Geração de dados sintéticos tabulares para detecção de *malware* Android: um estudo de caso

Angelo Gaspar Diniz Nogueira[1], Kayua Oleques Paim[2], Hendrio Bragança[3], Rodrigo Mansilha[1], Diego Kreutz[1]

[1]LEA e PPGES, Universidade Federal do Pampa (UNIPAMPA) – Alegrete, Brasil

[2]IComp, Universidade Federal do Amazonas (UFAM) – Manaus, Brasil

[3]IC, Universidade Federal do Rio Grande do Sul (UFRGS) – Porto Alegre, Brasil

Resumo. *Apresentamos um estudo sobre o uso de cGANs para expandir* datasets *de* malware *Android. Após uma exploração empírica de hiperparâmetros, aplicamos nossa cGAN para expandir quatro* datasets *e avaliamos os resultados considerando métricas de fidelidade estatística e a utilidade para algoritmos de ML na classificação de aplicativos Android. Os resultados confirmam a importância do ajuste adequado de hiperparâmetros, bem como a capacidade das cGANs de sintetizar* datasets *fiéis e úteis.*

1. Introdução

Para combater as ameaças e mitigar riscos relacionados a *malware* Android, técnicas de aprendizado supervisionado têm sido amplamente empregadas [Meijin et al., 2022]. Porém, a eficiência desses métodos está intimamente relacionada à quantidade de amostras (p. ex., número suficientemente representativo de amostras), aos tipos de características extraídas dessas aplicações, como permissões, intenções, chamadas de APIs, e à qualidade (p. ex., novidade) dos *datasets* utilizados no treinamento.

Esse tema tem sido foco de estudos recentes que apontam que entre os maiores problemas, que levam mais de 80% dos projetos de IA ao fracasso, está a escassez de dados em termos quantitativos e qualitativos para o treinamento e a construção de modelos realistas e eficazes [AI & Data Today, 2023]. A quantidade, qualidade e atualidade dos dados impactam diretamente as métricas resultantes dos modelos desenvolvidos [Allix, K. et. al., 2015].

Uma estratégia para ampliar a qualidade, quantidade e atualidade de *datasets* de *malware* Android é desenvolver métodos sistemáticos para obtenção de dados. Entretanto, estudos recentes demonstram que a obtenção de dados atualizados têm se demonstrado um desafio significativo [Rocha et al., 2023], isto é, esses métodos são extremamente custosos e demorados.

Recentemente, a comunidade tem explorado técnicas emergentes de inteligência artificial generativa como alternativa complementar ao método de obtenção sistemática de *datasets* [Zhao et al., 2024, Rajabi and Garibay, 2022]. Nosso desafio de pesquisa reside em moldar essas tecnologias para uso apropriado no contexto de *datasets malware* Android considerando pelo menos quatro aspectos. Primeiro, é necessário se considerar a evolução de modelos generativos que surgem a cada dia e sua

complexidade para implementação e treinamento adequados, como o fato de receberem como entrada dados tabulares e não imagens. Segundo, é necessário estudar os respectivos conjuntos de parâmetros fixos da rede (p. ex., número de épocas), conhecidos como hiperparâmetros e seus valores, o que exige experimentação extensiva. Terceiro, é necessário atingir níveis de estabilidade no modelo para evitar colapso durante o treinamento. Por fim, é necessário avaliar a qualidade dos dados gerados, o que ainda é considerado um problema em aberto [Platzer and Reutterer, 2021].

Neste trabalho, investigamos a utilização de arquiteturas de redes neurais artificiais do tipo rede adversária generativa condicional, do inglês conditional generative adversarial network (cGAN) [Mirza and Osindero, 2014], para ampliar *datasets* e assim obter melhores classificadores e resultados. Para a avaliação das arquiteturas de redes neurais, utilizamos quatro conjuntos de dados distintos e métricas de *fidelidade* estatística e *utilidade* para uso com técnicas de aprendizado supervisionado. Realizamos experimentações extensas para definir os melhores hiperparâmetros. Os resultados são positivos quando avaliamos as métricas de fidelidade e utilidade para os quatro conjuntos de dados utilizados.

