MH-FSF: um Framework para Reprodução, Experimentação e Avaliação de Métodos de Seleção de Características

Vanderson Rocha[1], Hendrio Bragança[1], Diego Kreutz[2], 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, efeitosa}@icomp.ufam.edu.br

Resumo. *Neste artigo apresentamos o framework MH-FSF, que reúne a reprodução e implementação de métodos de seleção de características de forma integrada, modular e extensível. Este é um esforço que envolveu diversos pesquisadores ao longo dos últimos anos. Para uma avaliação da extensiva do MH-FSF, implementamos e disponibilizamos 17 métodos de seleção de características, agrupados em 11 clássicos e 6 específicos de domínios, para o contexto de detecção de malware Android.*

1. Introdução

A seleção de características é essencial para a construção de modelos preditivos eficazes, desempenhando um papel crucial na redução da dimensionalidade dos dados, identificando e escolhendo subconjuntos de características pertinentes. Isto, não apenas melhora a precisão dos modelos como também diminui o tempo e os recursos computacionais necessários para o treinamento [Dhal and Azad, 2022]. Ao identificar e utilizar apenas as características mais relevantes, os métodos de seleção de características podem simplificar os modelos, tornando-os mais interpretáveis e eficientes [Naheed et al., 2020].

Muitas pesquisas são conduzidas em contextos específicos, utilizando *datasets* proprietários que não são disponibilizados publicamente. Essa prática dificulta a comparação direta entre diferentes abordagens, pois os parâmetros e resultados não podem ser replicados ou validados [Soares et al., 2021]. Como consequência, há uma grande dificuldade em avaliar a eficácia relativa dos diversos métodos de seleção de características, o que limita o avanço e a aplicabilidade das inovações nessa área.

Além disso, a maioria absoluta dos trabalhos (e.g., [Mahindru and Sangal, 2021], [Galib and Hossain, 2020], [Alazab, 2020], [Cai et al., 2021], [Bhat and Dutta, 2022], [Sun et al., 2016]) comparam a sua proposta com um conjunto bastante reduzido de métodos similares de seleção de características. Assim como no caso dos conjuntos de dados, isso resulta em uma visão parcial e potencialmente enviesada do desempenho e das vantagens de cada método proposto. A ausência de uma comparação mais exaustiva impede uma avaliação mais justa e abrangente dos métodos de seleção de características, reduzindo a confiança nas conclusões apresentadas.

A dificuldade em encontrar *datasets* representativos para avaliação e comparação é outra barreira significativa [Soares et al., 2021, Bragança et al., 2023]. A escassez de conjuntos de dados padronizados e publicamente disponíveis impede a validação consistente dos métodos propostos e a reprodução dos resultados em diferentes contextos e

aplicações. Esta limitação é crítica, pois a diversidade e a representatividade dos *datasets* são fundamentais para garantir que os métodos de seleção de características sejam robustos e generalizáveis.

No contexto da detecção de *malware* Android, a seleção de características é ainda mais importante devido à natureza complexa dos aplicativos. Os aplicativos podem ter um grande número de permissões, componentes e chamadas de APIs, o que torna desafiador a identificação de características relevantes.

A ferramenta MH-FSF tem como objetivo preencher as lacunas identificadas na literatura, através de um *framework* estruturado, configurável e extensível para métodos de seleção de características, fornecendo uma solução que suporta a inovação e a avaliação de novos métodos de seleção de características. Com isso, esperamos contribuir para o avanço da área, promovendo a utilização de técnicas mais eficazes e eficientes em uma ampla gama de aplicações preditivas.

Utilizando e analisando os resultados dos 17 métodos de seleção de características sobre 10 conjuntos de dados significativamente heterogêneos, ressaltamos que a variação entre diferentes *datasets* desbalanceados evidencia a complexidade da análise em dados reais, onde a presença de classes desequilibradas é comum. Abordar essa disparidade exige estratégias adequadas de pré-processamento de dados ou a utilização de métodos de seleção de características que considerem essa assimetria, com o objetivo de aumentar a precisão e confiabilidade das predições.

