Geração de Dados de Ataque em Internet das Coisas utilizando Redes Generativas Adversárias

Iran F. Ribeiro[1], **Guilherme S. G. Brotto**[1], **Giovanni Comarela**[1], **Vinícius F. S. Mota**[1]

[1]Departamento de Informática – Universidade Federal do Espírito Santo
Vitória – Brasil

{iran.ribeiro, guilherme.brotto}@edu.ufes.br, {gc, vinicius.mota}@inf.ufes.br

Abstract. *Analyzing the data generated by gadgets is crucial for spotting and minimizing cyberattacks on the Internet of Things. However, public data representing real attacks still tends to be scarce. To increase data availability, this work presents a study on the use of Generative Adversarial Networks (GANs) to generate synthetic attack data on IoT devices with high fidelity to real data, i.e., with similar characteristics. Simultaneously, ensuring privacy and that the utility of synthetic data in machine learning tasks is similar to real data. For this purpose, two GAN models, CTGAN and NetShare, were compared using a dataset containing normal traffic and attacks on IoT devices. The results indicate that both GAN models are efficient in generating synthetic data, both in fidelity and quality. However, CTGAN proves to be the most efficient model, considering execution time and memory consumption.*

Resumo. *A análise de tráfego de dados gerados por dispositivos é fundamental para detecção e mitigação de ataques na Internet das Coisas. Contudo, dados públicos que representem ataques reais ainda são escassos. Visando aumentar a disponibilidade de dados, este trabalho apresenta um estudo do uso de Redes Generativas Adversárias (GANs) para gerar dados sintéticos de ataque em dispositivos IoT com alta fidelidade em relação aos dados reais, isto é, com características similares. Ao mesmo tempo visa garantir privacidade e que a utilidade dos dados sintéticos em tarefas de aprendizado de máquina sejam similares aos reais. Para isso, foram comparamos dois modelos de GANs, CTGAN e NetShare, utilizando como base um conjunto de dados contendo tráfego normal e com ataques em dispositivos IoT. Os resultados indicam que ambos os modelos de GANs são eficientes na geração de dados sintéticos, tanto em fidelidade quanto em qualidade. Entretanto, a CTGAN apresenta-se como o modelo mais eficiente, considerando tempo de execução e consumo de memória.*

1. Introdução

O acesso a dados de tráfego de dados em redes é fundamental para o desenvolvimento e propostas de novos serviços, tais como, monitoramento de qualidade do serviço [Pundir et al. 2021], novos algoritmos de roteamento [Gheisari et al. 2020], e detecção de ataques e anomalias [Hossain et al. 2020]. No entanto, os dados reais de rede, tais como fluxos de redes e capturas de pacotes, podem conter informações sensíveis tanto da organização que detém os dados quanto de usuários [Sharafaldin et al. 2018, Aleroud et al. 2021].

A situação torna-se mais complexa quando se considera dados provenientes dos mais variados dispositivos de rede, na chamada Internet das Coisas (do inglês, *Internet of Things*-IoT). A análise de dados de tráfego gerado por dispositivos IoT permite a detecção prévia de ataques na rede que utilizam tais dispositivos. Embora existam diversos conjuntos de dados (*datasets*) de ataques no cenário IoT, a maioria é coletada em cenários simulados [Alex et al. 2023], limitando o nível de confiança em suas representatividades. *Datasets* reais, fornecidos por empresas e organizações, comumente utilizam técnicas de anonimização, mas disponibilizam uma quantidade limitada de dados. Isto dificulta que algoritmos de aprendizado de máquina gerem modelos com alta confiabilidade.

Uma abordagem para aumentar a disponibilidade de dados para pesquisadores é a utilização de algoritmos de aprendizado profundo para geração de dados sintéticos que reproduzam as principais características dos dados reais. Uma das principais técnicas para geração de dados sintéticos são as Redes Generativas Adversárias (*Generative Adversarial Networks* - GANS) [Goodfellow et al. 2014] e os *Autoenconders* Variacionais (*Variational Autoenconders* - VAEs) [Kingma and Welling 2013]. Esses modelos são construídos a partir de arquiteturas de aprendizado profundo, que conseguem aprender as principais características de um conjunto de dados e prover um nível aceitável de privacidade aos dados gerados. As GANs, especificamente, são mais conhecidas pelo seu uso na área de visão computacional e para geração de imagens [Wang et al. 2021], mas têm aplicações em séries temporais e dados tabulares [Brophy et al. 2023].

A utilização dessa alternativa, entretanto, precisa ser feita de maneira a manter a fidelidade e utilidade dos dados sintéticos enquanto garante a privacidade dos dados anonimizados [Aleroud et al. 2021]. A fidelidade dos dados sintéticos é mensurada ao comparar as distribuições entre os dados sintéticos e dos dados reais. Além disso, é necessário que os dados sintéticos mantenham correlações semelhantes aos dados reais entre os dados e o tempo. Por exemplo, se os dados reais apresentam picos ou ciclos, os dados sintéticos também devem apresentar tal comportamento. Por outro lado, a utilidade mensura se os dados sintéticos produzem resultados semelhantes aos dados reais em algoritmos de aprendizado. Por exemplo, um modelo de aprendizado de máquina treinado com dados sintéticos deve ter bons resultados quando testados com os dados reais.

Assim, neste trabalho apresentamos uma avaliação de fidelidade e utilidade de dados sintéticos de ataque em dispositivos IoT gerado por GANs. Como principal contribuição, apresentamos uma alternativa poderosa para auxiliar na disponibilização de dados reais de ataque em dispositivos IoT. Espera-se, nesse sentido, que detentores de dados sensíveis de ataques IoT possam compartilhar dados sintéticos ou até mesmo os modelos generativos. Desta forma, a comunidade científica pode trabalhar para melhorar algoritmos e sistemas de detecção e mitigação de ataques em dispositivos IoT. Para isto, utilizamos um *dataset* recente e aberto que contém diversos ataques a dispositivos IoT e dois modelos do estado da arte para geração de dados categóricos: CTGAN [Xu et al. 2019] e NetShare [Yin et al. 2022]. As demais contribuições deste artigo são sumarizadas a seguir:

- Apresenta um estudo comparativo exaustivo dos principais modelos GANs para dados tabulares. Para isto, comparamos a influência de hiperparâmetros em cada modelo generativo e como isso pode impactar a utilidade dos *datasets* sintéticos.
- É proposto uma métrica para avaliar a utilidade dos dados sintéticos em relação aos dados reais.

