DroidAugmentor: uma ferramenta de treinamento e avaliação de cGANs para geração de dados sintéticos

Karina Casola[1(‡)], Kayuã Oleques Paim[1(‡)], Rodrigo Brandão Mansilha[1(‡)], Diego Kreutz[1(‡)]

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

{{karinafernandes,kayuapaim}.aluno,rodrigomansilha,diegokreutz}@unipampa.edu.br

Resumo. *A DroidAugmentor utiliza cGANs para criar dados sintéticos rotulados como benignos ou malignos. A ferramenta incorpora métricas de similaridade e aplicabilidade para avaliar a qualidade desses dados. Demonstramos a eficácia da DroidAugmentor na geração de 12.126 amostras sintéticas para o dataset Drebin-215, indicando a capacidade de capturar padrões representativos para o aumento de dados.*

1. Introdução

Estatísticas recentes apontam que cerca de 80% dos projetos de Inteligência Artificial (IA) enfrentam falhas devido à falta de dados quantitativos para treinamento e construção de modelos realistas e eficazes [AI & Data Today, 2023]. O cenário é similar no contexto da detecção de *malwares* Android, pois faltam conjuntos de dados suficientemente representativos e de qualidade [Vilanova et al., 2022].

Um desafio adicional na detecção de *malwares* Android é a obsolescência dos *datasets* disponíveis [Soares et al., 2021a, Soares et al., 2021b]. A criação de *datasets* representativos (e.g., 100.000 ou mais amostras rotuladas) e atualizados pode levar meses. Este é possivelmente um dos principais fatores que leva a maioria das pesquisas a utilizar *datasets* pequenos, pouco representativos e defasados para treinar e validar modelos preditivos para detecção de *malwares* Android. Por exemplo, a API do VirusTotal[1], o serviço de rotulação mais completo e frequentemente utilizado [Vilanova et al., 2022], na versão gratuita, permite aproximadamente 250 rotulações atualizadas por dia, ou seja, seriam necessários em torno de 400 dias para rotular adequadamente 100.000 amostras. Este processo longo e demorado leva ao problema adicional da defasagem dos dados. Enquanto as últimas amostras analisadas estarão atualizadas, as primeiras estarão com uma defasagem superior a 12 meses.

Uma estratégia para enfrentar o desafio de construir conjuntos de dados representativos e atualizados é a aplicação de técnicas de aumento de dados. Abordagens que empregam modelos generativos, como as Redes Generativas Adversariais (GANs), têm sido estudadas para esse propósito, especialmente na área de visão computacional [Cap et al., 2020]. Essas metodologias viabilizam a criação de instâncias sintéticas que preservam os traços fundamentais do conjunto de dados original, ao

(‡) Todos os autores contribuíram igualmente para o desenvolvimento do trabalho.

[1]https://developers.virustotal.com/reference/overview

mesmo tempo, em que fomentam a diversificação das amostras, sem demandar a aquisição considerável de novos dados.

Mais recentemente, GANs começaram a ser utilizadas para a criação de vetores de características de aplicativos Android classificados como *malwares* [Renjith et al., 2022]. O propósito subjacente a essa estratégia é aprimorar a capacidade de evasão desses programas em relação às medidas de detecção. Embora a estratégia proposta possa gerar novas amostras a partir das existentes, ela não mantém as características mais relevantes do dado que permitem distinguir entre uma amostra benigna e maligna.

Diferentemente dos trabalhos existentes, apresentamos a ferramenta DroidAugmentor, uma solução para expandir rapidamente e de maneira prática a quantidade de dados em um conjunto de dados limitado. Essa ampliação é alcançada por meio do treinamento parametrizável de uma *Conditional Generative Adversarial Network* (cGAN) [Mirza and Osindero, 2014], permitindo a geração condicional de dados com suas respectivas rotulações. A DroidAugmentor é composta por módulos e funções utilitárias que facilitam a personalização das configurações do experimento, tornando-a altamente adaptável para atender a diversos requisitos específicos. Conforme demonstramos na avaliação, a DroidAugmentor possui a capacidade de aumentar o tamanho de um *dataset* real, como o Drebin-215, em questão de poucas horas ao invés de semanas ou meses.

