3, apresentamos os trabalhos relacionados e fornecemos uma descrição sucinta dos métodos de seleção de características utilizados. Em seguida, a metodologia é detalhada na Seção 4, e os resultados são discutidos na Seção 5. Por fim, as considerações finais e as sugestões para trabalhos futuros são abordadas na Seção 6.

2. Trabalhos Relacionados

Na Tabela 1 apresentamos um sumário dos trabalhos relacionados. A maioria dos estudos que realizam comparações entre métodos de seleção de características focam essencialmente em métodos tradicionais, estando entre os mais utilizados o Ganho de Informação (*Information Gain* - IG), Qui-Quadrado, Análise de Componentes Principais (*Principal Component Analysis* - PCA), Frequência Inversa de Documentos (*Term Frequency-Inverse Document Frequency* - TF-IDF) e Eliminação Recursiva (*Recursive Feature Elimination* - RFE) [Şahin et al., 2023b, Şahin et al., 2023a, Islam et al., 2023, Mahindru and Sangal, 2019, Zhao et al., 2015].

Dentre os trabalhos mais recentes e abrangentes, destacamos [Şahin et al., 2023b], que avalia onze métodos de seleção de características em um *dataset* proprietário (APKPure), utilizando as métricas de precisão, *recall* e F1. Apesar de fornecer uma avaliação mais extensa de métodos no contexto da detecção de *malwares*, algumas deficiências do trabalho merecem atenção, pois limitam a eficácia e a generalização das conclusões.

A primeira limitação é o *dataset*, pois embora possa fornecer boas informações de *malwares*, a dependência de uma única fonte de dados restringe a generalização do estudo. Além disso, a utilização de *dataset* proprietário, indisponível e sem muitos detalhes, dificulta a reprodutibilidade e a verificação independente dos resultados. Outro aspecto é o risco de *overffiting* às características específicas do APKPure, utilizado na construção do *dataset*. Uma segunda limitação significativa é o fato de os autores compararem o seu método com 10 outros métodos, implementados e avaliados em outros trabalhos, utilizando um conjunto de dados distinto, isto é, apesar de todos os trabalhos utilizarem apenas características do tipo permissões, cada trabalho utilizou um *dataset* distinto.

Exceto pelo estudo de [Islam et al., 2023], todos os demais trabalhos apresentados na Tabela 1, como [Salah et al., 2020], [Fatima et al., 2019], [Zhao et al., 2015], [Alomari et al., 2023] e [Mahindru and Sangal, 2019], podem apresentar problemas em suas conclusões pelos mesmos motivos discutidos no parágrafo anterior. Esses estudos não fornecem evidências suficientes para mitigar os riscos ou demonstrar a robustez dos métodos em diferentes conjuntos de dados. A diversidade de *datasets* que incluam diferentes tipos de *malwares* e suas características é fundamental para garantir a eficiência e a aplicabilidade dos métodos de seleção de características em diferentes ambientes.

Nossa pesquisa é significativamente mais abrangente e eficaz do que outros trabalhos relacionados. Apresentamos uma avaliação extensiva com 17 métodos de seleção de características (11 clássicos e 6 sofisticados) utilizando 10 *datasets* significativamente heterogêneos com métricas de avaliação apropriadas, o que permite avaliar de fato a capacidade de generalização dos métodos. Consequentemente, este estudo supera claramente os trabalhos relacionados em termos de robustez, confiabilidade e generalização.

É importante destacarmos que nossa diversidade de métodos é mais expressiva quando comparado aos demais trabalhos, indo de técnicas clássicas simples, como IG e PCA, até algoritmos genéticos como a Colônia Artificial de Abelhas (ABC). Ademais, outro diferencial significativo é a reprodução, implementação e avaliação sistemática de seis métodos sofisticados e específicos do domínio, incluindo SemiDroid, RFG, JOWNDroid, MT, SigPID e SigAPI. Portanto, este trabalho apresenta a primeira análise extensiva e comparativa de métodos clássicos e sofisticados aplicados ao domínio de *malwares*

Tabela 1. Ferramentas de seleção de características utilizadas no contexto de *malwares* Android.

Referência	#	Métodos	Métricas	Datasets
[Şahin et al., 2023a]	8	Ganho de Informação (IG), Razão de Probabilidade (OR), Qui-Quadrado, Frequência Inversa de Documento (IDF), Limite de Frequência de Documento, M2, ACC e ACC2	Precisão, Recall, F1	Próprio (APKPure, VirusShare)
[Şahin et al., 2023b]	11	Comparação de Permissões, Informação Mútua (MI), Subconjunto CFs, Árvore de Decisão Randomizada, Ganho de Informação (IG), Algoritmo Genético, FF-FA baseado em TF-IDF, Ganho de Informação e Frequência de Termo, Teoria dos Conjuntos Ásperos e PSO, Filtro Rápido baseado em Correlação, Regressão Linear (LR)	Precisão, Recall, F1	Próprio (APKPure)
[Islam et al., 2023]	2	RFE, PCA	Acurácia, Precisão, Recall, F1, R2	CICAndMal2020, Drebin, CICMaldroid2020
[Salah et al., 2020]	2	TF-IDF, FF-AF	Acurácia, Precisão, Recall, F1	Drebin
[Mahindru and Sangal, 2019]	8	Informação Mutua (MI), Rede Neural Profunda (DNN), Qui-Quadrado, Taxa de Ganho, OneR, Regressão Linear (LR), PCA, Ganho de Informação(IG)	Acurácia, F1	Próprio (Google Play)
[Fatima et al., 2019]	2	População de N Cromossomos	ROC, Acurácia, Sensibilidade, Especificidade, Complexidade de Tempo	Próprio (IIT Kanpur)
[Zhao et al., 2015]	3	Qui-Quadrado, Ganho de Informação (IG), FrequenSel	Acurácia, Precisão, FP Rate, Recall	Drebin
[Alomari et al., 2023]	2	LSTM, Correlações Múltiplas	Acurácia, Precisão, Recall, F1	Kaggle
Nosso Trabalho	17	Colônia Artificial de Abelhas (ABC), Análise da Variância (ANOVA), Qui-Quadrado, LASSO, Ganho de Informação (IG), Regressão Linear (LR), Desvio Médio Absoluto (MAD), Análise de Componentes Principais (PCA), Coeficiente de Correlação de Pearson (PCC), ReliefF, Eliminação Recursiva (RFE), JOWMDroid, Multi-Tiered (MT), RFG, SemiDroid, SigAPI, SigPID	Acurácia, F1, ROC, MCC	Adroit, AndroCrawl, Android Permissions, DefenseDroid PI, DefenseDroid A (C, D, K), Drebin-215, KronoDroid R, KronoDroid E

Android.