- Propõe uma avaliação da fidelidade dos modelos generativos em gerar dados de ataque de qualidade e variabilidade, possibilitando uma maior escalabilidade e generalização de algoritmos para classificação/identificação de ataques.

O restante desse artigo está organizado como segue. A Seção 2 descreve as GANs e métricas utilizadas neste trabalho. Na Seção 3 discutimos trabalhos relacionados. A metodologia e *datasets* utilizados no artigo são descritos na Seção 4. Na Seção 5 apresentamos os resultados dos experimentos realizados. Por fim, a Seção 6 conclui este artigo e discute trabalhos futuros.

2. Referencial Teórico

Esta seção descreve as características gerais de uma GAN e dois modelos que serão comparados: CTGAN e NetShare. Em seguida, são descritas as técnicas comumente utilizadas na literatura para avaliação da fidelidade e utilidade de *datasets* sintéticos.

2.1. Redes generativas adversárias

A GAN é um *framework* proposto por [Goodfellow et al. 2014] para otimizar o treino de modelos generativos. Por meio de um jogo de min-max, duas redes neurais competem entre si, sendo treinadas simultaneamente: um Gerador (G), um *Multilayer Perceptron* (MLP) que gera dados falsos baseados em entradas aleatórias, e um Discriminador (D), outro MLP que classifica os dados gerados, considerando um *dataset* real. O objetivo de G é gerar amostras cujas distribuições são tão próximas da distribuição dos dados reais, que D não consegue distinguí-las. Apesar de ter sido inicialmente desenvolvido para utilização no campo de visão computacional [Karras et al. 2017, Brock et al. 2018], as aplicações de GANs podem ser encontradas em áreas como geração de dados em medicina, astronomia e sensoriamento remoto [Dash et al. 2023].

Para geração de dados tabulares, a arquitetura original das GANs encontra alguns desafios. Geralmente, dados tabulares são compostos por dados numéricos e categóricos, onde as distribuições não são gaussianas (como ocorre com imagens). Durante o processamento para treinamento, os dados categóricos podem tornar-se matrizes esparsas (como ocorre com a técnica de *one-hot enconding*), dificultando o aprendizado do modelo. Por fim, a presença de colunas (*features*) desbalanceadas impõe desafios adicionas às GANs, pois a distribuição das classes com menos frequência tendem a não ser aprendidas corretamente.

Nesse contexto, destaca-se dois modelos para geração de dados tabulares, utilizados neste trabalho. A CTGAN foi proposta por [Xu et al. 2019] para a geração de dados tabulares, sendo um dos primeiros modelos a obter resultados expressivos na área. Em contrapartida, o NetShare [Yin et al. 2022], é o modelo GAN do estado arte para geração de dados tabulares e sequenciais, por exemplo, séries temporais de fluxos de rede, onde os dados podem ser dependentes do tempo. Assim, neste trabalhos considera-se a CTGAN como *baseline* e o NetShare o modelo que, espera-se, possua melhor desempenho que a *baseline*.

2.2. Métricas de avaliação

A **fidelidade** de um conjunto de dados sintético pode ser medida pela distância ou divergência de sua distribuição de probabilidade do conjunto de dados real. Destacamos três

técnicas de cálculo de divergência entre distribuições de probabilidade para avaliar GANs de séries temporais[1]:

2.2.1. Divergência de Kullback-Leibler (KL-d)

É um tipo de divergência que mede como uma distribuição de probabilidade P difere de uma segunda distribuição de probabilidade de referência Q. A Equação (1) mostra a definição da divergência de KL para o caso de duas distribuições, P e Q, de variáveis aleatórias discretas que assumem valores em um mesmo conjunto $\mathcal{X}$. Assim, quando P e Q são idênticas, tem-se KL-d$(P \parallel Q) = 0$, e à medida que P e Q se distanciam, o valor de KL-d$(P \parallel Q)$ aumenta.

$$\text{KL-d}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log \left(\frac{P(x)}{Q(x)} \right). \tag{1}$$

2.2.2. Distância de Jensen-Shannon (JS-d)

É uma métrica baseada em KL-d que compara duas distribuições, P e Q. JS-d usa KL-d para produzir uma pontuação simétrica, ou seja, JS-d$(P \parallel Q)$ é igual a JS-d$(Q \parallel P)$. JS-d é definida da seguinte forma:

$$\text{JS-d}(P \parallel Q) = \sqrt{\frac{\text{KL-d}(P \parallel M) + \text{KL-d}(Q \parallel M)}{2}}, \tag{2}$$

onde $M = \frac{P+Q}{2}$, representando a mistura de P e Q, e KL-d é definida na Equação (1).

2.2.3. Distância de Wasserstein-1 (W_1)

É uma distância que calcula o custo para transformar uma distribuição P em uma distribuição Q. Neste caso, considerando que P e Q são as distribuições dos dados reais e sintéticos respectivamente, quanto mais próxima de zero estiver a distância, mais os dados sintéticos são semelhantes aos reais. A definição formal de $W_1(P, Q)$ foge ao escopo desse trabalho. No entanto, a definição formal da distância e também uma variação de GAN que usa W_1 como função de perda, chamada WGAN, podem ser encontradas em [Arjovsky et al. 2017].