Para avaliar a qualidade (ou similaridade) e aplicabilidade dos dados sintéticos gerados, a ferramenta incorpora dois tipos de métricas. Num primeiro estágio, as métricas de Erro Quadrático Médio, Divergência Kullback-Leibler (KL) e Máxima Discrepância Média para avaliar a similaridade entre os dados reais e sintéticos. Num segundo estágio, as métricas de Acurácia, Precisão, Recall e F1-Score para avaliar a aplicabilidade dos dados. É importante destacarmos que a ferramenta gera a matriz de confusão e as curvas de perda de treinamento para todas as estratificações, o que permite o acompanhamento do processo de treinamento da cGAN e a identificação de possíveis anomalias (e.g., colapso ou não convergência da rede).

Iniciamos com a arquitetura e a implementação da DroidAugmentor na Seção 2. Na sequência, Seções 3 e 4, apresentamos a avaliação e as considerações finais, incluindo detalhes da demonstração.

2. DroidAugmentor

A Figura 1 resume a DroidAugmentor. Dado um *dataset* e um conjunto de parâmetros de entrada, a DroidAugmentor executa uma série de passos até gerar um *dataset* sintético e um conjunto de valores para métricas de desempenho e gráficos que permitem analisar a qualidade do *dataset*. Nas seções a seguir, discutimos o fluxo de execução e, em seguida, os principais aspectos de implementação da DroidAugmentor.

2.1. Fluxo de Execução

A **Entrada** é composta por parâmetros de execução e o conjunto de dados reais utilizado no treinamento da cGAN. Os parâmetros incluem configurações relacio-

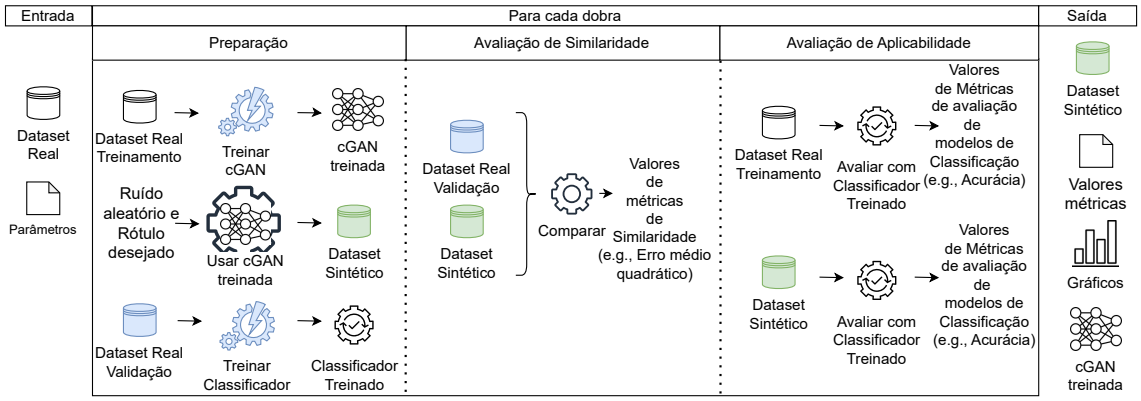


Figura 1. Entrada, saída e fluxo de execução da DroidAugmentor

nadas ao cenário de avaliação, cGAN e treinamento. Como exemplos do primeiro caso, destacamos o número de amostras benignas e malignas, número de dobras e classificadores utilizados. Em relação à cGAN, é possível configurar aspectos como número e densidade de camadas. Em relação ao treinamento da cGAN, é possível especificar itens como o número de épocas e o otimizador. O conjunto de dados reais pode ser visto como uma tabela na qual cada linha representa uma amostra. Uma coluna contém o rótulo da classe, onde cada amostra é rotulada como maligna ou benigna, e todas as outras colunas contêm características da amostra, como permissão de acesso à câmera.