3. Métodos de Seleção de Características

Métodos de seleção de características utilizam técnicas para identificar e escolher um subconjunto de características pertinentes para a construção de modelos preditivos. Essas

técnicas visam melhorar o desempenho do modelo, removendo características irrelevantes ou redundantes, reduzindo assim o *overfitting*, melhorando a generalização e diminuindo os gastos computacionais. Em nossa pesquisa, agrupamos os métodos em dois grupos: clássicos e específicos (ou sofisticados).

Os métodos de seleção de características clássicos têm sido amplamente estudados e aplicados em uma variedade de contextos, incluindo a detecção de *malware*. Esses métodos incluem técnicas como Análise de Componentes Principais (PCA), Regressão Linear (LR) e Ganho de Informação (IG).

Por outro lado, os métodos de seleção de características específicos para o contexto de malware Android têm sido desenvolvidos para lidar com as características únicas dos aplicativos Android e os padrões de comportamento associados à aplicações maliciosas. Esses métodos levam em consideração não apenas características estáticas, como permissões e chamadas de APIs, mas também características dinâmicas, como padrões de chamadas de sistema. Ao considerar essas características específicas do domínio, é assumido que métodos de seleção de características podem melhorar a capacidade de detecção de *malware* Android. A aplicabilidade dos métodos clássicos à detecção de *malware* Android pode ser limitada devido à natureza única das características para esse domínio específico.

Para avaliar de forma sistemática os métodos de seleção clássicos e específicos no domínio de *malware* Android (sofisticados), criamos um arcabouço de software que reúne a reprodução e implementação de seis métodos sofisticados e onze métodos clássicos. Esse é um esforço de aproximadamente três anos, envolvendo mais de sete pesquisadores. O esforço já resultou em três publicações que retratam minuciosamente a construção do arcabouço, a sistemática adotada para uma reprodução e a implementação fidedigna dos métodos e validações iniciais dos primeiros métodos [Soares et al., 2022, Costa et al., 2022, Neves et al., 2023].

Entretanto, em termos de resultados, as publicações já realizadas retratam essencialmente os resultados das métricas dos métodos sofisticados incorporados ao arcabouço, considerando de três a cinco conjuntos de dados, fornecendo pistas iniciais que nos levaram a este trabalho, como o desafio da generalização dos métodos quando utilizados em diferentes conjuntos de dados do domínio.

Para este trabalho, adicionamos onze métodos clássicos ao arcabouço, seguindo os mesmos passos e rigor dos trabalhos anteriores, detalhado em [Costa et al., 2022]. Por exemplo, cada reprodução e implementação passa pela revisão e avaliação técnica de pelo menos três pesquisadores. Ademais, estamos utilizando dez *datasets* heterogêneos, o que introduz um desafio significativo em termos de capacidade de generalização.

A Tabela 2 resume os dezessete métodos disponíveis no arcabouço de software disponibilizado em repositório público no GitHub [Rocha et al., 2024]. O repositório contém a terceira geração do arcabouço, incorporando mudanças estruturais significativas e onze novos métodos quando comparado à versão anterior. É importante destacarmos que, além dos seis métodos sofisticados apresentados na Tabela 2, o arcabouço inclui outros métodos, como o FSDroid, ABC e LR. Estes métodos foram excluídos das análises por terem sido identificados como inferiores ao SemiDroid, SigPID e RFC em pelo menos 3 dos conjuntos de dados aqui utilizados, cujos resultados foram apresentados em

Tabela 2. Métodos de seleção de características implementados

Tipo	Método	Seleção	Referência
CLÁSSICOS	Colônia Artificial de Abelhas (ABC)	Subconjunto	[Karaboga et al., 2014]
	Análise da Variância (ANOVA)	Subconjunto	[Sthle and Wold, 1989]
	Qui-Quadrado	Ordenação	[Tallarida et al., 1987]
	Ganho de Informação (IG)	Ordenação	[Azhagusundari et al., 2013]
	LASSO	Subconjunto	[Ranstam and Cook, 2018]
	Regressão Linear (LR)	Subconjunto	[Şahin et al., 2023b]
	Desvio Médio Absoluto (MAD)	Ordenação	[Konno and Koshizuka, 2005]
	Análise de Componentes Principais (PCA)	Subconjunto	[Kurita, 2019]
	Coeficiente de Correlação de Pearson (PCC)	Ordenação	[Cohen et al., 2009]
	ReliefF	Ordenação	[Robnik-Šikonja and Kononenko, 2003]
	Eliminação Recursiva (RFE)	Ordenação	[Darst et al., 2018]
ESPECÍFICOS	JOWMDroid	Subconjunto	[Cai et al., 2021]
	Multi-Tiered (MT)	Subconjunto	[Bhat and Dutta, 2022]
	RFG	Subconjunto	[Alazab, 2020]
	SemiDroid	Subconjunto	[Mahindru and Sangal, 2021]
	SigAPI	Subconjunto	[Galib and Hossain, 2020]
	SigPID	Subconjunto	[Sun et al., 2016]

trabalho anterior [Neves et al., 2023].