2.2.4. Utilidade dos dados sintéticos

A **utilidade** de um conjunto de dados sintéticos tem sido comumente medida por uma métrica *Train on Synthetic - Test on Real* (TSTR), o que significa treinar um modelo de rede neural com os dados sintéticos e testar em dados reais. Por exemplo, um classificador é treinado usando um conjunto de dados sintético, posteriormente testado em dados reais. Supondo que o classificador alcance resultados satisfatórios quando comparado com o

[1]Os leitores interessados devem consultar [Cunha et al. 2022] e [Borji 2022] para uma pesquisa mais abrangente de distância e probabilidade de divergência entre pontuações de distribuições.

mesmo classificador treinado com os dados reais, tem-se que o modelo GAN aprende as características dos dados e os dados sintéticos podem ser utilizados para pesquisa e aplicação prática como os reais.

Neste trabalho, foram utilizados algoritmos de classificação clássicos da literatura: *Decision Tree* (DT), *Random Forest* (RF), *Gradient Boost* (GB), M*ultilayer Perceptron* (MLP) e *Logistic Regression* (LR). O objetivo é mensurar a acurácia, precisão, *recall* e *F1-Score* para o problema de classificação de tráfego, comparando os resultados entre os dados reais e os dados sintéticos para cada modelo avaliado.

3. Trabalhos Relacionados

A utilização de GANs para geração de dados tem sido cada vez mais comum, principalmente no campo de visão computacional [Goodfellow et al. 2014]. Mais recentemente, alguns autores utilizaram GANs para geração de dados no cenário de redes (como tráfego e telemetria). Embora esses dados sejam similares ao tráfego em IoT, este último possui particularidades que podem não ser observadas em cenários genéricos de rede. Por isso, a quantidade de trabalhos relevantes que tratam da geração de *datasets* no cenário IoT ainda é relativamente pequena.

Um dos primeiros trabalhos focados na geração de dados IoT foi o de [Shahid et al. 2020], onde os autores propõe a geração de sequências de tamanhos de pacotes que correspondam a tráfegos bidirecionais em um dispositivo IoT real. Para isso, utilizam dados coletados de um dispositivo Google Home Mini, e os modelos Wasserstein-GAN (WGAN) clássico e algumas variações durante o treinamento. Na comparação entre as características, os autores utilizam uma VAE como base e verificam que, no geral, os dois modelos propostos são superiores ao VAE, com o modelo WGAN-C (versão da WGAN com *weight clipping*) apresentando os melhores resultados. Por fim, demonstra-se que a GAN consegue gerar dados sintéticos maliciosos que enganam algoritmos de detecção de intrusão em mais de 90% das vezes.

Considerando as restrições da área de Internet das Coisas Industriais (*Industrial Internet of Things* - IIoT), como alta acurácia, baixa latência e confiabilidade, [Qian et al. 2022] investigam a utilização de GANs para aumentar a quantidade de dados disponíveis para o treinamento de modelos de classificação em contextos de IIoT. Para isso utilizam o *dataset* T-LESS, que contém imagens de componentes IIoT. Os resultados indicam que, de fato, menos dados disponíveis impactam significativamente a acurácia de algoritmos de classificação nos cenários estudados. Além disso, os autores mostram ser possível evitar ataques de envenenamento de dados ao se utilizar o discriminador de um DCGAN previamente treinada para gerar dados de componentes industriais.

Similarmente, considerando que dados IoT podem ter suas particularidades em diferentes cenários, [Nekvi et al. 2023] utilizam GANs para gerar dados IoT no contexto de *Smart Homes*. Para isso, GANs originais com diferentes configurações são treinadas utilizando-se um *dataset* rotulado, que foi capturado em um ambiente simulado de *smart home*. Os modelos são treinados considerando 4 épocas diferentes em que, a cada 10 épocas, são gerados um *dataset* sintético. A qualidade dos *datasets* sintéticos é avaliada utilizando o TSTR a partir a acurácia de 4 classificadores conhecidos (*Logist Regression*, *Naive Bayes*, redes neurais e *Support Vector Machines*). Nesse caso, um *dataset* terá "boa qualidade" quando a acurácia dos classificadores for maior que 90%. No geral, os modelos

apresentam resultados similares, independente do *batch size* utilizado, sendo que no início e final do treinamento dos modelos os resultados tendem a ser um pouco inferiores.

Um dos grandes problemas para aprendizado de máquina ocorre quando a proporção entre as classes de *dataset* são desiguais. Nesse contexto, [Kumar and Sinha 2023] propõe a utilização de GANs como uma forma reduzir o problema de desbalanceamento de classes *datasets* de ataques. Especificamente, o objetivo é melhorar o desempenho de algoritmos de detecção de intrusão. Para isso, os autores utilizam três *datasets* de dispositivos IoT contendo tráfego normal e com ataques (NSL-KDD, UNSW-NB15 e BoY-IoT). Em relação à geração dos *datasets* sintéticos, os autores comparam modelos originais da GAN e WGAN, em conjunto com suas variantes condicionais, utilizando as métricas *precision*, *recall*, *f1-score* e *false alarm rate*. Os resultados indicam que a abordagem proposta melhora significativamente o desempenho de diferentes algoritmos de classificação.

Diferentemente da literatura, este trabalho apresenta um estudo exaustivo comparativo entre modelos generativos de dados tabulares de ataques em IoT. Nesse contexto, avaliamos a fidelidade e qualidade de *datasets* sintéticos de ataque em dispositivos IoT. Para isso comparamos o desempenho dos modelos CTGAN e NetShare considerando tempo de treinamento, geração dos dados, memória utilizada, e a influência dos hiper-parâmetros nos resultados obtidos. Além disso, propomos uma métrica para avaliar a utilidade dos dados sintéticos em relação aos reais.

4. Metodologia para geração de *datasets* sintéticos

Esta seção apresenta a metodologia para a geração de dados sintéticos. Em seguida, discute as métricas de avaliação dos dados gerados. Por fim, descreve o *dataset* de geração de ataques IoT que será utilizado como base para análise dos modelos generativos.