A **Saída** da ferramenta inclui o *dataset* sintético, as cGANs treinadas (uma para cada dobra), dados brutos e gráficos relacionadas a métricas de desempenho, como matrizes de confusão e curvas de perda de treinamento. O *dataset* de saída segue a mesma organização do *dataset* de entrada e deve servir para melhorar o treinamento de ferramentas de predição de *malwares* em aplicativos android. A cGAN treinada serve para gerar *datasets* sem precisar repetir o fluxo de treinamento, que é tipicamente a etapa mais demorada. Os dados e gráficos resultantes permitem entender a qualidade do *dataset* gerado e facilitar o desenvolvimento de cGANs de maneira sistemática.

O Fluxo de execução é composto por uma etapa de preparação e duas fases de avaliação. O fluxo é executado de modo a permitir validação cruzada. Por exemplo, quando o número de dobras é 5, temos, em cada iteração, um conjunto distinto de 80% dos dados usados para treinamento e os 20% restantes empregados para avaliação. Dessa forma, para cada dobra ocorre a preparação e avaliação dos artefatos gerados.

Na **Preparação**, primeiro ocorre a instanciação e treinamento da cGAN com (parte) dos dados reais. Em seguida, a cGAN treinada é utilizada para gerar dados sintéticos de determinado rótulo (i.e., benigno ou maligno) a partir de ruído aleatório. Por fim, os classificadores de interesse são instanciados e treinados usando dados reais.

A avaliação compreende duas etapas, que correspondem às análises de similaridade e aplicabilidade. A **Avaliação de Similaridade** gera valores para métricas como Erro Quadrático Médio entre dados sintéticos e reais. Essas métricas permi-

tem verificar se os dados gerados são diferentes dos dados originais e, ao mesmo tempo, seguem mesmo padrão estatístico. A **Avaliação de Aplicabilidade** consiste na aplicação de dados sintéticos e dados reais em classificadores. Ao verificar se os classificadores são capazes de classificar os dados sintéticos de maneira similar aos dados reais, pode-se inferir que os dados sintéticos são plausivelmente realistas e adequados para o treinamento. Nessa fase é possível utilizar métricas comuns em classificadores binários, como a acurácia e a precisão.

2.2. Implementação

A DroidAugmentor é organizada em um programa principal e dois módulos (`Models` e `Tools`). O primeiro módulo contém a classe `ConditionalGANModel`, responsável por configurar e criar GANs condicionais de forma genérica, o `AdversarialModel` que abrange o modelo adversarial generativo, e `Classifiers`, que implementa diferentes classificadores e facilita a extensibilidade e manutenibilidade da ferramenta. O módulo `Tools` inclui componentes responsáveis por tarefas como persistência e visualização dos valores das métricas de desempenho. O programa principal (`Main`) processa os parâmetros de entrada, fluxo de execução e organiza dados de saída. Adicionalmente, incluímos um programa auxiliar para automação das baterias de experimentos. Atualmente a ferramenta conta com um modelo adversarial, 5 classificadores, 3 métricas de aplicabilidade, 4 métricas de avaliação de classificação e 3 tipos de gráficos. O código-fonte e instruções instalação e execução estão disponíveis no GitHub[2].

3. Avaliação

O objetivo da avaliação é demonstrar como a ferramenta DroidAugmentor pode gerar *datasets* sintéticos para detecção de *malwares* Android analisando sua qualidade. Nesse sentido, são necessários meios para verificar (i) se o *dataset* sintético preserva as características essenciais do *dataset* original, ampliando-o e diversificando-o de forma efetiva; e (ii) se os dados sintéticos conseguem capturar os padrões relevantes dos dados reais, através da análise dos dados com classificadores.