Nossas descobertas reafirmam a importância de ferramentas como a MH-FSF, projetada para integrar uma ampla variedade de métodos de seleção, permitindo uma comparação direta e abrangente entre diferentes abordagens. Todas as nossas avaliações são realizadas em 10 conjuntos de dados públicos e representativos, facilitando a replicabilidade e a validação dos resultados. Isso promove a padronização nas pesquisas de seleção de características e oferece uma ferramenta escalável e acessível.

Como contribuições do trabalho podemos destacar: **(a)** um *framework* contendo a reprodução e implementação detalhada dos 17 métodos de seleção de características (11 clássicos e 6 específicos ao domínio); **(b)** uma análise dos métodos clássicos e específicos do domínio, aplicados em 10 conjuntos de dados frequentemente utilizados no contexto de detecção de *malwares* Android;

2. A Ferramenta MH-FSF

A ferramenta MH-FSF é composta por quatro etapas principais: (a) manipulação dos **dados**; (b) métodos de **seleção de características**, (c) **treinamento e avaliação** com modelos de aprendizado de máquina, e (d) **visualização** dos resultados. A Figura 1 apresenta a visão geral da MH-FSF.

A primeira etapa envolve a seleção e preparação de um conjuntos de dados. Os *datasets* selecionados visam dar a representatividade e a robustez necessária para os métodos de seleção de características. A etapa de preparação de dados é necessária para garantir a qualidade e a integridade dos dados, o que inclui processos como a remoção de valores nulos (NaN), duplicatas, o balanceamento de classes, anotações e a subamostragem.

Na sequência, a seleção de características reduz a dimensionalidade dos dados e identifica as características mais relevantes para o modelo de classificação. A atual versão

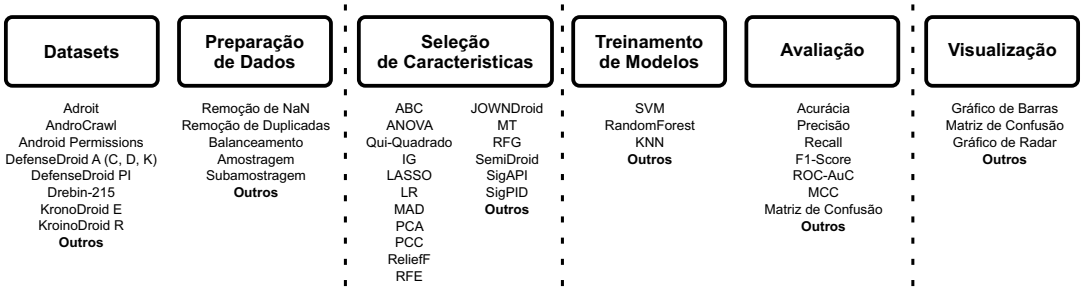

Figura 1. Visão geral da MH-FSF: As quatro principais etapas do *pipeline*.

da MH-FSF incorpora métodos de seleção de características específicos do domínio de detecção de *malwares* Android, como SemiDroid [Mahindru and Sangal, 2021], RFG [Alazab, 2020], JOWNDroid [Cai et al., 2021], MT [Bhat and Dutta, 2022], SigPID [Sun et al., 2016] e SigAPI [Galib and Hossain, 2020], e métodos clássicos de seleção de características, como ANOVA, Qui-Quadrado, LASSO, PCA, ReliefF e RFE. Cada método oferece uma abordagem distinta para avaliar e selecionar as características mais significativas, contribuindo para a construção de modelos mais precisos e eficientes.

Com as características selecionadas, na próxima etapa a MH-FSF procede para o treinamento de modelos utilizando algoritmos de aprendizado de máquina, como SVM, RandomForest e KNN. Após o treinamento, os modelos são avaliados utilizando métricas de desempenho como acurácia, precisão, *recall*, F1, ROC-AUC e MCC.

Por fim, os resultados são visualizados através de gráficos de barras, matrizes de confusão, gráficos de radar e outras técnicas de visualização. Estas ferramentas visuais são fundamentais para interpretar os resultados, identificar padrões e comunicar os achados de forma clara e intuitiva.