É interessante observarmos na Tabela 3 o contexto de avaliação dos trabalhos dos seis métodos de seleção de domínio especifico aqui utilizados. Originalmente, todos eles foram concebidos e avaliados utilizando apenas um único *dataset* privado, construído a partir da combinação de amostras de outros conjuntos e/ou fontes de dados. Conjuntos de dados como Drebin, Android Permissions Dataset (Android P. D.) e Android Malware Dataset (Android M. D.) aparecem com frequência entre os conjuntos de dados utilizados para construir o *dataset* dos autores nos respectivos trabalhos. Além de utilizar apenas um único conjunto de dados próprio, não disponível para reprodução, a maioria dos autores utiliza apenas *datasets* desbalanceados, algo que pode influenciar negativamente os métodos quando aplicados em outros conjuntos de dados. Essas são algumas das particularidades que ajudam a explicar alguns dos resultados abaixo do esperado quando comparados com métodos clássicos, como discutimos em detalhes na Seção 5.

4. Metodologia da Experimentação

Nesta seção, apresentamos os conjuntos de dados utilizados (Seção 4.1) e as técnicas aplicadas na avaliação do métodos (Seção 4.2). Ademais, na Seção 4.3, apresentamos também o ambiente e a configuração utilizada nos experimentos.

4.1. Conjuntos de Dados

Para avaliar e comparar os métodos apresentados, utilizamos dez conjuntos de dados frequentemente adotados em pesquisas no contexto de detecção de *malware* Android

Tabela 3. Apresentação das características do tipo intenção (I), permissão (P) e chamada de API (A); e *datasets* dos métodos específicos para o domínio de malware Android.

Método	Características	Balanceado	# Cars.	# Amostras	*Datasets*
SemiDroid	P + A	Não	1842	500K	Próprio (APKs + Android P. D.)
RFG	A	Não	27253	36915	Próprio (VirusTotal + AndroZoo + etc.)
JOWNDroid	P + A	Não	643	163556	Próprio (Drebin + AMD + APKs)
MT	P + A + I	Não	61	11449	Próprio (Virustotal, VirusShare + Drebin)
SigPID	P	Sim	135	310926	Próprio (APKs + Android M. D.)
SigAPI	A	Não	142	18769	Próprio (Drebin + Android P. D.)

[Soares et al., 2021]. Os *datasets* utilizados foram pré-processados, a fim de remover valores nulos, características irrelevantes (aquelas que possuem um único valor em todas as amostras) e amostras duplicadas (quando identificadas pelo *hash*).

Na Tabela 4, detalhamos os dez *datasets* em termos de quantidade, os tipos de características chamadas de API (A), Permissões (P), Intenções (I), *OpCodes* (O), e a quantidade de amostras benignas e malignas. Como podemos observar, há uma heterogeneidade representativa entre os conjuntos de dados, variando significativamente as quantidades de amostras, tipos e número de características.

Para o dataset DefenseDroid, utilizamos quatro variantes. Primeiro, o conjunto contendo permissões (P) e intenções (I), denominado de DefenseDroid PI. Segundo, três variações de chamadas de API (A), denominados DefenseDroid A (C, D e K), onde C, D e K representam as três variantes que utiliza as técnicas de *clossenes* (C), *degree* (D) e *katz* (K), utilizadas na normalização dos dados para geração do *dataset* tabular e binário.

Tabela 4. Sumário dos dez *datasets* utilizados neste estudo.

	Adroit[1]	AndroCrawl[2]	Android[3] Permissions	DefenseDroid PI[4]	DefenseDroid A			Drebin-215[5]	KronoDroid R.[6]	KronoDroid E.
					C	D	K			
Características										
Qtde.	166	81	151	2938	4275	6003	6003	215	246	268
Tipo	P	A (24) I (8) P (49)	P	P (1490) I (1448)	A			A (73) P (113) O (6) I (23)	P (146) A (100)	P (145) A (123)
Amostras										
Maliciosas	3418	10170	17787	6000	5254			5560	41382	28745
Benignas	8058	86562	9077	5975	5222			9476	36755	35246
Total	11476	96732	26864	11975	10476			15036	78137	63991

É importante destacarmos que *datasets* como o AndroCrawl possuem diversas características compostas, algumas delas criadas por especialistas. Isto introduz uma entropia ainda mais significativa, tornando o conjunto de dados escolhido um ótimo candidato para avaliação da capacidade de generalização dos métodos de seleção de características.

[1] https://www.kaggle.com/datasets/saurabhshahane/android-malware-dataset
[2] https://github.com/phretor/ransom.mobi/blob/gh-pages/f/filter.7z
[3] https://www.kaggle.com/datasets/saurabhshahane/android-permission-dataset
[4] https://github.com/DefenseDroid/DefenseDroid
[5] https://doi.org/10.6084/m9.figshare.5854653.v1
[6] https://github.com/aleguma/kronodroid

Como alguns desses conjuntos de dados possuem muitas características (e.g., mais de 6k), realizamos uma seleção preliminar das 1000 (mil) melhores características. Para essa pré-seleção, utilizamos o valor de Informação Mútua (*Mutual Information* - MI) para identificar as características irrelevantes ou menos significativas no *dataset*.

Com base em estudos empíricos realizados ao longo dos últimos três anos, constatamos que a maioria desses *datasets* possui menos de 100 características realmente relevantes para a detecção de aplicações maliciosas em Android. Portanto, acreditamos que nossa redução preliminar não impacta significativamente os resultados. Essa redução inicial foi aplicada apenas nas variantes do DefenseDroid para viabilizar a execução dos métodos sofisticados, uma vez que alguns deles podem demandar um tempo considerável de execução (e.g., mais de uma semana em nosso ambiente de experimentação) em conjuntos de dados com um número maior de características (e.g. mais de 5k).

Neste trabalho, significativamente extenso, focamos nas métricas de desempenho dos modelos resultantes dos *datasets* reduzidos gerados pelos métodos de seleção de características. Em trabalhos futuros, pretendemos apresentar também aspectos como complexidade analítica e eficiência computacional de cada método.