A Tabela 1 resume os valores utilizados na demonstração. A avaliação foi realizada através da validação cruzada com 5 dobras (k-fold = 5), utilizando 80% para treinamento e 20% para avaliação.

Nos concentramos no *dataset* Debrin-215 que é frequentemente empregado em pesquisas relacionadas à detecção de *malware* Android [Soares et al., 2021a, Soares et al., 2021b]. Ele possui 215 características, 5.560 amostras malignas e 6.566 amostras benignas. Usando esse *dataset*, geramos 12.126 amostras sintéticas, o que corresponde a um aumento de 100% das amostras, obtidas de maneira distribuída em cinco estratificações com 20% de amostras em cada uma. Os resultados principais são discutidos a seguir. Os resultados completos podem ser encontrados no repositório da ferramenta.

Na análise de similaridade, empregamos as métricas Divergência KL, Máxima Discrepância Média e Erro Médio Quadrático. Para as três topologias consideradas,

[2] `https://github.com/LEA-SF23/DroidAugmentor`

Tabela 1. Relação de parâmetros e valores.

	Parâmetro	Valores
Cenário	Número de dobras	5
	Dataset real	Debrin-215
	Classificador	{Random Forest, KNN, Decision Tree}
Treinamento	Otimizador	*Adam*[Kingma and Ba, 2014]
	Coeficiente β^1 G, D	0.5
	Coeficiente β^2 G, D	0.9
	Número de épocas	1.000
cGAN	Número de camadas do Gerador/Discriminador	2
	Número de neurônios por camada	{256, 1024, 4096}
	Funções de ativação intermediárias	LeakyReLU
	Função de ativação de saída	*Sigmoid*
	Tamanho de lote	32
	Função de perda	Entropia cruzada binária

os valores foram similares para as três métricas, em torno de 0,16. Esses resultados demonstraram que os dados sintéticos gerados pelas cGANs treinadas com Drebin-215 são estatisticamente similares aos dados reais em termos de distribuição e características. Especificamente, as métricas de Divergência KL (0,15) e a Máxima Discrepância Média (0,16) indicam uma proximidade considerável entre as distribuições dos dados sintéticos e reais. Observamos que o Erro Médio Quadrático (0,16), mesmo sendo maior que zero, é coerente e esperado, pois indica que os dados gerados não são idênticos aos dados reais.

Cabe destacar que, em alguns casos, a métrica de Erro Médio Quadrático pode indicar uma cópia exata dos dados reais quando seu valor for 0,0. Já um valor próximo de 1,0 pode indicar problemas na rede, incluindo colapso modal [Gonog and Zhou, 2019], que podem levar a geração de dados totalmente iguais aos reais. No contexto dos experimentos com a DroidAugmentor, um colapso modal pode ser atribuído a duas situações hipotéticas. Primeiramente, o colapso modal pode estar associado à configuração da rede, incluindo número de épocas, camadas e outros parâmetros. Nesse caso, a ferramenta oferece a possibilidade de automatizar a execução de várias baterias de testes para avaliar diferentes configurações da rede. Por outro lado, a segunda hipótese se refere aos dados de entrada que podem apresentar uma baixa variabilidade, resultando em dificuldades para a rede neural gerar dados sintéticos de alta qualidade.

Em testes iniciais, foram identificadas condições de não convergência e colapso modal durante o processo de treinamento. Por exemplo, a fim de garantir a estabilidade do processo de treinamento realizado por meio do algoritmo estocástico de otimização do gradiente Adam [Kingma and Ba, 2014], foi necessário realizar modificações nos parâmetros de taxa de aprendizado e de decaimento exponencial das estimativas de primeira e segunda ordem.

No caso do Drebin-215, em vez de empregar os valores convencionais de 10^{-3} para a taxa de aprendizado e 0,9 e 0,999 para as estimativas de primeira e segunda ordem, respectivamente, optamos por utilizar valores diferentes: 10^{-4} para a taxa de aprendizado, 0,5 para a estimativa de primeira ordem e 0,9 para a estimativa de segunda ordem. Esses valores foram selecionados com base em uma abordagem empírica, seguindo o critério de maior estabilidade e velocidade de convergência.