Além dos blocos básicos que compõem a ferramenta, podemos destacar outras vantagens do *framework* disponibilizado pela MH-FSF, como **extensibilidade**, permitindo que novos métodos de seleção de características sejam integrados de forma simples e rápida. Isso possibilita a evolução da ferramenta com as necessidades emergentes da pesquisa e do desenvolvimento de novas técnicas, mantendo-se atualizada com os avanços na área de aprendizagem de máquina. A capacidade de incorporar novos métodos sem grandes esforços de reestruturação é um diferencial importante de longevidade e relevância contínua da ferramenta.

Para adicionar um novo método de seleção de características ao *framework* da MH-FSF, basta: (a) criar um novo diretório com o identificador do método na pasta `methods`, (b) adicionar os arquivos `about.desc`, contendo a descrição do método, e o (c) código do método no arquivo `run.py`. No `run.py`, importe as bibliotecas necessárias utilizadas pelo novo método e defina duas funções obrigatórias: `add_arguments`, que adiciona argumentos específicos do novo método no `argparse.ArgumentParser`, e `run`, que executa o método de seleção de características e salva o *dataset* reduzido ao final da execução.

Sua estrutura oferece uma **metodologia diversificada**, suportando diferentes métodos de seleção de características, proporcionando **flexibilidade**, permitindo sua aplicação em diversos domínios e com variados *datasets*. Além disso, a MH-FSF conta

com um recurso de **personalização**, que permite aos usuários adaptarem as estratégias de seleção de características de acordo com os requisitos específicos de cada domínio.

Devido à sua concepção estrutural simples e independente, a MH-FSF oferece **escalabilidade** intrínseca, uma vez que os métodos são independentes e não há limite para a quantidade de métodos que podem ser incorporados. Em termos de **desempenho**, a ferramenta permite a paralelização da execução dos métodos, além incorpora mecanismos de tratamento de erros e registro, garantindo a confiabilidade e facilitando a depuração durante a execução

Detalhes adicionais sobre os conjuntos de dados, a implementação da ferramenta, as instruções de instalação, os ambientes e parâmetros de execução e outras informações técnicas estão disponíveis no repositório Github da MH-FSF[1].

3. Experimentação

Para avaliar e comparar os 17 métodos incorporados na MH-FSF, catalogamos e utilizamos 10 *datasets* públicos para a avaliação dos métodos de seleção de características e o treinamento de modelos de detecção de *malwares* Android. Na Tabela 1, detalhamos os *datasets* em termos de quantidade e tipos de características (chamadas de API (A), permissões (P), intenções (I) e *Op Codes* (O)), bem como a quantidade de amostras benignas e maliciosas. Como pode ser observado, há uma heterogeneidade significativa entre os conjuntos de dados, com variações expressivas nas quantidades de amostras, tipos e número de características.

Tabela 1. Sumário dos *datasets* utilizados pela ferramenta MH-FSF.

Dataset	Características		Desbalanceados		Balanceados	
	Qtde.	Tipos	Maliciosas	Benignas	Maliciosas	Benignas
Adroit[2]	166	P	3418	8058	3418	3418
AndroCrawl[3]	81	A (24) I (8) P (49)	10170	86562	10170	10170
Android Permissions[4]	151	P	17787	9077	9077	9077
DefenseDroid PI[5]	2938	P (1490) I (1448)	6000	5975	5975	5975
DefenseDroid A (C, D, K)	4275, 6003, 6003	A	5254	5222	5222	5222
Drebin-215[6]	215	A (73) P (113) O (6) I (23)	5560	9476	5560	5560
KronoDroid R.[7]	246	P (146) A (100)	41382	36755	36755	36755
KronoDroid E.	268	P (145) A (123)	28745	35246	28745	28745

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,

[1] https://github.com/SBSegSF24/MH-FSF
[2] https://www.kaggle.com/datasets/saurabhshahane/android-malware-dataset
[3] https://github.com/phretor/ransom.mobi/blob/gh-pages/f/filter.7z
[4] https://www.kaggle.com/datasets/saurabhshahane/android-permission-dataset
[5] https://github.com/DefenseDroid/DefenseDroid
[6] https://doi.org/10.6084/m9.figshare.5854653.v1
[7] https://github.com/aleguma/kronodroid

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.

Para avaliarmos os *datasets* produzidos pelos métodos de seleção de características (*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].

