Detecção de Ataques de Botnets em IoT via Variational Autoencoder

Andressa A. Cunha[1], **João B. Borges**[2], **Antonio A. F. Loureiro**[1]

[1]Departamento de Ciência da Computação
Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte, MG

[2]Departamento de Computação e Tecnologia
Universidade Federal do Rio Grande do Norte (UFRN)
Caicó, RN

andressa.amaral@dcc.ufmg.br, joao.borges@ufrn.br, loureiro@dcc.ufmg.br

Abstract. *Internet of Things (IoT) is a current trend worldwide, and its importance is hugely increasing in society. With its popularity, it also became important to fully understand the functionality of IoT networks, taking into account relevant requirements such as security. This work provides a deep learning model based on the Variational Autoencoder (VAE), capable of detecting network traffic anomalies on IoT devices caused by botnet attacks. The experiments are conducted over a known botnet detection dataset, the N-BaIoT, and achieved results are compared with baselines from recent studies. The results show that the proposed strategy is suitable for attack detection tasks and could improve security in IoT environments with low latency responses.*

Resumo. *A Internet das Coisas (IoT) é uma tendência mundial, e sua importância na sociedade cresce com o passar dos anos. Com sua popularidade, também tornou-se importante entender a funcionalidade das redes IoT, levando em consideração requisitos relevantes como a segurança. Esse trabalho fornece um modelo de Aprendizado Profundo (DL) baseado no Variational Autoencoder (VAE), capaz de detectar anomalias no tráfego de dispositivos IoT causadas por ataques de botnets. Os experimentos conduzidos sobre um conjunto de dados conhecido, o N-BaIoT, e os resultados alcançados são comparados aos baselines de estudos recentes. Os resultados demonstram que a estratégia proposta é adequada para a tarefa de detecção de ataques e pode melhorar a segurança em ambientes IoT com respostas em baixa latência.*

1. Introdução

Com o recente aumento da popularidade da Internet das Coisas (do inglês, *Internet of Things* - IoT) ao redor do mundo, torna-se necessário entender melhor os requisitos e desafios associados a essa tecnologia. A segurança dos dados em dispositivos IoT é um dos requisitos mais pesquisados atualmente, principalmente, quanto à sua integridade e confidencialidade. Tal interesse se justifica por conta do grande número de aplicações implantadas nesse meio, do alto tráfego de rede, da necessidade de proteger dados sensíveis (como no caso de casas e sistemas de saúde inteligentes) e a falta de padronização das aplicações existentes (Alqahtani et al., 2020; Laguduva et al., 2019).

Dispositivos IoT podem ser um alvo fácil para ataques vindos da rede, apesar de muitos esforços empregados na área para criar medidas de proteção adequadas (Meidan et al., 2018; Mirsky et al., 2018). Consequentemente, como uma rede IoT pode escalar rapidamente ao serem adicionados diferentes dispositivos no ambiente, criar um modelo de segurança diferente para cada dispositivo isolado torna-se impraticável. Desta forma, a metodologia empregada atualmente para construir um ambiente IoT seguro é a de desenvolver rotinas de detecção de ataques para garantir a segurança dos dados de rede em toda a infraestrutura (Alqahtani et al., 2020).

Alguns dos ataques de rede mais comuns e prejudiciais no meio IoT, que são tratados neste trabalho, são causados pelas *botnets* Mirai e Bashlite. Em geral, uma *botnet* é composta por um grande número de dispositivos controlados remotamente, os quais começam a infectar outros equipamentos, criando uma rede de dispositivos afetados. Uma *botnet* é criada quando um atacante, o *botmaster*, envia comandos remotamente para os dispositivos infectados causando um ataque coordenado que sobrecarrega e afeta a disponibilidade de um dado alvo. Um famoso exemplo é o DDoS (*Distributed Denial of Service*). Esse tipo de ataque pode causar danos em larga escala, portanto é um problema severo a ser combatido, especialmente em ambientes IoT.

Devido à grande quantidade de dados que ataques de *botnets* podem gerar, estudos recentes mostram que sistemas inteligentes são adequados para as tarefas de detecção e prevenção de ameaças (Aversano et al., 2021; Hussain et al., 2020). Assim, uma solução razoável é aplicar técnicas de Aprendizado Profundo (do inglês, *Deep Learning* - DL) nas redes IoT. DL é um subcampo do Aprendizado de Máquina (do inglês, *Machine Learning* - ML) que se utiliza de muitas camadas de redes neurais para adquirir conhecimento e entender os padrões da informação analisada. Utilizar técnicas de aprendizado profundo para resolver a tarefa proposta é uma solução interessante, uma vez que modelos DL podem alcançar alta acurácia e processar grandes quantidades de dados de maneira relativamente rápida. Em particular, este trabalho propõe o uso de um modelo de *Variational Autoencoder* (VAE) para detectar se os dados do tráfego analisado são benignos ou não.