4.2. Parâmetros e Métricas

Para avaliarmos os *datasets* produzidos pelos métodos de seleção de características, chamados de *datasets* reduzidos, utilizamos três classificadores, o KNN (agrupamento), RF (árvore) e SVM (basedo em margem). Para a implementação e experimentação, utilizamos a configuração padrão da biblioteca *scikit-learn* versão 1.4.2 [Pedregosa et al., 2011].

Para o KNN, utilizamos 5 (cinco) vizinhos com pesos iguais para as consultas. O RF foi configurado com 100 árvores na floresta, considerando a raiz quadrada do número de características ao buscar a melhor divisão entre as classes. Para o SVM, utilizou-se o kernel *Radial Basis Function* (RBF) e heurísticas para reduzir o tempo de treinamento.

Como métricas de avaliação, além das tradicionais (acurácia, precisão, *recall* e F1 Score), utilizamos também o Coeficiente de Correlação de Matthews (MCC) [Chicco and Jurman, 2020], indicando a correlação entre as predições e as classes reais. O MCC fornece uma medida balanceada que pode ser usada mesmo se as classes tiverem tamanhos diferentes. Combinando precisão e a cobertura da predição de maneira equilibrada, o MCC resulta em um valor alto somente se o classificador obtiver resultados ótimos em todas as quatro células da matriz de confusão [Cao et al., 2020, Chicco and Jurman, 2020].

É importante destacarmos que a *recall* é uma das métricas mais relevantes no contexto de detecção de *malwares* Android, pois ela representa a porcentagem de aplicativos reconhecidamente maliciosos e detectados como tal. Para a comparação dos métodos, utilizamos as métricas *recall*, F1 e MCC.

Utilizamos também a técnica de **validação cruzada estratificada**, que consiste em dividir o conjunto de dados em K partições (*folds*). Como o número de amostras nos conjuntos de dados é geralmente desbalanceado em *datasets* de *malware*, optamos por utilizar sua variação estratificada, pois mantém a proporção original de cada classe na etapa de teste. Frequentemente, a validação cruzada é realizada utilizando $K = 5$ ou $K = 10$, uma vez que estes valores produzem estimativas de taxa de erro de teste que

não sofrem de viés ou variância excessivamente altos [James et al., 2013]. Por padrão, a biblioteca *scikit-learn* utiliza $K = 5$, valor que adotamos nos experimentos.

4.3. Ambiente de execução dos experimentos

Para a execução dos experimentos e reprodutibilidade, utilizamos um computador com processador Intel Xeon E5-4617 Octa-core de 2.90GHz, 32GB RAM, armazenamento SSD de 150GB e sistema operacional Linux Ubuntu 22.04 LTS. Para a automação da execução dos métodos de seleção de características, evoluímos e utilizamos o nosso próprio arcabouço de software [Costa et al., 2022, Neves et al., 2023]. A implementação atual utiliza Python (versão 3.10.12) e as bibliotecas *numpy* (versão 2.0.0), *pandas* (versão 2.2.2), *scikit-learn* (versão 1.5.1) e *refieff* (versão 0.1.2).

5. Resultados

Na sequência (Seção 5.1), apresentamos o desempenho dos 17 métodos de seleção de características para 10 conjuntos de dados significativamente heterogêneos. Para comparar os resultados, utilizamos as métricas de F1, *recall* e MCC. Avaliamos nossos resultados com base em três classificadores, KNN, SVM e RF.

Finalmente, encerramos a avaliação da nossa pesquisa com uma discussão dos principais achados do trabalho (Seção 5.2). Acreditamos que nossas descobertas representam um passo significativo em termos de direções futuras e investigações adicionais, muito necessárias, no contexto de avaliação sistemática de métodos de seleção de características no domínio de detecção de *malware* Android.

5.1. Desempenho dos Métodos de Seleção de Características

Os resultados da Tabela 5 apresentam os melhores métodos (em ordem decrescente) em relação a F1 e ao *recall*. Esses resultados foram obtidos para cada método de seleção de características, agregando-se os resultados obtidos pelos três modelos, para cada *dataset*. Complementarmente, apresentamos também o MCC de cada método para cada *dataset* no mapa de calor da Figura 1.

Notamos que o método clássico LASSO alcançou o melhor desempenho geral, com valor médio da F1 de 0,9071 e desvio padrão de 0,0717. Isso indica que o LASSO teve um bom desempenho de forma consistente em todos os *datasets* avaliados, indicando uma maior capacidade de generalização para esse tipo de conjuntos de dados. Já os melhores métodos específicos (sofisticados) foram o RFE e SigAPI, que obtiveram uma F1 média muito próximos entre si e em relação ao LASSO, com desvios padrão comparativamente baixos. Estes resultados destacam a eficácia desses três métodos e também no equilíbrio alcançado entre a detecção das instâncias verdadeiramente positivas de *malwares* e a limitação da ocorrência de instâncias falso-positivas.

Em contraste, o método clássico ReliefF e o específico JOWMDroid apresentaram os piores resultados, com valores médios da F1 de 0,6352 e 0,6361, respectivamente, juntamente com grandes níveis de variabilidade. A diversidade observada implica em um nível inconsistente de desempenho, o que mostra a ineficiência desses métodos na identificação e categorização de *malwares*.

É interessante observar que temos métodos clássicos e específicos em ambos os extremos e com resultados similares. Esses resultados são um forte indicativo da falta

Tabela 5. Resumo do desempenho usando a métrica F1 Score e Recall para métodos de seleção de características em todos os conjuntos de dados.