Como métricas de avaliação, além das tradicionais (acurácia, precisão, *recall* e F1), utilizamos também o Coeficiente de Correlação de Matthews (MCC), que mede a correlação entre as predições e as classes reais. O MCC resulta em valores altos apenas quando o classificador obtém resultados ótimos em todas as quatro células da matriz de confusão [Cao et al., 2020].

Como o número de amostras nos conjuntos de dados é geralmente desbalanceado, optamos por utilizar a **validação cruzada estratificada**, que mantém a proporção original de cada classe na etapa de avaliação. A validação cruzada é frequentemente realizada com $K = 5$ ou $K = 10$, pois esses valores produzem estimativas da taxa de erro de teste com viés e variância controlados [James et al., 2013]. Por padrão, a biblioteca *scikit-learn* utiliza $K = 5$, valor que adotamos em nossos experimentos.

Para a execução dos experimentos e reprodutibilidade, utilizamos um computador com processador Intel(R) Xeon(R) CPU E5-4617 de 2.90GHz, com 64GB RAM e armazenamento de 800GB. O sistema operacional utilizado foi o Linux Ubuntu 22.04 LTS. Mais detalhes sobre os ambientes de experimentação podem ser encontrados no repositório Github da MH-FSF.

4. Resultados

Os resultados da Tabela 2 apresentam os melhores métodos de seleção de características (em ordem decrescente) considerando as métricas *recall* e F1. Esses resultados foram obtidos para cada método de seleção, agregando-se os resultados obtidos pelos três modelos, para cada conjunto de dados, tanto balanceado quanto desbalanceado.

Os métodos LASSO e RFE apresentam os melhores desempenhos, com médias de F1 e *recall* acima de 0.9. Esses métodos destacam-se por sua capacidade consistente de identificar características relevantes, mantendo baixo desvio padrão, o que sugere uma estabilidade na performance em diferentes contextos de pesquisa.

Por outro lado, métodos como PCA, ReliefF e SigPID apresentam desempenho inferior, com médias de F1 abaixo de 0,75 e valores de *recall* correspondentes mais baixos. Esses resultados indicam uma capacidade limitada desses métodos em manter a precisão das características relevantes e em lidar eficazmente com a variação nos dados.

Além disso, apresentamos também o mapa de calor do MCC para cada método em dois tipos de *datasets*: completos (Figura 2) e balanceados (Figura 3). O MCC oferece *insights* significativos sobre a eficácia dos métodos. Por exemplo, observamos que métodos como LASSO e RFE demonstram consistentemente um desempenho superior em conjuntos de dados como KronoDroid R, KronoDroid E e Drebin-215, tanto nas versões balanceadas quanto nas desbalanceadas (*datasets* completo). Isso indica que essas técnicas são capazes de identificar e selecionar características relevantes de forma mais precisa, mesmo quando as classes estão representadas de maneira desproporcional.

Tabela 2. Valores de F1 e *recall* dos métodos nos *datasets*.

Balanceados				Desbalanceados			
Método	**F1**	**Método**	**Recall**	**Método**	**F1**	**Método**	**Recall**
LASSO	0.90	LASSO	0.89	LASSO	0.91	LASSO	0.91
RFE	0.90	RFE	0.89	RFE	0.90	RFE	0.90
SemiDroid	0.90	SemiDroid	0.89	SigAPI	0.90	SigAPI	0.90
JOWMDroid	0.90	JOWMDroid	0.89	MAD	0.90	PCC	0.90
MAD	0.90	Qui-Quadrado	0.88	IG	0.90	Qui-Quadrado	0.90
PCC	0.90	MAD	0.88	PCC	0.90	IG	0.90
Qui-Quadrado	0.90	PCC	0.88	Qui-Quadrado	0.90	MAD	0.90
IG	0.90	IG	0.88	ANOVA	0.90	ANOVA	0.90
ANOVA	0.90	ANOVA	0.88	SemiDroid	0.90	SemiDroid	0.90
LR	0.87	SigAPI	0.86	RFG	0.87	LR	0.87
SigAPI	0.87	LR	0.86	LR	0.87	RFG	0.87
ABC	0.85	ABC	0.84	MT	0.79	MT	0.78
MT	0.77	MT	0.76	ABC	0.72	ABC	0.73
RFG	0.72	RFG	0.70	PCA	0.67	SigPID	0.67
ReliefF	0.71	ReliefF	0.68	SigPID	0.65	PCA	0.66
SigPID	0.71	SigPID	0.66	JOWMDroid	0.65	JOWMDroid	0.64
PCA	0.67	PCA	0.63	ReliefF	0.63	ReliefF	0.64