As contribuições propostas por este trabalho são:

- Propor uma estratégia de aprendizado profundo semi-supervisionada capaz de detectar ataques com antecedência em dispositivos IoT e aplicá-la em uma base de dados conhecida (N-BaIoT, fornecida por Meidan et al. (2018));
- Comparar os resultados obtidos neste projeto com outros trabalhos, com o intuito de avaliar o desempenho geral do modelo proposto;
- Avaliar o tempo gasto na etapa de teste, fornecendo uma maneira de alcançar bom desempenho do algoritmo em termos de tempo com latência reduzida;

A base de dados usada neste projeto, coletada e disponibilizada por Meidan et al. (2018), compreende informações de tráfego adquiridas a partir de nove dispositivos IoT diferentes. Nela, os autores obtiveram dados benignos e malignos do tráfego de rede dos dispositivos, monitorando-os quando o uso do tráfego era regular e quando atacados pelas *botnets* Mirai e Bashlite.

2. Trabalhos Relacionados

Nesta seção, são apresentados trabalhos relacionados que tratam da detecção de ataques de *botnets* em dispositivos IoT, especialmente os que se utilizam de estratégias de apren-

dizado de máquina para lidar com a tarefa. Tais estudos representam um incentivo para o desenvolvimento do presente trabalho.

Hussain et al. (2020) sumarizam em sua pesquisa diversos trabalhos relacionados à segurança em IoT, enfatizando o uso de modelos de aprendizado de máquina e aprendizado profundo nas abordagens pesquisadas. Redes IoT são normalmente compostas por muitos dispositivos heterogêneos conectados entre si que transferem uma grande quantidade de dados simultaneamente e requerem comunicação ultra-confiável e de baixa latência. Os autores sugerem que utilizar modelos de ML e DL são técnicas promissoras para enfrentar os desafios recentes da segurança IoT ao garantir sistemas robustos, inteligentes e confiáveis. Uma discussão semelhante é proposta em Aversano et al. (2021), mas seu foco reside no uso de estratégias de aprendizado profundo para detecção de ataques. Modelos de DL, em geral, são mais adequados para executar em contextos complexos, com uma quantidade massiva de dados, o que dá suporte à escolha de um modelo de aprendizado profundo na detecção de anomalias na rede.

Para identificar ataques de *botnets* e prevenir que os ambientes IoT sejam infectados, Meidan et al. (2018) criaram uma base de dados completa contendo o comportamento do tráfego de dispositivos distintos em uma rede IoT, adquiridos em um dado intervalo de tempo. Esses dados foram coletados considerando dispositivos que estavam tanto funcionando apropriadamente (tráfego benigno) quanto sendo atacados pelas *botnets* Mirai e Bashlite (tráfego maligno). Eles propuseram o uso de uma rede de Autoencoders profundos para detectar previamente a existência de ataques malignos no ambiente.

O trabalho de Meidan et al. (2018) encorajou diversos outros pesquisadores a empregar esforços na tarefa de detecção de *botnets*, especialmente através de técnicas de aprendizado de máquina. Por exemplo, Borges et al. (2022) usaram uma Isolation Forest para identificar possíveis anomalias nesse conjunto de dados ao avaliar a dinâmica das séries temporais de tráfego dos dispositivos avaliados. Alqahtani et al. (2020) desenvolveram um classificador XGBoost otimizado combinado a um algoritmo genético para detectar ataques em sistemas IoT usando a base de dados N-BaIoT. Além disso, adicionaram um passo extra para seleção de características com um algoritmo baseado na pontuação Fisher para escolher as colunas que forneçam mais informações, limitando o tamanho da base de dados. Nomm and Bahsi (2018) aplicaram uma solução não-supervisionada (baseada em *Support Vector Machines*, SVMs) para detectar tráfego anômalo com um espaço de características reduzido e gerando um único modelo a ser aplicado nos diferentes dispositivos. Com essa abordagem, os autores buscam otimizar o uso de recursos computacionais e reduzir a complexidade dos modelos.