Embora os parâmetros selecionados contribuam para uma maior estabilidade durante o processo de treinamento, é importante destacar que outros desafios podem surgir. Por exemplo, é possível que situações de não convergência ocorram ao longo do treinamento, resultando em obtenção de resultados significativamente aquém das expectativas. Essa situação pode ser identificada em primeira mão nos eventuais resultados anormais dos classificadores, cuja causa mais comum é a geração de dados aleatórios. Alguns casos de não convergência em redes GAN podem ser eventualmente confirmados também através da observação do monitoramento das curvas de perda. As curvas de perda devem diminuir e estabilizar com o tempo. Se as curvas se desviarem muito, pode ser um sinal de não convergência. A ferramenta DroidAugmentor gera as curvas de perda para cada estratificação, permitindo ao usuário identificar visualmente desvios e oscilações que podem eventualmente indicar a não convergência.

Para avaliar a aplicabilidade dos dados gerados, consideramos três classificadores clássicos: Random Forest (RF), Decision Tree (DT) e K-Nearest-Neighbor (KNN). Para cada dobra, aplicamos cada um dos classificadores no *dataset* real e no *dataset* sintético criado com cada uma das topologias cGAN. Com base nos resultados, geramos matrizes de confusão e extraímos as seguintes métricas, populares no domínio de aprendizado de máquina: Acurácia, Precisão, Recall e F1-Score.

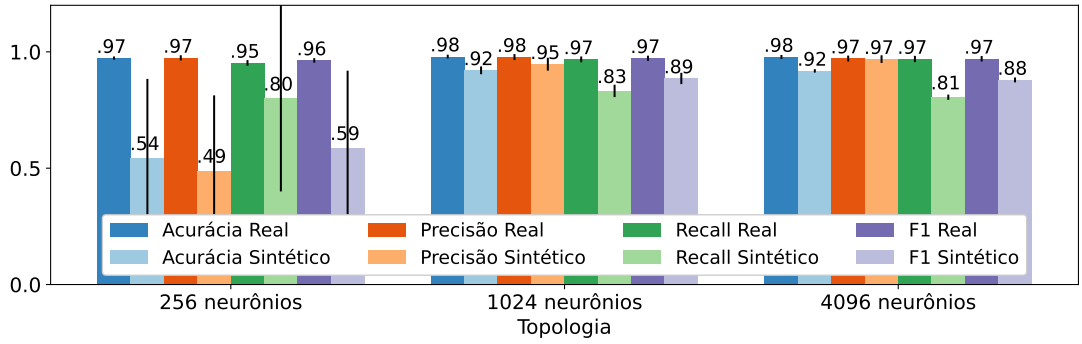


Figura 2. Resultados com 3 topologias e classificador KNN.

A Figura 2 mostra o impacto da topologia nas métricas de desempenho do classificador KNN. Para cada topologia são apresentados 8 barras, sendo uma para cada combinação entre métricas e tipos de *dataset* (real e sintético). É importante observar que os dados sintéticos obtidos com dados reais podem variar para diferentes topologias, pois os algoritmos foram treinados e executados múltiplas vezes. Em geral, observa-se que o número de neurônios pode impactar significativamente na cGAN com 2 camadas. No restante da avaliação, focamos nos resultados obtidos com 1024 neurônios por camada.

A Figura 3 mostra o desempenho da cGAN com 1.024 neurônios por camada considerando múltiplos classificadores. A exemplo do gráfico anterior, para cada classificador são apresentados 8 barras, sendo uma para cada combinação entre métricas e tipos de dados (real e sintético).