2. Trabalhos relacionados

Na Tabela 1 apresentamos os principais trabalhos relacionados no contexto de aumento de dados tabulares. Destacamos a técnica, as métricas e os *datasets* utilizados. Como podemos observar, todas as técnicas envolvem a arquitetura GAN ou variações dela. Em termos de métricas, todos trabalhos considerados utilizam métricas clássicas, como precisão, acurácia, *recall* e pontuação F1. Diferentemente dos demais trabalhos, utilizamos métricas de *fidelidade* estatística como erro quadrático médio e similaridade de cosseno. Ademais, na avaliação de *utilidade* utilizamos o teste de hipótese *Mann-whitney* [McKnight and Najab, 2010] através do cálculo do valor de p.

A maioria dos trabalhos explora dados tabulares de diferentes contextos e domínios, como medicina, fraudes, censo, entre outros. Há também trabalhos que utilizam dados de *malware* considerando imagens e aplicações *Visual Basic*, por exemplo. Diferentemente, neste trabalho investigamos e utilizamos dados tabulares do contexto de *malware* Android.

Além dos trabalhos apresentados na Tabela 1, há outros que também exploram técnicas no contexto de dados tabulares, como [Snow, 2020, Fang et al., 2022, Fakoor, R. et. al., 2020, Machado et al., 2022]. O que há em comum entre os trabalhos é o fato de aplicarem diferentes técnicas em diferentes domínios e contextos para tentar gerar dados tabulares de qualidade.

Finalmente, é importante destacarmos que, como pode ser observado nos trabalhos relacionados, a geração de dados sintéticos de qualidade é um tema recente e ainda incipiente. Ainda não há consenso sobre métricas e técnicas a serem utilizadas, por exemplo. Ao que sabemos, este trabalho é o primeiro a investigar técnicas generativas para dados tabulares considerando quatro *datasets* distintos no contexto de *malware* Android.

Tabela 1. Trabalhos relacionados

Referência	Técnica	Métricas	Datasets
[Tanaka and Aranha, 2019]	GANs	Recall, Precisão, desvio padrão, distância euclidiana	2 de Medicina 1 de Fraudes
[Lu and Li, 2019]	DCGAN	Recall, Precisão, F1-Score	1 de Malware (Imagens)
[Mimura, 2020]	GANs	Acurácia, Recall, F1-Score	1 de Malware (Visual Basic Application)
[Rajabi and Garibay, 2022]	GANs	Acurácia, F1-Score, discrimination Score	4 de Censo
[Zhao et al., 2024]	cGANs	Acurácia, F1-Score, Curva ROC JSD, WD, AUC, correlação entre pares, MAPE, EDS e R^2	7 Tabulares (Geral)
Este trabalho	cGAN	Acurácia, Recall, Precisão, F1-Score, erro quadrático, similaridade de cosseno e valor de p	4 de Malware (Android)

3. Metodologia

Na Figura 1, ilustramos a nossa metodologia. Primeiramente, é necessário balancear os *datasets* usando técnicas de amostragem. Em seguida, o *dataset* balanceado é dividido em dobras, que são subconjuntos criados para a validação cruzada. Para cada dobra é realizada uma iteração de treino e avaliação da cGAN. A arquitetura geral de uma cGAN contém dois modelos principais: um gerador, que cria novos exemplos de dados a partir de entradas condicionais, e um discriminador, que tenta distinguir entre exemplos reais e gerados. Os conjuntos de treinamento são utilizados tanto para treinar o discriminador quanto o gerador, enquanto os conjuntos de avaliação são utilizados posteriormente para avaliar o desempenho da cGAN.

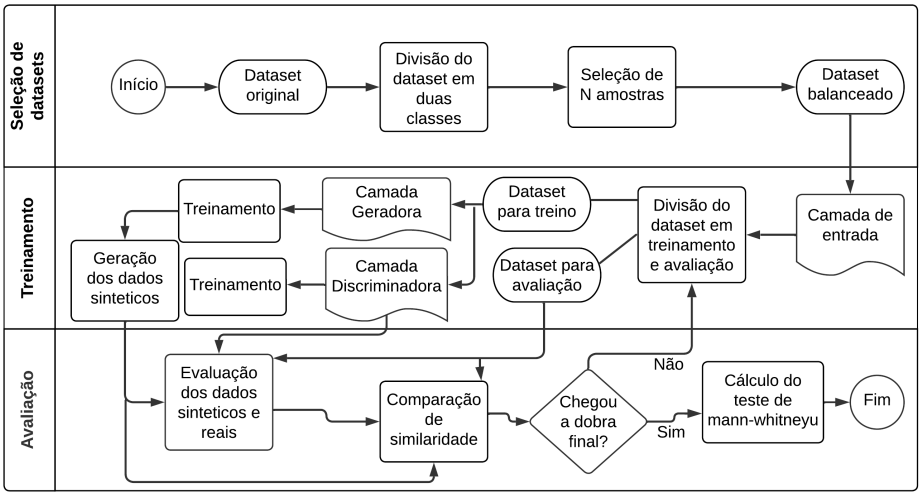

Figura 1. Fluxograma de execução.