A arquitetura proposta por Mirsky et al. (2018), chamada Kitsune, é um modelo de detecção online composto por um conjunto de Autoencoders. Os autores argumentam que o uso de *Artificial Neural Networks* (ANNs) nesse tipo de tarefa é interessante, pois é possível aprender conceitos não-lineares dos dados de entrada com uma excelente performance. Eles também propuseram um passo de extração de características para adquirir as informações essenciais fornecidas por um conjunto de câmeras de segurança. A metodologia proposta neste trabalho para coletar os dados dos dispositivos é a mesma usada por Meidan et al. (2018) para montar a base N-BaIoT. O trabalho de Parra et al. (2020) também empregou o N-BaIoT para treinar e testar uma estratégia de aprendizado profundo para detecção de ataques de *phishing* e de *botnets*. Nesse caso, os autores criaram

um modelo de DL para detectar e mitigar ataques usando uma *Distributed Convolutional Neural Network* (DCNN) e um modelo *Long-Short Term Memory* (LSTM). O primeiro foi avaliado com um dataset de *phishing* desenvolvido pelos autores, e o segundo foi aplicado aos dados do N-BaIoT.

A escolha do *Variational Autoencoder* para a tarefa de detecção de ataques em redes IoT é encorajado pelo amplo uso desse modelo na literatura. Lopez-Martin et al. (2017) apresentaram um dos primeiros trabalhos a aplicar um VAE no problema de detecção de ataques usando o conjunto de dados NSL-KDD, obtendo bons resultados se comparado a outros métodos. Yang et al. (2019) conseguem melhorar a eficácia na classificação de intrusos de rede com um *Conditional* VAE, aplicando-o tanto no conjunto NSL-KDD quanto no UNSW-NB15. No trabalho de Kim et al. (2020), um *Recurrent Variational Autoencoder* é usado para classificar o conjunto de dados CTU-13 e o modelo é avaliado com um método de detecção em tempo real. Song et al. (2021) avalia o uso de Autoencoders para diferentes conjuntos de dados, incluindo o N-BaIoT, com distintos hiper-parâmetros. Além disso, Alsoufi et al. (2021) fizeram uma revisão completa do uso de técnicas de DL para detecção de anomalias, considerando os distintos *datasets* públicos que estão disponíveis e as estratégias aplicadas. No geral, as estratégias mais utilizadas para essa tarefa são as CNNs, LSTMs e os Autoencoders, e ainda existe uma lacuna a ser explorada no uso de *Variational Autoencoders* especificamente para a base N-BaIoT.

3. Fundamentos

3.1. Ataques de *Botnets* em Dispositivos IoT

A Internet das Coisas é um conceito que define o uso de instrumentos físicos (atuadores, sensores) que são conectados à Internet e entre si, criando uma rede de dispositivos que podem adquirir e processar informações do ambiente. Nos últimos anos, a segurança dos dados em dispositivos IoT, principalmente quanto à sua confidencialidade, integridade e privacidade, tem sido uma área de estudo muito pesquisada, uma vez que tais aplicações estão cada vez mais se tornando parte da sociedade conectada (Aversano et al., 2021; Hussain et al., 2020). Desta forma, tornou-se essencial garantir a segurança dos dados sensíveis dos usuários e garantir o correto funcionamento dos equipamentos.

Um dos motivos que dificulta a criação de uma solução única capaz de garantir a segurança destes dispositivos é a falta de padronização dos protocolos e arquiteturas na IoT (Alqahtani et al., 2020). Desta forma, uma prática que tem alcançado bons resultados nos ambientes IoT, consiste na utilização de modelos de ML aplicados para a detecção instantânea dos ataques vindos da rede (Meidan et al., 2018; Aversano et al., 2021; Hussain et al., 2020; Borges et al., 2022). Uma vez que os modelos podem ser executados diretamente no nível de rede, a arquitetura e os protocolos de cada dispositivo não representam mais um problema de padronização, tornando possível gerar alertas e promover a desconexão de dispositivos afetados por algum *malware*.

Meidan et al. (2018) apontam que dispositivos IoT são especialmente suscetíveis a ataques DDoS, os quais inundam o ambiente conectado e fazem os dispositivos ficarem indisponíveis. Kolias et al. (2017) apresentam dois exemplos de *botnets* que executam ataques DDoS em equipamentos IoT: Mirai e Bashlite (ou Gafgyt). O primeiro é capaz de deduzir as credenciais de acesso do usuário por força bruta e espalhar-se através da rede;

o segundo usa *scanners* para detectar dispositivos vulneráveis que possam ser infectados e, posteriormente, usados para infectar outros.

3.2. Variational Autoencoders (VAEs)

O *Variational Autoencoder* (VAE) é um modelo probabilístico generativo de aprendizado profundo introduzido por Kingma and Welling (2014) que desconstrói e reconstrói uma dada entrada para aprender as características existentes em seus dados. Ao passo que modelos discriminativos tentam predizer uma saída correta ao observar a dinâmica dos dados de entrada, modelos generativos tentam simular exatamente como os dados são construídos no mundo real. Através da literatura de aprendizado profundo, modelos generativos (e.g., VAEs, *Generative Adversarial Networks*) têm sido amplamente usados para detectar anomalias ou ruídos nos dados ao identificar as relações causais dos modelos e gerar novas bases que imitam as informações do mundo real (Kingma & Welling, 2019; An & Cho, 2015).