Os resultados indicam convergência e ratificam os resultados das métricas de

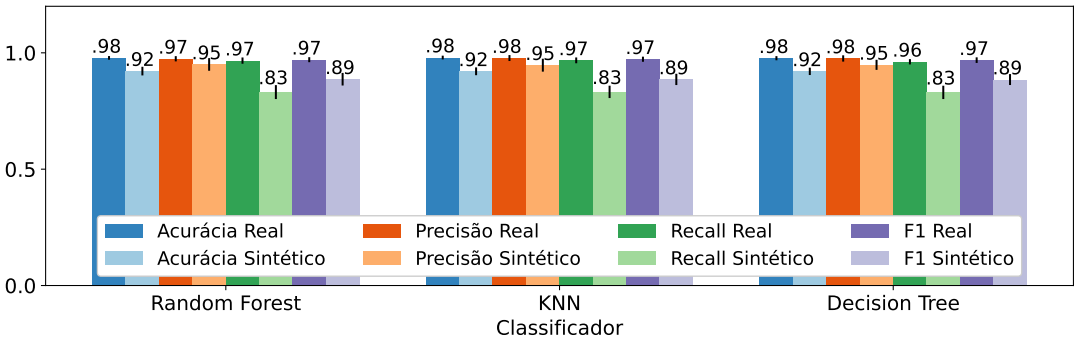
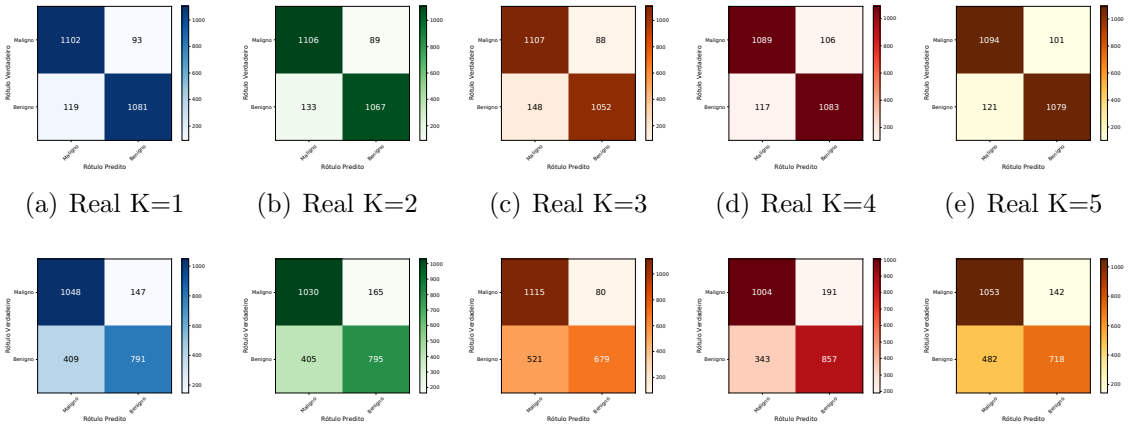


Figura 3. Resultados com 3 classificadores e cGAN com 1024 neurônios por camada.

similaridade. Todos os classificadores considerados apresentam todas as métricas acima de 80% para os *datasets* real e sintético, resultados estes que podem ser considerados bons.

Na Figura 4 apresentamos as matrizes de confusão obtidas para o classificador KNN em cada dobra com *dataset* real (linha superior) e *dataset* sintético gerado usando cGAN com 1024 neurônios (linha inferior). Em linhas gerais, o contraste entre a diagonal principal mais escura e a diagonal inversa, mais clara, permitem concluir que o classificador KNN funcionou muito bem para todos as dobras do *dataset* real. Esse resultado corrobora as altas médias e baixo desvios padrão do gráfico anterior. Nessa mesma direção, os dados sintéticos também são bons, embora haja espaço para melhoria, especialmente em termos de falso positivo. Outro aspecto a destacar é o fato de as matrizes de confusão indicarem que não há vícios (viés) na predição.