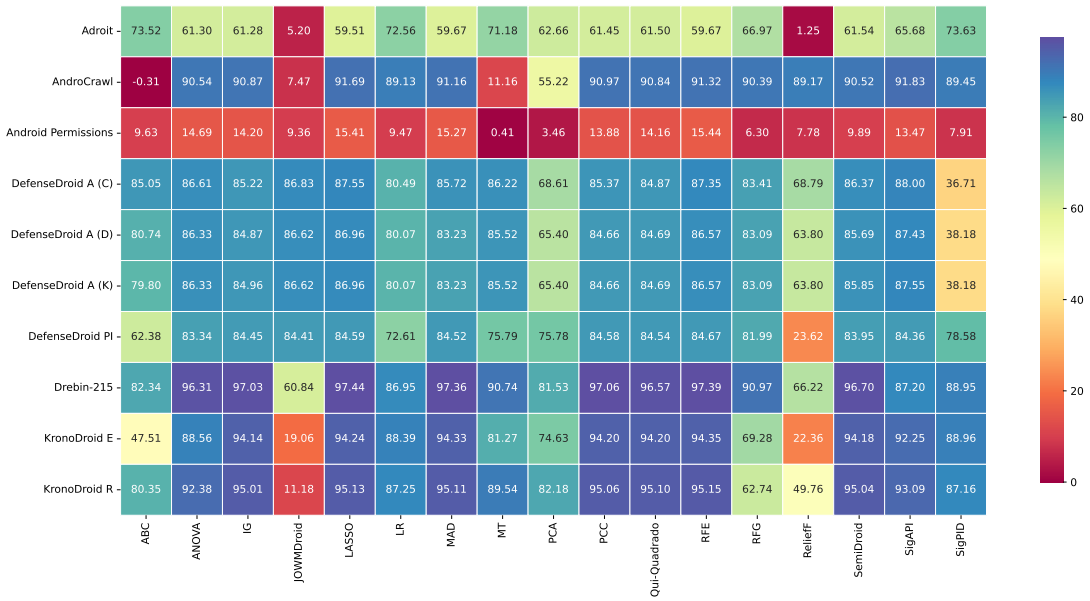

Figura 2. MCC: *Datasets* completos *vs.* métodos de seleção de características.

É importante destacarmos também que *datasets* como Adroit, AndroCrawl e Android Permissions revelam alguns desafios para os métodos de seleção de características. Os métodos que se destacaram nas versões balanceadas dos *datasets* (e.g., ABC, JOWMDroid, PCA) demonstraram um desempenho superior na maioria dos casos, conforme refletido pelo MCC mais altos. Isso sugere que a predominância de uma classe sobre a outra nos dados desbalanceados pode comprometer a capacidade dos métodos de seleção de características em distinguir efetivamente os comportamentos maliciosos.