O VAE, assim como um *Autoencoder* (AE) regular, é formado por dois componentes: o codificador e o decodificador. O primeiro componente aprende uma função para representar os dados de entrada no espaço latente. Contudo, diferentemente de um AE tradicional, o VAE aprende como gerar dois vetores, que representam a média (μ) e a variância (σ) de uma distribuição de probabilidade $q(z|x)$ para a saída z dada uma entrada x. O espaço latente é uma representação em baixa dimensionalidade dos dados que compartilham informações semelhantes no espaço. A Equação 1 mostra a Gaussiana multivariada dos vetores latentes com uma dimensão m, a qual gera os z pontos para cada entrada x. O termo diag(σ^2) representa a matriz de co-variâncias.

$$q(z|x) = N_m(\mu, \text{diag}(\sigma^2)), \tag{1}$$

O decodificador é, então, responsável por reproduzir a entrada a partir da rede de z pontos usando a função aprendida $p(x|z)$. Quando o decodificador realiza esta ação, ele assinala um erro de reconstrução para a saída que indica se os dados reconstruídos são exatos ou não. Altos erros de reconstrução sugerem que a saída não é bem correlacionada com a entrada original e, por conta deste comportamento, o modelo proposto é eficiente para identificar anomalias na informação.

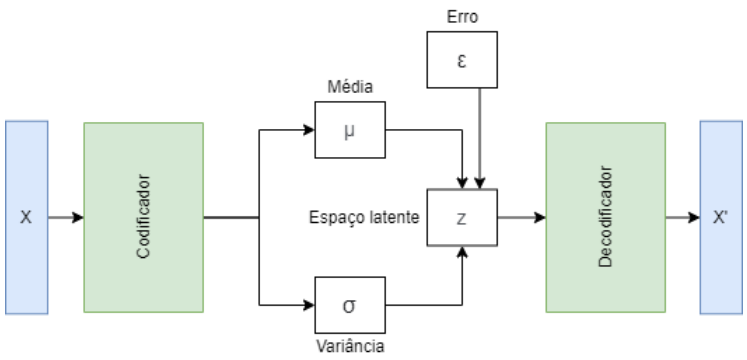

Figura 1. Diagrama que demonstra a arquitetura básica de um Variational Autoencoder (VAE).

Parâmetros de ambas as redes, codificador e decodificador, são treinados com o intuito de minimizar a função de perda composta por dois termos: o erro de reconstrução, já mencionado, e a divergência Kullback-Leiber (KL), que funciona como um fator de regularização. A Figura 1 apresenta a arquitetura básica de um modelo VAE para melhor entendimento de como tais unidades funcionam, baseada em An and Cho (2015) e Kingma and Welling (2014).

4. Modelo de Detecção de Ataques de *Botnets*

4.1. Dados de Entrada

Como discutido nas Seções 2 e 3.1, detectar possíveis anomalias em uma rede de tráfego de informações é bastante significativo para construir ambientes IoT seguros. A principal proposta deste trabalho é aplicar um modelo VAE, cuja explicação teórica está explicitada na Seção 3.2, em uma base de dados conhecida, a fim de avaliar os prós e contras desse tipo de estratégia na identificação de ataques de *botnets*, além de comparar os resultados com o estado da arte.

A base de dados empregada, N-BaIoT, disponibilizada por Meidan et al. (2018), é composta por informações adquiridas de nove dispositivos IoT diferentes. Para construí-la, os dados de tráfego de cada dispositivo foram propriamente coletados enquanto executando seu comportamento normal e enquanto sendo atacados por alguma *botnet*, Mirai ou Bashlite. Os autores do N-BaIoT seguiram a metodologia proposta por Mirsky et al. (2018), aplicando os passos descritos nos dispositivos disponíveis em seu laboratório. A metodologia proposta é composta pelos seguintes passos:

- Coletar os pacotes de tráfego gerado pelo dispositivo observado e criar um arquivo binário bruto a partir dele. O processo de obtenção é conduzido em cinco intervalos de tempo diferentes: 100 ms, 500 ms, 1 s, 1.5 s, 10 s e 1 min.
- Determinar informações obtidas a partir do binário bruto, como tamanho do pacote, tempo de chegada e endereços de rede, entre outros.
- Extrair do arquivo binário 115 características estatísticas (média, variância, etc.) relacionadas às informações do tráfego, que são sumarizadas para cada um dos cinco intervalos de tempo descritos.