Após realizarmos as avaliações de cada dobra, aplicamos o teste *Mann-whitney* [McKnight and Najab, 2010] sobre as distribuições das métricas de desempenho escolhidas, conforme detalhado na Seção 3.2.

3.1. Datasets

Na Tabela 2 apresentamos as especificações de cada *dataset* considerado neste estudo em termos de quantidade de características (p. ex, requer acesso à câmera), percentual de amostras de cada classe binária de aplicação (*malware* ou benignas), número total de amostras de aplicações, e o total de amostras consideradas após a realização do balanceamento (50% malware e 50% benigno).

Tabela 2. Datasets considerados neste estudo.

Dataset	Características	Amostras			N. Amostras (Balanceado)
		Malware	Benignos	Total	
Adroit	166	29,80%	70,20%	11476	6836
Drebin	215	37%	63%	15031	11110
Kronodroid	286	53%	47%	78137	20000
Android P	151	66%	34%	26864	18154

3.2. Métricas de Desempenho

Neste estudo consideramos métricas de *fidelidade* do cosseno e do erro quadrático para verificar se o *dataset* sintético reproduz as características estatísticas do *dataset* original. Isso inclui não apenas a distribuição de características individuais, mas também as correlações multivariáveis entre elas, assegurando que o conjunto sintético se assemelhe à mesma população do conjunto original. Adicionalmente, as amostras positivas e negativas são avaliadas separadamente.

Consideramos também uma métrica de *utilidade*, para avaliar o desempenho dos classificadores de aprendizagem de máquina treinados com dados sintéticos (p. ex, método conhecido como TRTS proposto em [Esteban et al., 2017]). Na avaliação de *utilidade* utilizamos as métricas de acurácia, precisão, *recall* e pontuação F1. Para verificar a validade estatística das métricas foi empregado o teste de *Mann-whitney*.

O teste de *Mann-whitney* opera sobre duas amostras independentes com duas hipóteses: H0, que afirma que não há diferença estatisticamente significativa entre as distribuições das amostras, e H1, que afirma que há uma diferença estaticamente significativa. Isto é determinado através do cálculo de U sobre a soma do *ranking* ordenado de observações da mesma amostra. Esse cálculo resulta em um valor de p, que é comparado com um limiar *threshold* para rejeitar ou aceitar a hipótese H0. Nós utilizamos a média calculada entre os valores de p resultantes das métricas de *utilidade* para a nossa hipótese. A razão para seu uso é obter uma avaliação geral dos parâmetros dos classificadores.

Para avaliar a utilidade dos dados gerados, consideramos quatro classificadores clássicos: *Random Forest* (RF), *Decision Tree* (DT), *Perceptron* e *Stochastic Gradient Descent Regressor* (SGDR). Em cada dobra, aplicamos os classificadores no *dataset* real e no sintético criado com cada uma das topologias da cGAN. Com base nos resultados, geramos as matrizes de confusão e as curvas ROC e extraímos as métricas de *utilidade* previamente definidas.

3.3. Ambiente e Tecnologias

Para a implementação e execução da arquitetura básica da nossa cGAN, utilizamos Python (versão 3.8) e as seguintes bibliotecas principais: *numpy* 1.21.5, *Keras* 2.9.0,

Tensorflow 2.9.1, *pandas* 1.4.4 e *scikit-learn* 1.1.1. Detalhes da implementação da cGAN, recursos utilizados e resultados de todas as execuções podem ser vistos no **repositório GitHub [GitHub, 2024]**.

4. Resultados

Na sequência (Seção 4.1), apresentamos a lista de hiperparâmetros utilizados em nossos experimentos, selecionados após uma série de avaliações empíricas. Ressaltamos a importância dos hiperparâmetros para a estabilidade da rede geradora e seus resultados. Em seguida, apresentamos os resultados através das métricas de *utilidade* (Seção 4.2) e *fidelidade* (Seção 4.3).

4.1. Hiperparâmetros

A Tabela 3 apresenta uma análise comparativa de hiperparâmetros utilizados no treinamento de um modelo cGAN, implementado pelos autores, para a classificação de *malware* Android em quatro conjuntos de dados distintos: Kronodroid, Adroit, Debrin e Android P. Esses valores foram determinados empiricamente, após a realização de vários experimentos para identificar os mais adequados para cada *dataset*.