Método	F1		Método	Recall	
	Média	Desvio		Média	Desvio
LASSO	0.9071	0.0717	LASSO	0.9086	0.0586
RFE	0.9030	0.0768	RFE	0.9034	0.0705
SigAPI	0.9008	0.0659	SigAPI	0.9027	0.0573
PCC	0.8997	0.0735	MAD	0.9004	0.0616
Qui-Quadrado	0.8996	0.0713	IG	0.9001	0.0658
IG	0.8994	0.0728	PCC	0.8996	0.0668
MAD	0.8990	0.0731	Qui-Quadrado	0.8981	0.0666
ANOVA	0.8960	0.0681	ANOVA	0.8958	0.0586
SemiDroid	0.8934	0.0923	SemiDroid	0.8942	0.0954
LR	0.8744	0.0541	RFG	0.8691	0.0579
RFG	0.8665	0.0591	LR	0.8665	0.0724
MT	0.7838	0.2636	MT	0.7893	0.2743
ABC	0.7320	0.2742	ABC	0.7195	0.2797
SigPID	0.6861	0.3139	PCA	0.6708	0.3294
PCA	0.6650	0.2962	SigPID	0.6525	0.3377
JOWMDroid	0.6361	0.3340	JOWMDroid	0.6486	0.3649
ReliefF	0.6352	0.2780	ReliefF	0.6253	0.3148

de uma análise mais aprofundada dos métodos específicos *versus* as dezenas de métodos clássicos. Infelizmente, a maioria dos autores que propõe métodos de seleção de características limitam-se a comparar o método proposto com mais um a três outros métodos, possivelmente comprometendo uma análise mais efetiva e justa da proposição de um novo método de seleção de características.

Em nosso próximo resultado, conforme mostra a Tabela 6, apresentamos um *ranking* dos melhores métodos de seleção para cada *dataset*, obtidos pela média de desempenho dos modelos de classificação. Apresentamos na tabela os valores de *recall* e F1, bem como o desvio padrão, por conjunto de dados.

Mais uma vez, o LASSO se destacou na grande maioria dos conjuntos de dados, com uma média de 0,9086 e um desvio padrão de 0,0586, indicando uma variação relativamente baixa. Os métodos RFE e SigAPI tiveram um desempenho similar, com resultados muito próximos aos alcançados pelo LASSO.

Os melhores resultados dos *datasets* formados unicamente por chamadas de API (C: *closeness*, D: *degree*, K: *katz*) são obtidos pelo método SigAPI, com MCC de 88,00%, 87,43% e 87,55%, respectivamente (ver mapa de calor da Figura 1). Isso ocorre porque esse método é especializado nesse tipo de características. Além disso, apesar de não atingir o melhor resultado geral, o SigAPI obtém bons resultados nos demais *datasets*,

Tabela 6. F1 e Recall dos melhores métodos de seleção de características em cada *dataset* avaliado.

Dataset	Método	F1		Método	Recall	
		Média	Desvio		Média	Desvio
Adroit	LR	0.7952	0.0015	LASSO	0.7817	0.0252
	RFG	0.7933	0.0217	MAD	0.7815	0.0331
	SigAPI	0.7901	0.0266	SigAPI	0.7806	0.0168
AndroCrawl	SigAPI	0.9239	0.0038	SigAPI	0.9088	0.0023
	LASSO	0.9197	0.0091	LASSO	0.8984	0.0152
	RFE	0.913	0.0136	RFG	0.8966	0.0078
Android Permissions	ReliefF	0.7877	0.0135	ReliefF	0.9617	0.0355
	SigPID	0.7779	0.0333	SigPID	0.9368	0.0931
	ANOVA	0.7764	0.0228	PCA	0.926	0.0964
DefenseDroid A (C)	SigAPI	0.9316	0.0079	SigAPI	0.9188	0.0022
	LASSO	0.9281	0.0125	RFE	0.9185	0.0079
	RFE	0.9281	0.0116	LASSO	0.9185	0.0077
DefenseDroid A (D)	SigAPI	0.9292	0.0074	SigAPI	0.9183	0.0049
	LASSO	0.9259	0.011	LASSO	0.9159	0.0084
	RFE	0.9237	0.0111	RFE	0.9144	0.01
DefenseDroid A (K)	SigAPI	0.9302	0.0074	SigAPI	0.9173	0.0055
	LASSO	0.9259	0.011	LASSO	0.9159	0.0084
	RFE	0.9237	0.0111	RFE	0.9144	0.01
DefenseDroid PI	LASSO	0.9121	0.0112	MT	0.8901	0.0602
	Qui-Quadrado	0.9118	0.0098	RFE	0.8885	0.0055
	MAD	0.9114	0.0107	Qui-Quadrado	0.8875	0.0054
Drebin-215	RFE	0.9781	0.0052	MAD	0.9706	0.0044
	LASSO	0.9777	0.0055	RFE	0.9704	0.0042
	MAD	0.9774	0.0057	LASSO	0.9696	0.0056
KronoDroid E	LASSO	0.9573	0.0094	LASSO	0.9548	0.0101
	RFE	0.9571	0.0102	RFE	0.9543	0.0106
	SemiDroid	0.9571	0.0094	SemiDroid	0.9542	0.0101
KronoDroid R	LASSO	0.9709	0.0053	LASSO	0.9654	0.0071
	RFE	0.9705	0.0058	Qui-Quadrado	0.9648	0.0071
	MAD	0.9705	0.0056	SemiDroid	0.9647	0.0073

especificamente quando há chamadas de API.

Podemos notar também que o método JOWMDroid figura sistematicamente entre os piores resultados para cada conjunto de dados, com valores médios de 0,6253 e 0,6486, com desvios padrão bem elevados quando comparado aos outros métodos avaliados. Isso pode ser explicado devido a uso da Informação Mutua (MI) como um dos passos do método que, em conjunto de dados menores, pode ignorar atributos que são relevantes

quando combinados, mas não têm alto MI individualmente.