4.2. Modelo e Metodologia

A Figura 2 demonstra o fluxo de criação e experimentação do modelo proposto no presente trabalho. As informações da base N-BaIoT são fornecidas no formato CSV (1). Cada coluna representa uma característica estatística do tráfego do canal e as linhas são os dados coletados durante o intervalo de obtenção. Esses dados são pre-processados com um escalador máximo-mínimo a fim de garantir uma melhor performance do modelo ao limitar o intervalo dos valores (2). Após isso, os dados benignos do tráfego são usados como entrada do modelo VAE na etapa de treinamento.

O conjunto de treinamento é construído com 70% do total dos dados benignos (3). Essa abordagem tem como objetivo deixar o modelo aprender e reproduzir todas as características de tráfego do uso regular do dispositivo. A entrada do treino é composta apenas por dados benignos porque, ao conhecer como os dados devem se comportar em um cenário típico, o modelo deve ser capaz de identificar qualquer anomalia que poderia vir através da rede. Consequentemente, o VAE é uma estratégia promissora para lidar

com dados anômalos (An & Cho, 2015; Kim et al., 2020), um dos motivos principais pelos quais foi escolhido para estudar a detecção de *botnets*.

A principal diferença entre este trabalho e o proposto por Meidan et al. (2018), que usa um conjunto de *Autoencoders* profundos típicos, é a aplicação do modelo VAE. Considerando a discussão apresentada na Seção 3.2, as vantagens do VAE são: ao representar os dados como um modelo de distribuição de probabilidades, o espaço latente é regularizado e capaz de gerar saídas mais significativas no passo de decodificação, alcançar melhores resultados e obter melhor performance em dados desbalanceados; pode ser treinado e testado rapidamente, quando comparado a outras estratégias (como por exemplo, *Convolutional Neural Networks*); e também possui propriedades generativas, diferente dos AE regulares (Chung et al., 2015).

Uma vez que o passo de treino é empregado para treinar o modelo previamente com o tráfego benigno, a presente abordagem pode ser assinalada como uma estratégia semi-supervisionada. Algoritmos não-supervisionados não utilizam o passo de treinamento, de modo que o modelo aprende as características a partir dos dados de entrada por si mesmo, técnica que pode causar uma redução na acurácia quando comparado a soluções supervisionadas. Por outro lado, técnicas supervisionadas requerem que todos os dados sejam marcados com *labels*, o que pode ser um processo dispendioso. A estratégia semi-supervisionada é usada pois requer uma quantidade menor de dados marcados para treinar o modelo, e o VAE pode alcançar melhores resultados quando apresentado somente aos dados benignos para aprender de maneira apropriada a diferença entre informações normais e ataques maliciosos.

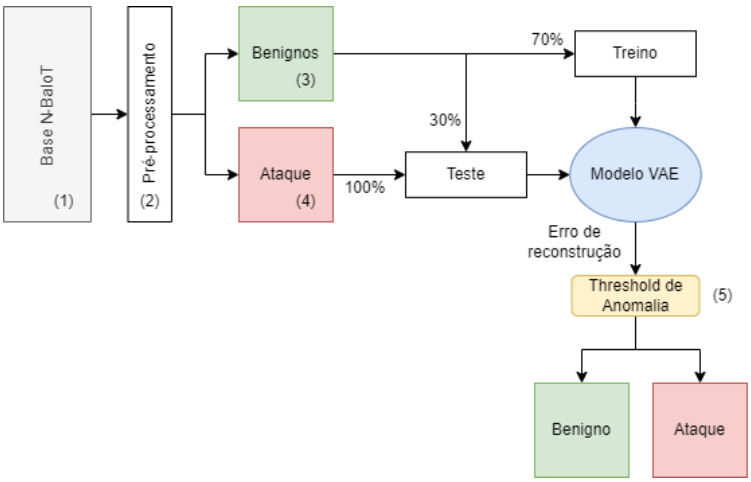

Figura 2. Diagrama de fluxo da arquitetura da metodologia empregada.

O conjunto de testes compreende os 30% restantes do tráfego benigno somado a todos os dados de ataques coletados (4). Enquanto o VAE desconstrói os dados, ele assinala altos valores ao erro de reconstrução para dados errados, os quais representam um evento de ataque. Essa avaliação torna possível diferenciar quais informações apresentam dados maliciosos, ao escolher os que possuem maiores erros de reconstrução. O limite do erro de reconstrução é dado por um hiper-parâmetro de anomalia (*threshold*), o qual define quando um dado pode ser assinalado como ataque ou não (5). Os hiper-parâmetros do modelo são posteriormente explorados na Seção 5.1.