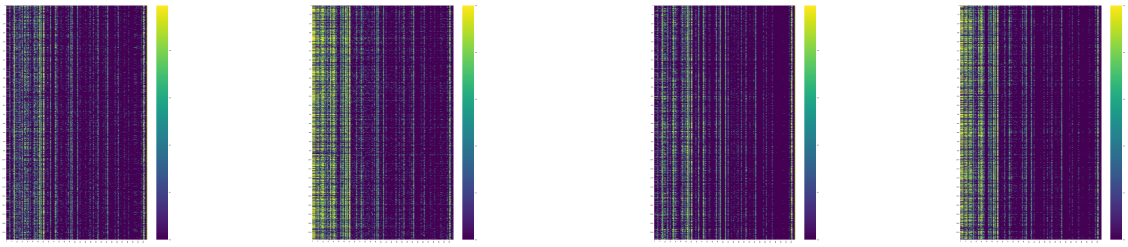
(a) Real K=1 (b) Real K=2 (c) Real K=3 (d) Real K=4 (e) Real K=5

(f) Sint. K=1 (g) Sint. K=2 (h) Sint. K=3 (i) Sint. K=4 (j) Sint. K=5

Figura 4. Matrizes de confusão da cGAN 1024 neurônios e classificador KNN.

Finalmente, na Figura 5 apresentamos o mapa de calor dos dados reais (Real) e sintéticos (Sint.). Os gráficos demonstram que os dados sintéticos incorporaram padrões similares aos dados reais, tanto para amostras classificadas como malignas (M.) quanto para aquelas classificadas como benignas (B.). No GitHub é possível

visualizar os detalhes das *features*.



(a) Real M. (b) Real B. (c) Sint. M. (d) Sint. B.

Figura 5. Mapa de calor das *features* dos dados

4. Considerações Finais

A DroidAugmentor é a primeira ferramenta prática e livremente disponível para o treinamento e validação de cGANs para a geração de dados sintéticos para *datasets* reais de uma dimensão, como os frequentemente utilizados no contexto da detecção de *malwares* Android. Além de a ferramenta permitir ao usuário automatizar e agilizar o teste de parâmetros da rede para conjuntos de dados reais diversos, os resultados apresentados indicam que ela é capaz de gerar dados sintéticos de qualidade a partir de dados reais frequentemente utilizados para desenvolver classificadores de *malwares* Android, como no Drebin-215.

Agradecimentos. O trabalho foi apoiado pela CAPES – Código de Financiamento 001.

Referências

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

Cap, Q. H., Uga, H., Kagiwada, S., and Iyatomi, H. (2020). Leafgan: An effective data augmentation method for practical plant disease diagnosis. *IEEE TASE*, 19(2):1258.

Gonog, L. and Zhou, Y. (2019). A review: generative adversarial networks. In *14th IEEE ICIEA*, pages 505–510. IEEE.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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

Renjith, G., Laudanna, S., Aji, S., Visaggio, C. A., and Vinod, P. (2022). GANG-MAM: GAN based engine for modifying android malware. *SoftwareX*, 18:100977.

Soares, T., Mello, J., Barcellos, L., Sayyed, R., Siqueira, G., Casola, K., Costa, E., Gustavo, N., Feitosa, E., and Kreutz, D. (2021a). Detecção de malwares android: Levantamento empírico da disponibilidade e da atualização das fontes de dados. In *Anais da XIX ERRC*, pages 49–54. SBC.

Soares, T., Siqueira, G., Barcellos, L., Sayyed, R., Vargas, L., Rodrigues, G., Assolin, J., Pontes, J., Feitosa, E., and Kreutz, D. (2021b). Detecçao de malwares android: datasets e reprodutibilidade. In *Anais da XIX ERRC*, pages 43–48. SBC.

Vilanova, L., Kreutz, D., Assolin, J., Quincozes, V., Miers, C., Mansilha, R., and Feitosa, E. (2022). Adbuilder: uma ferramenta de construçao de datasets para detecçao de malwares android. In *Anais Estendidos do XXII SBSeg*, pages 143–150. SBC.