Por fim, nosso ultimo resultado é apresentado através do mapa de calor na Figura 1, que apresenta o MCC nos conjuntos de dados e métodos de seleção de características apresentados em nosso estudo. Optamos por também mostrar o MCC porque ele é uma métrica que avalia a qualidade das classificações binárias, sendo muito útil particularmente em conjuntos de dados desequilibrados. Valores mais altos de MCC indicam melhor desempenho, com valores variando de -1 (-100%) a +1 (+100%), onde +1 representa previsão perfeita, 0 não é melhor que previsão aleatória e -1 discordância completa entre previsão e observação.

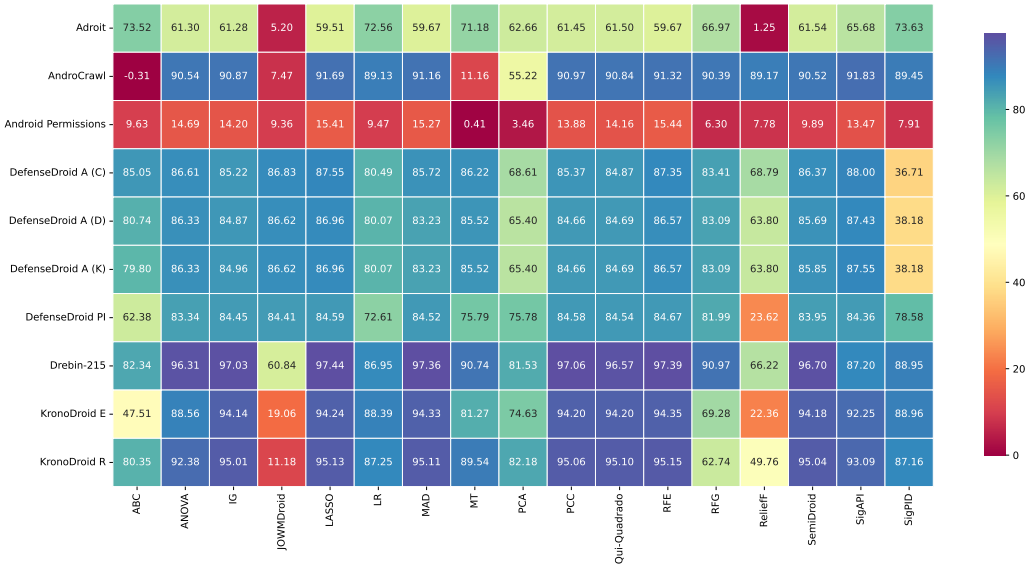

Figura 1. MCC: *malware datasets* x métodos de seleção de características.

No resultado do MCC, LASSO e RFE também se destacam como os métodos com melhor desempenho em todos os conjuntos de dados avaliados, alcançando valores consistentes de MCC e adaptabilidade para diferentes *datasets*. Esses métodos demonstram ser viáveis na detecção de *malware*. Em particular, o LASSO esteve entre os melhores resultados, como no KronoDroid R e E, com MCC de 97,44% (melhor desempenho), ou como no KronoDroid E com MCC de 94,24% (terceiro melhor), apresentando um diferença de 0,11% para o melhor resultado (Figura 1).

Diferente de alguns *datasets* que apresentaram MCC próximo a zero, o *dataset* AndroCrawl apresentou um valor negativo (-0.31%), indicando uma predição inversa com uso do método ABC. Tal *dataset* apresenta classes altamente desbalanceadas, o que pode ter afetado o processo de seleção, favorecendo características que não são representativas.

É interessante observarmos que os melhores resultados dos *datasets* formados unicamente por chamadas de API (DefenseDroid A (C), (D) e (K)) são obtidos novamente pelo método SigAPI (MCC de 88,00%, 87,43% e 87,55%, respectivamente). De fato, este método é especializado nesse tipo de características, o que parece ser corroborado pelos resultados similares em todos os cenários. Apesar de não atingir o melhor resultado geral, com pequena margem de diferença em relação ao LASSO, o SigAPI obtém bons resultados nos demais *datasets* também. Isto sugere que os autores do método foram

bastante diligentes na sua concepção e avaliação, resultando em uma boa capacidade de generalização.

Os dados apresentados indicam que LASSO, RFE e SigAPI são as técnicas mais confiáveis, exibindo um desempenho médio alto e pouca variação tanto na F1 quanto no *recall*. Estas abordagens oferecem um forte equilíbrio entre precisão e *recall*, que podem fornecer uma detecção eficiente e uniforme de *malware*. No entanto, as técnicas como ReliefF e JOWMDroid apresentam restrições, ressaltando a necessidade de mais refinamento ou outras estratégias para garantir a identificação correta de *malwares*. Esta revisão destaca a necessidade de escolher métodos de detecção fortes para melhorar a confiabilidade e a precisão das soluções de detecção de *malwares*, levando, em última análise, a ambientes computacionais mais seguros.

Ademais, esses resultados indicam um possível viés dos autores na concepção de métodos como o JOWMDroid, cujos testes são limitados (e.g., um único conjunto de dados privado não disponível para reprodução) e uma comparação igualmente limitada (e.g., apenas dois outros métodos clássicos escolhidos de maneira muitas vezes questionável).

5.2. Discussão

Em primeiro lugar, é importante destacarmos que a avaliação de diferentes métodos de seleção de características, utilizando uma quantidade mais representativa e significativa de conjuntos de dados (e.g., dez), é crucial para o desenvolvimento de modelos eficazes de classificação de aplicativos maliciosos. Os resultados obtidos com essas avaliações fornecem informações sobre os pontos fortes e fracos de diferentes métodos, que podem orientar futuros esforços de pesquisa.

A seguir, apresentamos algumas questões de pesquisa que procuramos responder. As respostas e resultados podem ajudar outros pesquisadores com o direcionamento e a evolução de pesquisas relacionadas à métodos de seleção de características.

Na detecção de malwares Android, os métodos de seleção de características de domínio especifico são mais eficientes do que os clássicos?

De modo geral, alguns métodos clássicos de seleção de características se demonstraram mais eficientes que os métodos de domínio especifico. Contudo, ambos os tipos de métodos têm suas próprias vantagens e desvantagens.