Tabela 3. Hiperparâmetros utilizados nos experimentos.

hiperparâmetros	**Kronodroid**	**Adroit**	**Debrin**	**Android P**
Dense Layer Sizes (d)	1024	1400	1800	2048
Dense Layer Sizes (g)	2048	1800	3012	3012
Dropout Decay Rate (d)	0,4	0,4	0,4	0,3
Dropout Decay Rate (g)	0,2	0,2	0,2	0,1
Initializer Deviation	0,004	0,01	0,003	0,001
Initializer Mean	0,1	0,1	0,1	0
Latent Dimension	128	150	128	128
Latent Stander Deviation	0,5	1	0,8	0,8
Number Epochs	500	450	300	450
Optimizer Discriminator Learning	0,0001	0,0001	0,0002	0,0004
Optimizer Generator Learning	0,0001	0,0001	0,003	0,0002
Conditional GAN ADA Delta Learning Rate	0,001	0,001	0,002	0,002
Conditional GAN RMS Prop Learning Rate	0,001	0,001	0,002	0,002
Conditional GAN ADAM Learning Rate	0,0001	0,0001	0,0002	0,0002

A variabilidade nos hiperparâmetros nos conjuntos de dados ressalta a necessidade de estratégias de otimização personalizadas para lidar com as características e desafios particulares de cada *dataset* avaliado. Por exemplo, os hiperparâmetros *dense layer sizes (d)* e *dense layer sizes (g)* indicam, respectivamente, as quantidades de camadas densas do discriminador e do gerador da arquitetura cGAN. Os hiperparâmetros utilizados foram escolhidos de forma empírica.

A descrição destalhada sobre o impacto e a relevância de cada hiperparâmetro pode ser encontrada no repositório GitHub [GitHub, 2024]. Embora não reportado na tabela para evitar redundância, todas as instâncias dos experimentos foram realizadas com 10 dobras (k-fold = 10) e tamanho de *batch* de 256.

4.2. Utilidade (para classificadores de ML)

A Tabela 4 fornece valores p obtidos através do teste de *Mann-whitney* para quatro modelos diferentes de aprendizado de máquina – *Random Forest*, *Decision Tree*, *Perceptron* e *SGDRegressor* – aplicados aos quatro conjuntos de dados. Esses valores p indicam a significância estatística dos modelos em cada conjunto de dados. Para a

realização dos experimentos foi adotado o valor de 0,05 como limiar de p, um valor amplamente aceito e utilizado na literatura.

A nossa hipótese H0 é que não existe diferença estatisticamente significativa entre as métricas obtidas pelos classificadores com os dados sintéticos e reais. Conforme observado nos resultados, as médias dos valores de p dos classificadores foram superiores a 0,05. Portanto, conclui-se que não há evidências suficientes para rejeitar a hipótese H_0.

Tabela 4. Valor de p obtido por classificadores

Datasets	P_value			
	Random Forest	Decision Tree	Perceptron	SGD Regressor
Kronodroid	0,10	0,40	0,19	0,45
Adroit	0,23	0,07	0,05	0,10
Drebin	0,58	0,17	0,08	0,48
Android P	0,25	0,17	0,12	0,11

4.3. Fidelidade (estatística)

A Tabela 5 fornece uma comparação de duas métricas de desempenho - similaridade de cosseno e erro quadrático médio - para classificações das amostras positivas e falsas de *malware* nos quatro conjuntos de dados.

Tabela 5. Fidelidade:

Dataset	Positivo		Negativo	
	Cosseno	Erro quadrático	Cosseno	Erro quadrático
Kronodroid	0,69	0,06	0,74	0,07
Adroit	0,70	0,03	0,65	0,03
Drebin	0,37	0,12	0,50	0,16
Android P	0,22	0,03	0,51	0,03

Na similaridade de cossenos, um valor de 0 indica que os dois vetores são ortogonais e não possuem similaridade, enquanto que o valor de 1 indica um vetor idêntico. Os resultados da métrica de similaridade de cossenos nos *datasets* Kronodorid, Adroit e Drebrin indicam que dados sintéticos e reais possuem um grau elevado de similaridade, mas não são idênticos entre si. Já os resultados do *dataset* Android P, variando um pouco mais entre os diferentes classificadores (de 0,05 a 0,23) podem ser explicados por uma alta variância entre membros da mesma distribuição. Em relação aos valores de erro quadrático, os *datasets* Kronodorid, Adroit e Android P apresentam valores próximos a zero, mas não zero, o que indica alta similaridade. Vale destacar que um erro quadrático médio de zero indica uma cópia exata dos dados reais, enquanto um valor próximo de 1 pode indicar problemas na rede, incluindo colapso modal [Gonog, L. et. al., 2019], que podem indicar a geração um conjunto limitado de amostras muito similares, que impedem o aprendizado.