De modo a fornecer uma avaliação completa da estratégia proposta, os experimentos foram conduzidos com as seguintes abordagens: criando tanto um modelo VAE por dispositivo quanto um modelo único que contém a informação de todos os equipamentos. Com isso, é possível identificar se a técnica funciona melhor quando treinando os dados de maneira separada ou considerando o espaço completo de dados. A segunda possibilidade é interessante pois economiza o tempo de treinar separadamente todos os modelos, além de ser um avanço em relação à metodologia proposta por Meidan et al. (2018).

Para cada um destes passos, os modelos foram treinados considerando todas as colunas de características (115 colunas) e reduzindo número de colunas (92, 69, 46 e 23 colunas, respectivamente). As primeiras colunas reduzidas são aquelas com intervalo de 1 min do tempo de aquisição, em seguida as com 10 s e assim sucessivamente, até sobrarem somente características com um tempo de aquisição de 100 ms. Este é um método atrativo para se estudar a importância de cada intervalo de aquisição no treinamento do modelo e alcançar intervalos menores de treino e teste, também sendo uma diferença em relação à metodologia de (Meidan et al., 2018).

5. Resultados e Discussão

5.1. Materiais e Métodos

Materiais. Os algoritmos e experimentos deste trabalho foram implementados com o Python 3.6.9. Todos os códigos usados nos experimentos estão disponíveis em um repositório público[1]. Todos os experimentos foram executados em um computador com Intel Core i9-9900X CPU a 3.50GHz x 20, 128 GB RAM, executando o sistema operacional Linux Ubuntu 18.04.4 LTS 64-bit.

Conjunto de Dados. Seguindo a metodologia proposta por Mirsky et al. (2018), a base de dados criada por Meidan et al. (2018) compreende informações de valores estatísticos de algumas características do canal de tráfego de nove diferentes dispositivos IoT. Os equipamentos monitorados são: campainhas das marcas Danmini e Ennio, termostato Ecobee, monitor de bebê Philips, quatro diferentes câmeras de segurança das marcas Provision e SimpleHome e uma webcam Samsung. Os dados de tráfego de cada um dos dispositivos foram coletados em situações normais de operação e quando atacados pelas *botnets* Mirai e Bashlite.

No total, 115 características foram extraídas de cada dispositivo. Cada tipo de característica (são cinco tipos no total) foi coletado considerando cinco intervalos diferentes: 1 min, 10 s, 1.5 s, 500 ms e 100 ms (L0.01, L0.1, L1, L3, e L5, respectivamente). A Tabela 1 sumariza tais tipos de características (Alqahtani et al., 2020; Meidan et al., 2018; Mirsky et al., 2018). Por exemplo, uma característica possível de existir neste conjunto de dados seria a coluna **H_L5_mean**, a qual representa a característica estatística **média** obtida com **100 ms** de intervalo e o tipo de tráfego **Host-IP**.

Tanto as *botnets* Mirai quanto Bashlite possuem diferentes tipos de ataques simulados. Os ataques da Mirai são: Ack, Scan, Syn, Udp e Udpplain. Já o Bashlite executa os ataques Combo, Junk, Scan, Udp e Tcp.

[1]O repositório com os códigos está disponível em `https://github.com/dekinks/DeepLearningSecurity`, os quais foram desenvolvidos baseados no tutorial público disponível em `shorturl.at/djxzH`.

Tabela 1. Descrição de cada característica coletada na base de dados N-BaloT.

Tipo	Acrônimo	Detalhes
Host-IP	H	Os dados de tráfego vindos de um endereço IP específico.
Host-MAC&IP	MI	Os dados de tráfego vindos de endereços IP e MAC (protocolo de acesso médio) específicos.
Channel	HH	Quando o tráfego é coletado por *hosts* específicos.
Socket	HpHp	Quando o tráfego é coletado entre *hosts* particulares que contém portas.
Network jitter	HH_jit	Um subtipo da categoria Channel, que considera o valor estatístico das características em relação ao intervalo de aquisição dos pacotes recebidos.

Hiper-parâmetros dos Modelos. Os hiper-parâmetros são parâmetros externos usados no contexto de ML para controlar o processo de aprendizado. Eles são passados nas arquiteturas e executam ações significativas para ajustar apropriadamente os modelos e predizer corretamente as respostas no passo de teste. Os parâmetros podem ser otimizados manualmente ou algoritimicamente (Gron, 2019). Para facilitar o processo manual de sintonização, os parâmetros fornecidos por Meidan et al. (2018) foram usados como um ponto de partida e modificados levemente de maneira arbitrária até se obter os melhores resultados. Um possível trabalho futuro para este projeto é otimizar automaticamente os hiper-parâmetros de forma a reduzir a quantidade de trabalho dispensada na refinação manual e obter os melhores parâmetros possíveis para a tarefa. A Tabela 2 sumariza os parâmetros utilizados durante nossos experimentos.