4.1. Modelos generativos de *datasets* sintéticos

Tabela 1. Descrição dos parâmetros

Parâmetro	Sigla	Função
noise_dimensions	nd	Tamanho da amostra aleatória enviada ao Gerador
generator_dimensions	gd	Tamanho da saída que será produzida por cada entrada de noise_dimension
critic_dimensions	cd	Tamanho da saída para cada uma das camada do discriminador
gen_attribute_num_units	ganu	Número de neurônios em cada camada do gerador de metadados
gen_attribute_num_layers	ganl	Número de camadas no gerador de metadados
disc_num_units	dnu	Número de unidades em cada camada do discriminador auxiliar
attr_disc_num_units	adnu	Número de unidades em cada camada do discriminador de metadados auxiliar
attr_disc_num_layers	adnl	Número de camadas no discriminador auxiliar
batch_size	bs	Tamanho dos lotes de treino

Para geração dos *datasets* sintéticos de ataque IoT treinamos os dois modelos considerando parâmetros que, de acordo a implementação de cada modelo, influenciam a geração de maneira equivalente. Nesse sentido, a Tabela 1 apresenta uma descrição

dos parâmetros e a Tabela 2 apresenta a lista de hiperparâmetros utilizadas que variamos para cada modelo. Em teoria, aumentar o valor do parâmetro significa aumentar sua capacidade de aprendizado e, consequentemente, da qualidade do *dataset* sintético gerado. Isso permite, por exemplo, que verifiquemos se variar tais parâmetros influencia de fato o resultado da geração e, consequentemente, a fidelidade e qualidade dos *datasets* gerados. Para facilitar a visualização, os nomes dos parâmetros foram abreviados.

Tabela 2. Hiperparametrização dos modelos NetShare e CTGAN

Experimento	**CTGAN**				**NetShare**						
	bs	gd	cd	nd	bs	ganu	ganl	dnu	dnl	adnu	adnl
h_1	50	256	256	264	50	512	5	512	5	512	5
h_2	100	256	256	264	100	512	5	512	5	512	5
h_3	200	256	256	264	200	512	5	512	5	512	5
h_4	50	192	192	198	50	384	4	384	4	384	4
h_5	100	192	192	198	100	384	4	384	4	384	4
h_6	200	192	192	198	200	384	4	384	4	384	4
h_7	50	128	128	132	50	256	3	256	3	256	3
h_8	100	128	128	132	100	256	3	256	3	256	3
h_9	200	128	128	132	200	256	3	256	3	256	3

Cada modelo foi treinado pelo mesmo número de épocas, em que monitoramos o tempo de execução e consumo de memória para o treinamento e geração dos novos dados. Nesse contexto, utilizamos 80% do *dataset* (14.4 Mb) para treino e o restante (5.4 Mb) para teste. Contudo, dado o número de parâmetros e número de treinos necessários para cada modelo, reduziu-se o tamanho do *dataset* de treino e teste. Dessa forma, para treinar os modelos GANs usamos 35% dos dados de treino e 35% dos dados de teste. Os dois modelos foram executados em uma máquina com GPU dedicada NVidia GeForce RTX 4090, processador Intel i9 @3,70Ghz x 20 núcleos, e 128 Gb de memória RAM.

4.2. Métricas

4.2.1. Fidelidade

A divergência da distribuição dos dados sintéticos em relação aos dados reais foram comparadas usando as métricas KL-d, JS-d e W_1, calculados de acordo com a Seção 2.2. Considerando que os dados de fluxo são tabulares, foram analisadas a distribuição de cada coluna do *dataset* estudado.

4.2.2. Utilidade média

Considerando a abordagem proposta no TSTR score, propomos uma nova métrica quantitativa para avaliar a qualidade de um *dataset* sintético. A medida se baseia na ideia de que a utilização de diferentes modelos de classificação/predição podem ser aplicados para obtenção do TSTR. Nesse sentido, estudos da literatura [Esteban et al. 2017, Marani and Nehdi 2022] demonstraram que, no geral, *datasets* sintéticos produzidos por um modelo eficiente irão obter os melhores scores no TSTR independente do modelo utilizado. Assim, a ideia central da métrica é utilizar diferentes modelos de classificação e calcular a média desses modelos em relação à média obtida pelo *dataset* real. Nesse caso,

pode-se utilizar diferentes métricas para avaliar o desempenho do modelo, como *accuracy*, *precision*, *recall* e *f1-score*.

Dessa forma, seja $c_1, \cdots, c_n$ os modelos de classificação utilizados para obter o TSTR *score*, e $s_1, \cdots, s_n$ os scores obtidos por cada modelo nos dados sintéticos, onde s_i pode ser *accuracy*, *precision*, *recall* ou *f1-score*. A utilidade será calculada por

$$\bar{u} = \frac{\frac{1}{n}\sum_{i=1}^{n} s_i}{u_r} \tag{3}$$

onde u_r é a média dos scores obtidos em cada modelo $c_1, \cdots, c_n$ quando aplicados no *dataset* real.

4.3. Datasets

Para a avaliação dos modelos generativos utilizamos o *dataset* IoT-23 [Sebastian Garcia 2020], que contém 20 capturas de *malware* e 3 capturas benignas executadas em aparelhos IoT. Foi publicado em janeiro de 2020, com capturas que variam de 2018 até 2019. Para fins desse trabalho, serão utilizados apenas as capturas do tipo *malware*. Essas capturas foram obtidas de aparelhos IoT infectados com uma amostra de um *malware* específico executado em um Raspberry Pi.

Tabela 3. Proporção de fluxos úteis dos *datasets* utilizados.