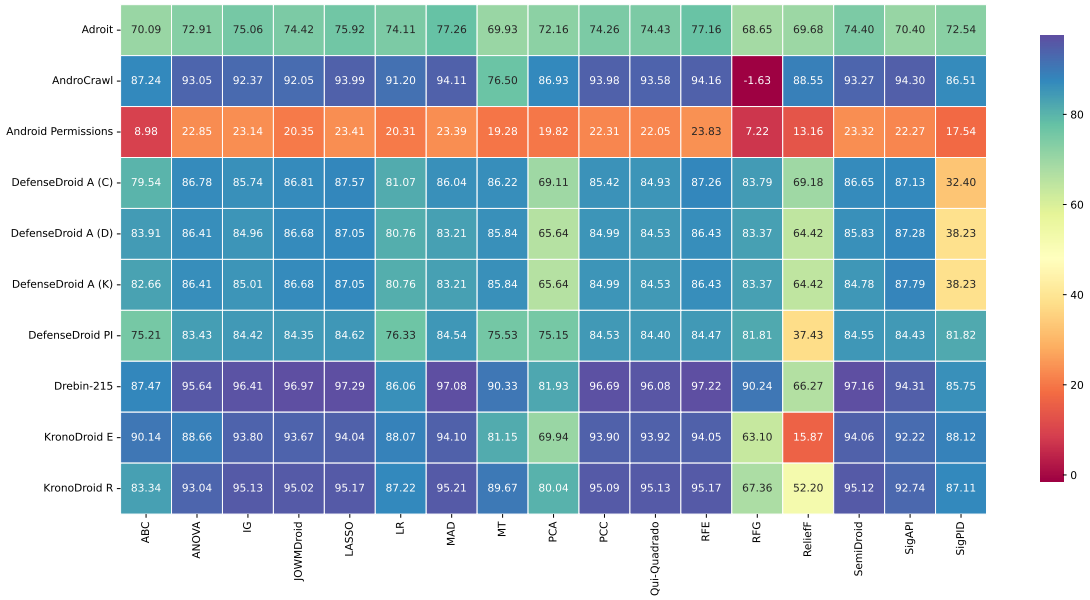

Figura 3. MCC: *Datasets* balanceados *vs.* métodos de seleção de características.

A variação nos resultados entre diferentes *datasets* desbalanceados destaca a complexidade da análise em dados reais, onde desequilíbrios nas classes são comuns. Lidar com essa disparidade requer estratégias adequadas de pré-processamento de dados ou a adoção de métodos de seleção de características adaptados para considerar essa assimetria, a fim de melhorar a precisão e a confiabilidade das predições.

Os resultados obtidos demonstram que o balanceamento adequado dos dados é um requisito essencial para maximizar a eficiência dos métodos de seleção de características, garantindo que as características mais relevantes sejam identificadas independentemente da distribuição das classes nos *datasets*.

5. Conclusão e Demonstração

A ferramenta MH-FSF foi projetada para facilitar a incorporação de uma ampla variedade de métodos de seleção de características, permitindo uma comparação direta e abrangente entre diferentes abordagens no processo de seleção de características, modelos de classificação e métricas de avaliação. Nossos resultados revelam a importância da MH-FSF para avaliar e comparar extensivamente métodos de seleção de características em contextos diversos e complexos, como o domínio de *malwares* Android.

Como trabalhos futuros podemos destacar: (a) incorporação de novos métodos de seleção de características; (c) avaliação de novos *datasets*, modernos e atualizados; (d) utilização de técnicas de explicabilidade (XAI) para entender o comportamento dos métodos de seleção; (e) avaliação dos métodos de seleção em um ambiente com coleta de dados em tempo real; e (f) investigar como ataques adversariais podem influenciar a seleção das características.

Demonstração. A ferramenta será apresentada utilizando um microcomputador portátil dos autores. A demonstração irá incluir: (a) apresentação dos módulos e funcionalidades; (b) sumarização dos métodos e conjuntos de dados disponíveis na ferramenta; (c) introdução aos parâmetros de execução; (d) apresentação da execução e resultados para

diferentes métodos e *datasets*.

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.

Bhat, P. and Dutta, K. (2022). A Multi-Tiered Feature Selection Model for Android Malware Detection Based on Feature Discrimination and Information Gain. *King Saud Univ. - Comp, and Inf. Sciences*, 34(10, Part B).

Bragança, H., Rocha, V., Souto, E., Kreutz, D., and Feitosa, E. (2023). Capturing the Behavior of Android Malware with MH-100K: A Novel and Multidimensional Dataset. In *XXIII SBSeg*.

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*.

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.

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*.

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

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

Naheed, N., Shaheen, M., Khan, S. A., et al. (2020). Importance of Features Selection, Attributes Selection, Challenges and Future Directions for Medical Imaging Data: A Review. *CMES*, 125(1).

Pedregosa, F., Varoquaux, G., et al. (2011). Scikit-learn: Machine Learning in Python. *JMLR*, 12:2825–2830.

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

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.