5. Considerações Finais

Neste trabalho, demonstramos a complexidade e a variabilidade inerentes à geração de *datasets* sintéticos de *malware* Android em diferentes conjuntos de dados usando cGANs. A otimização de hiperparâmetros é fundamental na adaptação de modelos às características específicas de cada *dataset*, conforme evidenciado pelas diversas configurações empregadas. Os resultados demonstram que *datasets* sintetizados por cGANs podem ser considerados *úteis* e *fiéis* a quatro *datasets* originais.

Como trabalhos futuros, podemos incluir a avaliação de outras métricas de *fidelidade*, *utilidade*, *privacidade* e *eficiência computacional*, como aquelas propostas recentemente na literatura [Platzer and Reutterer, 2021, Alaa, A. et. al., 2022].

Agradecimentos. A pesquisa contou com apoio parcial da RNP (Programa Hackers do Bem - GT Malware DataLab), da CAPES (Código de Financiamento 001) e da FAPERGS, por meio dos editais 08/2023 e 09/2023.

Referências

AI & Data Today (2023). Top 10 reasons why ai projects fail. `https://t.ly/wMBj5`.

Alaa, A. et. al. (2022). How faithful is your synthetic data? In *ICML*.

Allix, K. et. al. (2015). Are your training datasets yet relevant? an investigation into the importance of timeline in machine learning-based malware detection. In *ESSoS*.

Esteban, C., Hyland, S. L., and Rätsch, G. (2017). Real-valued (medical) time series generation with recurrent conditional GANs. *arXiv preprint arXiv:1706.02633*.

Fakoor, R. et. al. (2020). Fast, accurate, and simple models for tabular data via augmented distillation. *Advances in Neural Information Processing Systems*, 33:8671–8681.

Fang, J., Tang, C., Cui, Q., Zhu, F., Li, L., Zhou, J., and Zhu, W. (2022). Semi-supervised learning with data augmentation for tabular data. In *ACM CIKM*.

GitHub (2024). Malsyngen. `https://github.com/SBSegSF24/MalSynGen`.

Gonog, L. et. al. (2019). A review: generative adversarial networks. In *IEEE ICIEA*.

Lu, Y. and Li, J. (2019). Generative adversarial network for improving deep learning based malware classification. In *WSC*. IEEE.

Machado, P., Fernandes, B., and Novais, P. (2022). Benchmarking data augmentation techniques for tabular data. In *IDEAL*, page 104. Springer.

McKnight, P. E. and Najab, J. (2010). Mann-whitney u test. *The Corsini encyclopedia of psychology*, pages 1–1.

Meijin, L., Zhiyang, F., Junfeng, W., Luyu, C., Qi, Z., Tao, Y., Yinwei, W., and Jiaxuan, G. (2022). A systematic overview of android malware detection. *Applied AI*, 36(1).

Mimura, M. (2020). Using fake text vectors to improve the sensitivity of minority class for macro malware detection. *JISA*, 54:102600.

Mirza, M. and Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv 1411 1784*.

Platzer, M. and Reutterer, T. (2021). Holdout-based empirical assessment of mixed-type synthetic data. *Frontier in Big Data*.

Rajabi, A. and Garibay, O. O. (2022). Tabfairgan: Fair tabular data generation with generative adversarial networks. *Machine Learning and Knowledge Extraction*, 4(2):488.

Rocha, V., Assolin, J., Bragança, H., Kreutz, D., and Feitosa, E. (2023). AMGenerator e AMExplorer: Geração de metadados e construção de datasets android. In *XXIII SBSeg*.

Snow, D. (2020). Deltapy: A framework for tabular data augmentation in python. *Available at SSRN 3582219*.

Tanaka, F. H. K. D. S. and Aranha, C. (2019). Data augmentation using GANs. *arXiv preprint arXiv:1904.09135*.

Zhao, Z., Kunar, A., Birke, R., Van der Scheer, H., and Chen, L. Y. (2024). Ctab-gan+: Enhancing tabular data synthesis. *Frontiers in Big Data*, 6.