	Malware 1-1	Malware 7-1	Malware 9-1	Malware 33-1	Malware 36-1	Malware 39-1	Malware 60-1
Registros	1008748	11454714	6378293	54454591	13645098	73568981	3581028
Após limpeza	212448	35172	16612	17343	9671	6881	624
%	21	0.30	0.26	0.03	0.07	0.01	0.02

O *dataset* possui as seguintes colunas de nosso interesse: *Transport Stream* (ts), IP de origem (SrcIp), porta de origem (SrcPort), IP de destino (DstIp), porta de destino (DstPort), protocolo (proto), duração do fluxo (*duration*), número de bytes (bytes) e pacotes (pkts). Os rótulos possíveis para cada fluxo são *Attack*, *Benign*, C&C, DDoS, FileDownload, HeartBeat, Mirai, Okiru, PartOfAHorizontalPortScan e Torii. Para facilitar a classificação e análise dos dados, definimos dois rótulos possíveis, o mesmo *Benign*, e, *Non-Benign* (não benigno), mesclando todas as outras *labels*.

Antes de ser utilizado, uma etapa de pré-processamento removeu registros os quais uma das colunas de interesse possuia valores nulos. As reduções para cada captura utilizada são apresentadas na Tabela 3. Nesse sentido, ressalta-se que a redução teve impactos mínimos na representatividade dos dados, tendo em vista que o dataset já era originalmente desbalanceado (com mais classes de ataque). Contudo, devido à pequena quantidade de dados úteis disponíveis por captura, optou-se por unir os dados de todas as capturas, resultando em um *dataset* final com 298751 registros ($\approx$ 18Mb).

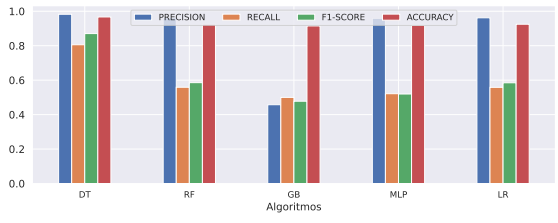
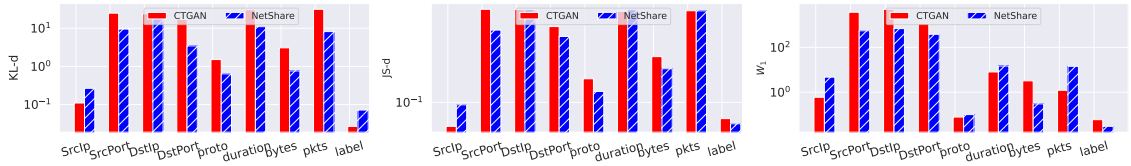


Figura 1. Classificação de ataques em IoT baseado nos dados reais após filtros.

A Figura 1 apresenta a *Accuracy*, *Precision*, *Recall* e *F1-Score* para a classificação de tráfego benigno e maligno utilizando o *dataset* após pré-processamento. Foram comparados os resultados dos algoritmos de classificação DT, RF, GB, MLP e LR, mencionados anteriormente. Observa-se que, com exceção do GB, os demais algoritmos obtiveram *Accuracy* e *Precision* similares.

5. Resultados

Para cada GAN, foi gerado um *dataset* sintético para cada um dos h hiperparâmetros, totalizando nove *datasets* sintéticos para a CTGAN e nove para o Netshare. A Figura 2 apresenta as médias (considerando os 9 modelos treinados) das métricas de fidelidade para cada coluna do dado sintético. Relembramos que, de acordo com essas métricas, quanto mais próximas de 0, mais similar ao *dataset* real será o *dataset* sintético. Além disso, para facilitar a visualização, o eixo Y está em escala logarítmica. Nota-se, assim, que em média, o NetShare aparenta gerar *datasets* mais similares. Exceto nas colunas SrcIp e label, para o KL-d (Figura 2(a)), SrcIP e duration, para o JS-d (Figura 2(b)) e, para o W_1 (Figura 2(c)), as colunas SrcIp, proto, duration e pkts.

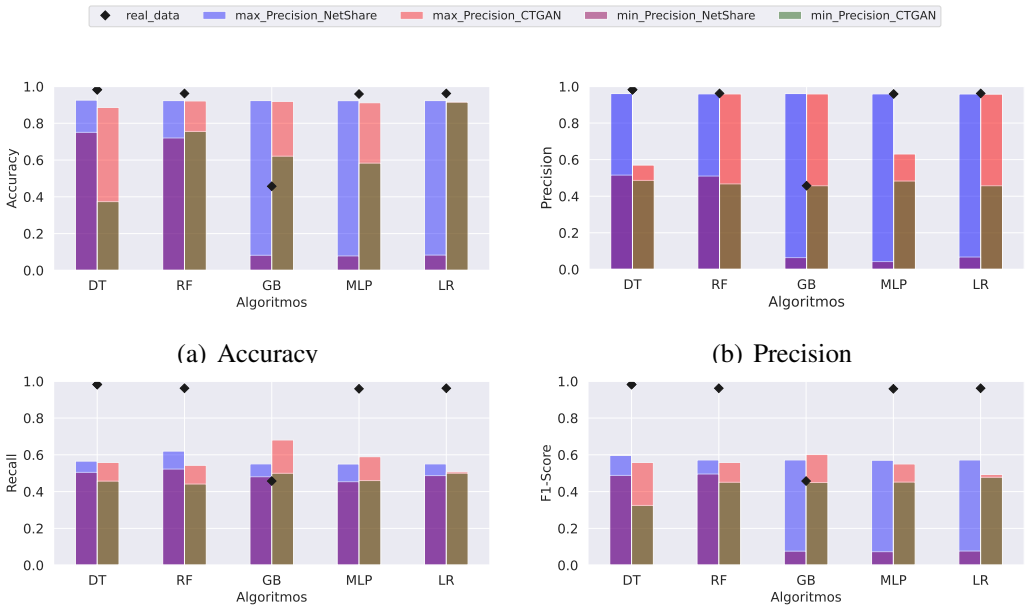


(a) KL-Divergence (b) JS Distance (c) W_1

Figura 2. Fidelidade dos dados sintéticos para cada modelo generativo.