Tabela 2. Hiper-parâmetros usados nos modelos VAE (treino e detecção de anomalias).

Dispositivo	Taxa de Aprendizado	Limite de Anomalia	Batch	Épocas
Danmini Doorbell (DD)	0.0001	0.042	82	800
Ecobee Thermostat (ET)	0.0001	0.05	20	250
Ennio Doorbell (ED)	0.0001	0.05	22	350
Philips Baby Monitor (PB)	0.0001	0.030	65	100
P737 Security Camera (P7)	0.0001	0.035	32	300
P838 Security Camera (P8)	0.0001	0.038	43	450
S1002 Security Camera (S2)	0.00005	0.056	23	450
S1003 Security Camera (S3)	0.00008	0.01	25	500
Samsung Webcam (SW)	0.0001	0.05	32	150
Unique Model (UM)	0.0001	0.05	40	100

Métricas. As métricas escolhidas para avaliar a performance geral do método proposto são Precisão, Revocação, F1-score e Acurácia, além da análise de tempo de detecção.

5.2. Resultados na Detecção de Ataques de *Botnets*

Como apresentado na Seção 4, os experimentos deste trabalho contemplam dois cenários principais: (i) resultados obtidos para cada modelo de dispositivo e (ii) resultados obtidos do modelo único, que compreende todos os dispositivos. Para cada um desses cenários, o

número das características estatísticas usadas é decrescido com o intuito de avaliar a performance do modelo enquanto recebe informações limitadas. Desta forma, o conjunto de experimentos é o seguinte: base de dados com 115 colunas (base completa), 92 colunas (exceto informações com intervalo L0.01), 69 colunas (exceto informações com intervalo L0.01 e L0.1), 46 colunas (apenas informações com intervalo L3 e L5) e 23 colunas (apenas informações com intervalo L5). Ao escolher características que possuam tempo de aquisição reduzido e obter bons resultados, seria possível realizar a tarefa em cenários reais com baixa latência de detecção, o que é uma característica relevante a ser considerada para dispositivos IoT de borda, por exemplo.

As Tabelas 3 e 4 mostram os resultados percentuais extremos de cada conjunto de teste: possuindo a base de dados com todo o espaço de características (115 colunas) e limitada no intervalo L5 (23 colunas). Os resultados obtidos nos permitem chegar a algumas conclusões, considerando trabalhos que utilizam-se do mesmo conjunto de dados com abordagens semi-supervisionadas, não-supervisionadas e supervisionadas:

- O modelo chega a 100% de Precisão para detectar os dois tipos de dados quando treinado utilizando o espaço de características completo em 8 dos 9 dispositivos. Em média, este valor é de $99.98 \pm 0.10\%$. Tal resultado alcança o proposto por Meidan et al. (2018), cujo modelo possui 100% de Taxa de Verdadeiro Positivo (TPR) e baixa Taxa de Falso Positivo (FPR). Este é um bom resultado, pois demonstra que o modelo proposto é capaz de alcançar o estado da arte.
- O F1-score, que não é avaliado por Meidan et al. (2018), tem valor de cerca de 99% para detecção de ataques com 115 ou 23 características, tanto para modelos distintos quanto para o modelo único. O F1-score dos dados benignos é levemente menor, mas ainda acima de 90%. Isso indica que o modelo é preciso para detectar ataques, mas pode ocorrer de identificar algum tráfego benigno como sendo maligno. Este não é um grande problema, considerando que o principal propósito da estratégia é detectar os ataques corretamente.
- Os valores de Precisão alcançam e também superam alguns dos resultados obtidos por Nomm and Bahsi (2018), os quais usam uma estratégia não-supervisionada, considerando tanto os dispositivos separadamente quanto a média de todos os dispositivos. A avaliação da redução do espaço de características com base no tempo de aquisição dos dados também é um avanço em relação a esse trabalho.
- A Precisão também alcança os resultados obtidos pela metodologia supervisionada de Alqahtani et al. (2020) com o *Xtreme Gradient Boosting* e seleção de características, e de Parra et al. (2020) com a DCNN e o LSTM. Como avanço em relação a esses trabalhos, o uso de uma metodologia semi-supervisionada e de um modelo mais simples pode ser apropriado para fazer uma classificação binária, que é a diferenciação entre ataques malignos e benignos.
- Os experimentos com número de colunas reduzido apresentam bons resultados, com alguns valores atípicos (dispostivos ED e ET, especialmente). Isso indica que reduzir o espaço de características afeta a Precisão dos dados benignos e de ambas as Revocações, mas também garante uma melhora no tempo de detecção (Tabela 5). Este é um *tradeoff* importante a se considerar para escolher o melhor modelo a ser executado em cenários reais. Um tempo de execução menor poderia reduzir a latência geral do sistema, mas o operador poderia ser afetado ao receber mais dados benignos classificados como sendo de ataque.