A maioria dos métodos de seleção de características consegue identificar bons subconjuntos de características nos *datasets*. No entanto, esses métodos podem enfrentar dificuldades com *datasets* complexos ou que contêm um grande número de características, o que pode levar a um ajuste inadequado quando aplicados a conjuntos de dados que utilizam características diferentes das quais foram desenvolvidos. Um exemplo disso é o comportamento do SigPID, que é especializado em permissões, ao lidar com *datasets* baseados em chamadas de API.

Como apresentado na Seção 5.1, uma seleção de características adequada é capaz de levar a modelos de classificação com resultados melhores. Logo, é importante considerar cuidadosamente as características do conjunto de dados antes de escolher qual método usar para a detecção de *malwares* Android. Além disso, é possível que uma combinação de diferentes algoritmos e técnicas possa produzir os melhores resultados.

Qual melhor método de seleção de características para detecção de malware Android?

Com base no MCC dos resultados apresentados, apresentado no mapa de calor da Figura 1, podemos notar que o LASSO obteve um bom desempenho em quase todos os conjuntos de dados. Isso indica que o LASSO é um método promissor para selecionar características capazes de auxiliar os modelos de aprendizado de maquina a detectar amostras maliciosas. O método apresentou valores de MCC consistentes em todos os resultados e, por este motivo, é um forte candidato a ser escolhido como um seletor de características confiável neste cenário.

No entanto, também é importante observar que o desempenho do LASSO foi adverso no *dataset* Adroit. Acreditamos que o motivo para esse resultado é que o LASSO tende a selecionar uma única variável de um grupo de variáveis altamente correlacionadas e ignorar as demais. Como as características do *dataset* Adroit são altamente correlacionadas, o LASSO pode não ser capaz de selecionar todas as características relevantes, levando a uma perda de informações importantes.

Finalmente, é importante destacarmos ainda que disponibilizamos também a relação completa de todos os resultados, para todos os 17 métodos, 10 *datasets* e 3 modelos, no repositório do arcabouço de software no GitHub [Rocha et al., 2024]. Ademais, disponibilizamos no repositório também todos os dados das execuções com os dez conjuntos de dados balanceados, oferecendo condições para uma análise ainda mais ampla e rica dos métodos de seleção[7]. Isto permite aos leitores e pesquisadores aprofundar análises e discussões pontuais relacionadas a métodos ou conjuntos de dados específicos.

6. Considerações Finais e Trabalhos Futuros

Neste trabalho avaliamos 17 métodos de seleção de características, sendo 11 clássicos e 6 sofisticados, utilizando 10 conjuntos de dados, para o domínio de detecção *malware* Android. Os resultados indicam que o método clássico LASSO consegue ser igual ou superior aos melhores métodos sofisticados, como RFE e SigAPI. Esse é um resultado contra-intuitivo, pois o natural seria métodos sofisticados serem superiores aos métodos clássicos. Tipicamente, métodos sofisticados incorporam uma combinação de métodos e técnicas clássica, como PCA, IG, ANOVA e LR. Além disso, utilizam também técnicas de correlação e modelos de aprendizado de máquina para, supostamente, melhorar a seleção de características.

Ademais, é importante destacarmos que disponibilizamos o arcabouço e os extensos resultados do trabalho no GitHub [Rocha et al., 2024]. No repositório os leitores e pesquisadores poderão encontrar todas as implementações, *scripts* de automação dos experimentos, todos os 10 conjuntos de dados utilizados, bem como todas as tabelas e gráficos referentes aos resultados detalhados de cada conjunto de dados (original e balanceado) e cada um dos 3 modelos (KNN, SVM e RF) utilizados.

Como trabalhos futuros, podemos incluir (a) avaliar outras abordagens para seleção de características, buscando reduzir os falsos positivos e falsos negativos por meio da análise das amostras que não foram classificadas corretamente, (b) combinar diferentes métodos para seleção de características, (c) incorporar *datasets* com *multiclasse* na

[7]Devido a limitações de espaço, optamos por apresentar apenas os dados dos *datasets* originais, sem balanceamento, neste trabalho.

avaliação, (d) incluir métodos de seleção de características utilizados em outros trabalhos, como os baseados em TF-IDF, (e) agrupamento das amostras dos conjuntos de dados por famílias, buscando especializar ainda mais a seleção de características, (f) avaliar o desempenho a escalabilidade e eficiência computacional dos métodos em conjuntos de dados significativamente grandes (e.g., mais de 100k amostras e 20k características).

Agradecimentos. Esta pesquisa foi parcialmente financiada, conforme previsto nos Arts. 21 e 22 do Decreto No. 10.521/2020, nos termos da Lei Federal No. 8.387/1991, através do convênio No. 003/2021, firmado entre ICOMP/UFAM, Flextronics da Amazônia Ltda e Motorola Mobility Comércio de Produtos Eletrônicos Ltda. O presente trabalho foi realizado também com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001 e da FAPERGS, através dos editais 08/2023 e 09/2023.

Referências

Alazab, M. (2020). Automated Malware Detection in Mobile App Stores Based on Robust Feature Generation. *Electronics*, 9:435.

Alomari, E. S., Nuiaa, R. R., Alyasseri, Z. A. A., Mohammed, H. J., et al. (2023). Malware Detection Using Deep Learning and Correlation-Based Feature Selection. *Symmetry*, 15(1):123.

Azhagusundari, B., Thanamani, A. S., et al. (2013). Feature Selection Based on Information Gain. *IJITEE*, 2(2):18–21.

Bhat, P. and Dutta, K. (2022). A Multi-Tiered Feature Selection Model for Android Malware Detection Based on Feature Discrimination and Information Gain. *Journal of King Saud University - Computer and Information Sciences*, 34(10, Part B):9464–9477.

Cai, L., Li, Y., and Xiong, Z. (2021). JOWMDroid: Android Malware Detection Based on Feature Weighting with Joint Optimization of Weight-Papping and Classifier Parameters. *Computers & Security*, 100:102086.