Para verificar se é possível utilizar os *datasets* sintéticos em aplicações práticas de classificação de forma satisfatória, apresentamos na Figura 3 a comparação entre o NetShare e a CTGAN utilizando o TSTR *score* baseado na *Accuracy*, *Precision*, *Recall* e *F1-score*. Percebe-se que para as métricas *Accuracy* e *Precision*, ao menos uma das 9 configurações de cada modelo conseguiu obter um *score* próximo ao valor do *score* real. Por exemplo, os modelos do NetShare h_5 e h_9 obtiveram maior *precision* para o GB/DT ($h5$) e RF/MLP (h_9) e os modelos h_5 e h_6 apresentaram maior valor na *Accuracy* para os algoritmos GB/MLP e RF/LR, respectivamente. Para os modelos da CTGAN, h_2 obteve melhor *Precision* nos algoritmos DT, RF e MLP e o modelo h_2 apresentou melhor *Accuracy* nos algoritmos GB e LR. Nesse sentido, é interessante notar que no algoritmo GB, alguns modelos tanto da CTGAN (h_2/h_7) quanto do NetShare (h_5) geraram *datasets* sintéticos cujos *scores* foram superiores ao do *dataset* real. Além disso, percebe-se também que, no geral, considerando o TSTR, os modelos da CTGAN são mais consistentes. Por exemplo, os valores mínimos dos modelos da CTGAN são maiores que os valores mínimos do NetShare, exceto na *Accuracy* (DT) e nas métricas *Precision*, *Recall* e *F1-score* (algoritmos DT e RF).

A Tabela 4 apresenta a utilidade média de cada configuração dos modelos CTGAN e NetShare. Os valores em negrito destacam os modelos com melhor utilidade média. Nesse sentido, nota-se que não é possível identificar uma relação clara entre o tamanho/capacidade do modelo treinado com seu desempenho, considerando a utilidade média. Como mostrado



(a) Accuracy (b) Precision

(c) Recall (d) F1-score

Figura 3. Utilidade dos dados sintéticos para o TSTR.

na Figura 3 anterior, alguns modelos irão apresentar melhor desempenho em uma métrica e desempenho inferior em outra. Entretanto, percebe-se que os modelos h_1 e h_2 do NetShare aparentam ter um desempenho bem inferior, com utilidades médias de 0.46 e 0.57, respectivamente, quando comparados aos modelos com configurações equivalentes da CTGAN. Além disso, exceto pelo modelo h_7, métrica *Precision*, os modelos do CTGAN têm utilidade média igual ou superior a 0.67.

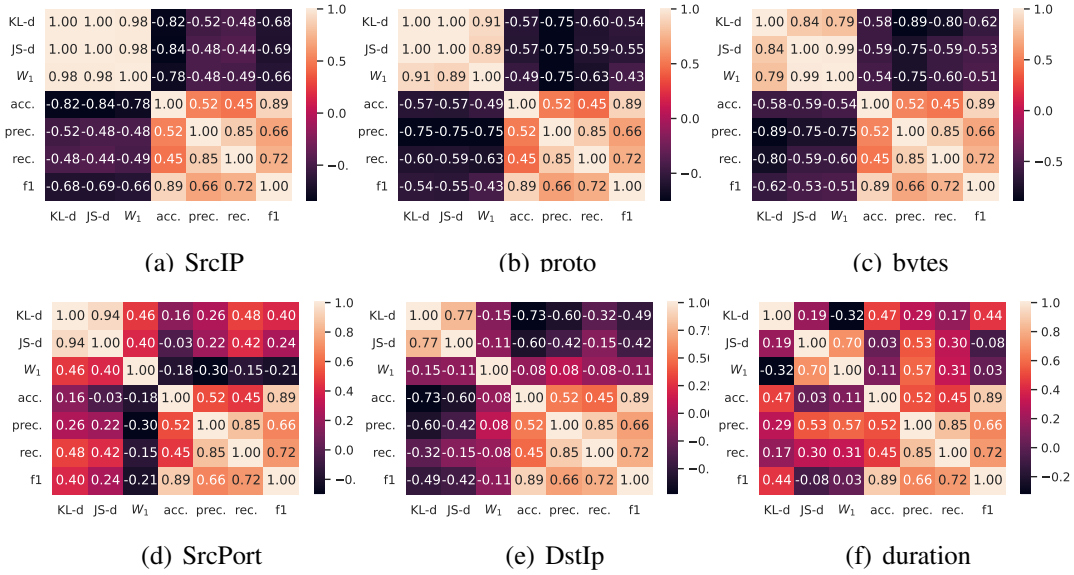
Tabela 4. Utilidade media dos modelos

Métricas	**CTGAN**									**NetShare**								
	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9
Accuracy	0.88	0.86	0.95	0.79	0.94	0.79	0.82	0.94	**0.98**	0.56	0.63	0.73	0.79	**0.99**	0.81	**0.99**	**0.99**	**0.99**
Precision	0.84	0.67	0.79	0.65	**0.91**	0.8	0.56	0.68	0.83	0.46	0.57	0.71	0.79	**1.11**	0.7	1.08	1.06	0.86
Recall	**0.93**	0.84	0.9	0.85	0.89	0.88	0.83	0.86	0.89	0.9	0.86	0.88	0.89	0.91	0.89	0.9	**0.92**	0.89
F1-score	0.83	0.77	**0.86**	0.76	0.83	0.76	0.77	0.8	**0.86**	0.56	0.58	0.69	0.73	0.9	0.72	0.89	**0.92**	0.86

Ressalta-se, neste caso, que com a métrica proposta é possível ter-se um panorama mensurável do potencial de aplicação prática dos datasets sintéticos, nos diferentes modelos de classificação. Unificando os TSTR *scores* de diferentes modelos em um único resultado, é possível identificar mais facilmente qual modelo está produzindo datasets sintéticos segundo os objetivos esperados. Por exemplo, na literatura o TSTR *score* tende a ser calculado por meio da acurácia. A utilidade média consegue indicar, nesse caso, que o modelo h_9, para o CTGAN, e h_5, h_7, h_8 e h_9, para o NetShare, apresentam o melhores resultados para a acurácia.

A fim de avaliar possíveis correlações entre as métricas de fidelidade e a utilidade média, apresentamos na Figura 4 as correlações das métricas de fidelidade com a utilidade média, para 2 colunas categóricas e uma numérica da CTGAN (linha superior) e NetShare (linha inferior). Em suma, espera-se que um *dataset* com alta fidelidade possua também valores mais altos para as métricas de utilidade média. Ou seja, espera-se encon-

trar correlações positivas entre a fidelidade e utilidade. Por questão de espaço no texto, mostramos apenas as correlações mais relevantes de cada modelo e abreviamos os nomes das métricas. No geral, para a CTGAN, nota-se que as métricas de fidelidade, especificamente a W_1, possuem maior correlação com a *Accuracy*: 0.62 na coluna SrcIP (Figura 4(a)), 0.66 na coluna proto (Figura 4(b)) e 0.53 na coluna bytes (Figura 4(c)). É interessante notar, nesse caso, que há uma correlação positiva moderada entre W_1 e *F1-score* de 0.53, na Figura 4(b). Além disso, é evidente que as métricas de fidelidade produzem resultados similares, com a menor correlação, na Figura 4(c), sendo uma correlação positiva forte.



(a) SrcIP (b) proto (c) bytes

(d) SrcPort (e) DstIp (f) duration

Figura 4. Comparação entre as métricas de distribuição e a utilidade media

Para o NetShare, notam-se correlações positivas moderadas entre W_1 e *Accuracy* nas colunas SrcPort (0.51) e DstIp (0.57), respectivamente, nas Figuras 4(d) e 4(e). Nesse sentido, especificamente na coluna DstIp, nota-se também correlações positivas moderadas entre JS-d e *Accuracy* (0.64), *Precision*(0.55), e *F1-score* (0.62). Relações positivas fracas também podem ser vistas entre W_1 e JS-d (0.40) e W_1 e KL-d(0.46). Ressalta-se, por fim, que é possível identificar também algumas correlações negativas entre as métricas de fidelidade e utilidade média, embora elas não sejam significativas.

Tabela 5. Tempos de execução dos modelos CTGAN e NetShare.

	CTGAN		**NetShare**	
Experimento	Treinamento	Geração	Treinamento	Geração
h_1	03:12:52	00:00:39	4:56:56	00:36:57
h_2	01:36:59	00:00:20	2:24:37	00:47:05
h_3	00:48:21	00:00:11	1:14:12	00:50:53
h_4	3:09:45	0:00:39	4:41:48	00:46:36
h_5	1:35:19	0:00:20	2:21:31	00:50:34
h_6	0:47:50	0:00:11	1:11:56	00:50:42
h_7	3:13:23	0:00:39	4:32:58	00:45:37
h_8	1:35:39	0:00:20	2:18:59	00:49:57
h_9	00:47:33	0:00:11	1:09:05	00:51:09

Para avaliar o desempenho dos modelos em relação ao treinamento de geração dos *datasets* sintéticos, mensuramos o tempo gasto por cada configuração dos modelos para treinar e gerar os dados sintéticos, apresentados na Tabela 5. É evidente que o NetShare possui um treinamento mais lento que a CTGAN (entre 20 minutos e 1 hora à mais), entretanto, a maior diferença entre eles está na geração. O modelo do NetShare com geração mais rápida (h_1), demora cerca de 36 minutos para gerar um único *dataset* sintético. Por outro lado, na CTGAN o modelo mais lento na geração lava cerca de 39 segundos para gerar um *dataset* sintético. Nesse caso, é evidente que a CTGAN é mais indicada quando se necessita de uma geração de um grande volume de *datasets* sintéticos.

Coletamos o consumo de CPU ($\%$) e Memória (Gb) utilizadas durante o treino e geração a cada 5 segundos, para os dois modelos. No geral, o consumo de CPU e Memória dos *datasets* foi similar. O NetShare usa pouca CPU durante o treino, uma média de 34% com um desvio padrão de $\approx 47\%$. Na geração, contudo, o consumo de CPU aumenta consideravelmente, com uma média entre 94% e 100% e um desvio padrão entre 7 e 9. De maneira similar, há pouca memória alocada para o treinamento (uma média de 4.6GB com um desvio padrão de 0.5Gb), mas um alto consumo durante a geração: em alguns casos a média de uso passa dos 19Gb, podendo chegar a um pouco mais). A CTGAN, por outro lado, manteve-se consistente durante o treino e geração. Apesar do consumo mais alto da CPU (uma média entre 53% e 75%, e desvios podendo chegar a $\approx 110\%$), pouca memória foi utilizada, com uma média de 1.7 Gb e desvio de ≈ 0.9 Gb nos piores casos. É importante ressaltar que o consumo de CPU acima de 100% significa que mais de um núcleo disponível na máquina foi utilizado.

Por fim, destacamos a influência dos parâmetros nos resultados obtidos. Observou-se que, no geral, modelos com maiores capacidades de aprendizado (por exemplo, mais camadas, mais unidades por camadas e alimentados por entradas aleatórias maiores) apresentaram melhores resultados quando se considera a fidelidade, tanto no CTGAN quanto no NetShare. Nesse sentido, não é possível identificar com clareza quais dos dois modelos é melhor para geração dos datasets de ataque. Contudo, nota-se que o NetShare é o modelo mais lento tanto no treinamento quanto na geração, além de consumir mais memória na geração (provavelmente pelo processo de decodificação dos dados sintéticos). Assim, o custo de tempo e memória despendido pelo NetShare não justifica os poucos ganhos em fidelidade e qualidade dos dados sintéticos gerados por ele.