Cao, C., Chicco, D., and Hoffman, M. M. (2020). The MCC-F1 Curve: A Performance Evaluation Technique for Binary Classification. *arXiv preprint arXiv:2006.11278*.

Chicco, D. and Jurman, G. (2020). The Advantages of the Matthews Correlation Coefficient (MCC) over F1 Score and Accuracy in Binary Classification Evaluation. *BMC Genomics*, 21(1):1–13.

Cohen, I., Huang, Y., Chen, J., Benesty, J., Benesty, J., Chen, J., Huang, Y., and Cohen, I. (2009). Pearson Correlation Coefficient. *Noise Reduction in Speech Processing*, pages 1–4.

Costa, E., Kreutz, D., Rocha, V., Leão, L., Sabóia, S., Neves, N., and Feitosa, E. (2022). FS3E: Uma Ferramenta para Execução e Avaliação de Métodos de Seleção de Características para Detecção de Malwares Android. In *Anais Estendidos do XXII SBSeg*, pages 151–158.

Darst, B. F. et al. (2018). Using Recursive Feature Elimination in Random Forest to Account for Correlated Variables in High Dimensional Data. *BMC Genetics*, 19:1–6.

Dhal, P. and Azad, C. (2022). A Comprehensive Survey on Feature Selection in the Various Fields of Machine Learning. *Applied Intelligence*, 52(4):4543–4581.

Fatima, A., Maurya, R., et al. (2019). Android Malware Detection Using Genetic Algorithm Based Optimized Feature Selection and Machine Learning. In *TSP*, pages 220–223. IEEE.

Galib, A. H. and Hossain, M. (2020). Significant API Calls in Android Malware Detection (Using Feature Selection Techniques and Correlation Based Feature Elimination). In *SEKE*.

Islam, R., Sayed, M. I., Saha, S., et al. (2023). Android Malware Classification Using Optimum Feature Selection and Ensemble Machine Learning. *IOTCPS*, 3:100–111.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R*. Springer.

Karaboga, D., Gorkemli, B., Ozturk, C., and Karaboga, N. (2014). A Comprehensive Survey: Artificial Bee Colony (ABC) Algorithm and Applications. *AI Review*, 42:21–57.

Konno, H. and Koshizuka, T. (2005). Mean-Absolute Deviation Model. *IIE Transactions*, 37(10):893–900.

Kurita, T. (2019). Principal Component Analysis (PCA). *Computer Vision: A Reference Guide*, pages 1–4.

Mahindru, A. and Sangal, A. (2019). Deepdroid: Feature Selection Approach to Detect Android Malware Using Deep Learning. In *IEEE ICSESS*, pages 16–19. IEEE.

Mahindru, A. and Sangal, A. L. (2021). SemiDroid: A Behavioral Malware Detector Based on Unsupervised Machine Learning Techniques Using Feature Selection Approaches. *International Journal of Machine Learning and Cybernetics*, 12(5):1369–1411.

Meijin, L., Zhiyang, F., Junfeng, W., et al. (2022). A Systematic Overview of Android Malware Detection. *Applied AI*, 36(1).

Neves, N., Rocha, V., Kreutz, D., et al. (2023). Avaliação de Métodos de Seleção de Características de Amostras Android com a Ferramenta FS3E (v2). In *Anais da XX ERRC*, pages 139–144.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Ranstam, J. and Cook, J. A. (2018). LASSO Regression. *Journal of British Surgery*, 105.

Robnik-Šikonja, M. and Kononenko, I. (2003). Theoretical and Empirical Analysis of ReliefF and RReliefF. *Machine Learning*, 53:23–69.

Rocha, V., Bragança, H., Kreutz, D., and Feitosa, E. (2024). MH-FSF: um Framework para Reprodução, Experimentação e Avaliação de Métodos de Seleção de Características. `https://github.com/SBSegSF24/MH-FSF`.

Şahin, D. Ö., Kural, O. E., Akleylek, S., and Kılıç, E. (2023a). A Novel Android Malware Detection System: Adaption of Filter-Based Feature Selection Methods. *JAIHC*, pages 1–15.

Şahin, D. Ö., Kural, O. E., et al. (2023b). A Novel Permission-Based Android Malware Detection System Using Feature Selection Based on Linear Regression. *Neural Comp Appl*, pages 1–16.

Salah, A., Shalabi, E., and Khedr, W. (2020). A Lightweight Android Malware Classifier Using Novel Feature Selection Methods. *Symmetry*, 12(5):858.

Soares, T., Kreutz, D., Rocha, V., Costa, E., Leão, L., Pontes, J., Assolin, J., Rodrigues, G., and Feitosa, E. (2022). Uma Análise de Métodos de Seleção de Características Aplicados à Detecção de Malwares Android. In *Anais do XXII SSBSeg*, pages 288–301.

Soares, T., Siqueira, G., Barcellos, L., et al. (2021). Detecção de Malwares Android: Datasets e Reprodutibilidade. In *Anais da XIX ERRC*, pages 43–48, Porto Alegre, RS, Brasil. SBC.

Sthle, L. and Wold, S. (1989). Analysis of Variance (ANOVA). *Chemometr Intell Lab.*, 6(4):259–272.

Sun, L., Li, Z., Yan, Q., Srisa-an, W., and Pan, Y. (2016). SigPID: Significant Permission Identification for Android Malware Detection. In *MALWARE*, pages 1–8. IEEE Computer Society.

Tallarida, R. J., Murray, R. B., Tallarida, R. J., and Murray, R. B. (1987). Chi-Square Test. *Manual of Pharmacologic Calculations with Computer Programs*, pages 140–142.

Zhao, K., Zhang, D., Su, X., and Li, W. (2015). Fest: A Feature Extraction and Selection Tool for Android Malware Detection. In *IEEE ISCC*, pages 714–720. IEEE.