6. Conclusão

Neste trabalho investigamos a fidelidade e utilidade de *datatasets* sintéticos de ataque em dispositivos IoT gerados por GANs. Para isso, utilizamos um *dataset* contendo fluxos normais e de ataque em dispositivos IoT e comparamos o desempenho de modelos generativos considerando tempo de treinamento e geração, memória utilizada e comparamos a influência de hiperparâmetros nos resultados dos modelos.

Os experimentos mostram que, no geral, a variação dos parâmetros pouca influenciou nos resultados quando se considera a utilidade dos datasets sintéticos. Contudo, é evidente que há uma influência maior para os resultados de fidelidade. Nesse contexto, embora os resultados tenham sido satisfatórios para os dois modelos, não está evidente qual deles pode ser considerado melhor, considerando a fidelidade e utilidade. Além disso, a métrica de utilidade média proposta, provê uma visão geral da utilidade dos modelos generativos,

podendo facilitar a escolha de modelos generativos para aplicação em cenários práticos.

Como trabalhos futuros, pretende-se avaliar, com maior profundidade, a influência de outros hiperparâmetros nos resultados dos modelos. Tanto em relação à fidelidade e qualidade, quanto no consumo de memória e tempos de execução. Além disso, pretende-se desenvolver um gerador de fluxo de dados que possa ser integrado aos simuladores de rede, como NS-3, por exemplo.

Agradecimentos

Este trabalho possui financiamento de: CNPq, CAPES (Código de Financiamento 001), FAPES (#2023/RWXSZ; #2022/ZQX6; #2022/NGKM5; #2021/GL60J) e Fapesp/MCTI/CGI.br (#2020/05182-3 e #2023/00148-0).

Referências

Aleroud, A., Yang, F., Pallaprolu, S. C., Chen, Z., and Karabatis, G. (2021). Anonymization of network traces data through condensation-based differential privacy. *Digital Threats: Research and Practice (DTRAP)*, 2(4):1–23.

Alex, C., Creado, G., Almobaideen, W., Alghanam, O. A., and Saadeh, M. (2023). A comprehensive survey for iot security datasets taxonomy, classification and machine learning mechanisms. *Computers & Security*, page 103283.

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR.

Borji, A. (2022). Pros and cons of gan evaluation measures: New developments. *Computer Vision and Image Understanding*, 215:103329.

Brock, A., Donahue, J., and Simonyan, K. (2018). Large scale gan training for high fidelity natural image synthesis.

Brophy, E., Wang, Z., She, Q., and Ward, T. (2023). Generative adversarial networks in time series: A systematic literature review. *ACM Computing Surveys*, 55(10):1–31.

Cunha, V. C., Zavala, A. Z., Magoni, D., Inácio, P. R. M., and Freire, M. M. (2022). A complete review on the application of statistical methods for evaluating internet traffic usage. *IEEE Access*, 10:128433–128455.

Dash, A., Ye, J., and Wang, G. (2023). A review of generative adversarial networks (gans) and its applications in a wide variety of disciplines: From medical to remote sensing. *IEEE Access*.

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

Gheisari, M., Alzubi, J., Zhang, X., Kose, U., and Saucedo, J. A. M. (2020). A new algorithm for optimization of quality of service in peer to peer wireless mesh networks. *Wireless Networks*, 26:4965–4973.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Hossain, M. D., Ochiai, H., Doudou, F., and Kadobayashi, Y. (2020). Ssh and ftp brute-force attacks detection in computer networks: Lstm and machine learning approaches. In *2020 5th international conference on computer and communication systems (ICCCS)*, pages 491–497. IEEE.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Kumar, V. and Sinha, D. (2023). Synthetic attack data generation model applying generative adversarial network for intrusion detection. *Computers & Security*, 125:103054.

Marani, A. and Nehdi, M. L. (2022). Predicting shear strength of frp-reinforced concrete beams using novel synthetic data driven deep learning. *Engineering Structures*, 257:114083.

Nekvi, R. I., Saha, S., Al Mtawa, Y., and Haque, A. (2023). Examining generative adversarial network for smart home ddos traffic generation. In *2023 International Symposium on Networks, Computers and Communications (ISNCC)*, pages 1–6. IEEE.

Pundir, M., Sandhu, J. K., and Kumar, A. (2021). Quality-of-service prediction techniques for wireless sensor networks. In *Journal of Physics: Conference Series*, volume 1950, page 012082. IOP Publishing.

Qian, C., Yu, W., Lu, C., Griffith, D., and Golmie, N. (2022). Toward generative adversarial networks for the industrial internet of things. *IEEE Internet of Things Journal*, 9(19):19147–19159.

Sebastian Garcia, Agustin Parmisano, . M. J. E. (2020). Iot-23: A labeled dataset with malicious and benign iot network traffic (version 1.0.0) [data set].

Shahid, M. R., Blanc, G., Jmila, H., Zhang, Z., and Debar, H. (2020). Generative deep learning for internet of things network traffic generation. In *Pacific Rim International Symposium on Dependable Computing*, pages 70–79. IEEE.

Sharafaldin, I., Gharib, A., Lashkari, A. H., and Ghorbani, A. A. (2018). Towards a reliable intrusion detection benchmark dataset. *Software Networking*, 2018(1):177–200.

Wang, Z., She, Q., and Ward, T. E. (2021). Generative adversarial networks in computer vision: A survey and taxonomy. *ACM Computing Surveys (CSUR)*, 54(2):1–38.

Xu, L., Skoularidou, M., Cuesta-Infante, A., and Veeramachaneni, K. (2019). Modeling tabular data using conditional gan. *Advances in neural information processing systems*, 32.

Yin, Y., Lin, Z., Jin, M., Fanti, G., and Sekar, V. (2022). Practical gan-based synthetic ip header trace generation using netshare. In *Proceedings of the ACM SIGCOMM 2022 Conference*, pages 458–472.