- Utilizar o modelo único pode gerar valores suavemente menores de Precisão, Revocação e F1-score, mas pode ser considerado como um *tradeoff* a depender da tarefa do operador. Esse tipo de proposta também não foi apresentado por Meidan et al. (2018).

Tabela 3. Precisão, Revocação e F1-score quando a modelagem é feita com 115 características (os dados completos).

Modelo	Precisão (%)		Revocação (%)		F1-Score (%)	
	Benigno	Ataque	Benigno	Ataque	Benigno	Ataque
DD	100 ± 0	99.92 ± 0.01	98.33 ± 0.02	100 ± 0	99.19 ± 0.05	99.96 ± 0.01
ET	99.90 ± 0.10	99.97 ± 0.01	94.02 ± 0.20	99.99 ± 0.03	96.65 ± 0.08	99.98 ± 0.01
ED	100 ± 0	99.97 ± 0.01	99.25 ± 0.15	100 ± 0	99.61 ± 0.12	99.99 ± 0.01
PB	100 ± 0	99.88 ± 0.03	97.87 ± 0.05	100 ± 0	98.93 ± 0.06	99.94 ± 0.01
P7	100 ± 0	99.93 ± 0.01	96.61 ± 0.20	100 ± 0	98.40 ± 0.20	99.96 ± 0.02
P8	100 ± 0	99.93 ± 0.01	97.94 ± 0.10	99.99 ± 0.09	99.18 ± 0.09	99.95 ± 0.04
S2	100 ± 0	99.97 ± 0.01	98.40 ± 0.15	100 ± 0	99.25 ± 0.25	99.98 ± 0.01
S3	100 ± 0	99.90 ± 0.02	85.28 ± 0.90	100 ± 0	91.90 ± 0.20	99.95 ± 0.05
SW	100 ± 0	99.92 ± 0.01	98.33 ± 0.05	100 ± 0	99.16 ± 0.07	99.96 ± 0.01
Média	99.98 ± 0.10	99.93 ± 0.01	96.22 ± 0.2	99.99 ± 0.01	98.03 ± 0.12	99.96 ± 0.19
UM	99.77 ± 5.00	99.74 ± 0.02	90.18 ± 0.10	99.99 ± 0.10	94.73 ± 5.00	99.87 ± 0.25

Tabela 4. Precisão, Revocação e F1-score quando a modelagem é feita com 23 características (apenas dados L5).

Modelo	Precisão (%)		Revocação (%)		F1-Score (%)	
	Benigno	Ataque	Benigno	Ataque	Benigno	Ataque
DD	100 ± 0	99.99 ± 0.01	99.23 ± 0.07	100 ± 0	99.61 ± 0.05	99.99 ± 0.01
ET	98.37 ± 0.30	99.91 ± 0.01	81.80 ± 0.20	99.99 ± 0.01	89.33 ± 0.07	99.95 ± 0.01
ED	50.98 ± 0.90	99.99 ± 0.01	97.62 ± 0.18	96.53 ± 0.90	66.88 ± 0.90	96.52 ± 0.90
PB	100 ± 0	99.84 ± 0.01	97.15 ± 0.08	100 ± 0	98.55 ± 0.01	99.92 ± 0.02
P7	99.98 ± 0.02	99.71 ± 0.05	88.68 ± 0.80	100 ± 0	93.78 ± 0.60	99.85 ± 0.02
P8	99.95 ± 0.05	99.52 ± 0.06	90.14 ± 0.20	99.98 ± 0.02	94.69 ± 0.70	99.97 ± 0.01
S2	99.90 ± 0.10	99.93 ± 0.02	96.05 ± 0.09	99.97 ± 0.03	96.40 ± 0.50	99.95 ± 0.02
S3	100 ± 0	99.10 ± 0.07	87.50 ± 0.70	100 ± 0	93.34 ± 0.40	99.95 ± 0.06
SW	99.81 ± 0.10	99.91 ± 0.05	97.25 ± 0.40	99.99 ± 0.01	98.55 ± 0.07	99.93 ± 0.01
Média	94.33 ± 0.16	99.77 ± 0.03	92.82 ± 0.28	99.60 ± 0.37	92.35 ± 0.37	99.56 ± 0.12
UM	95.44 ± 0.30	99.65 ± 0.05	86.88 ± 0.04	99.89 ± 0.10	90.96 ± 0.40	99.77 ± 0.03

A métrica de Revocação não é tão explorada na literatura, mas ela apresenta um comportamento importante para a tarefa de detecção. Uma Revocação baixa pode indicar que o modelo está prevendo um grande número de Falsos Negativos, e esse comportamento não é interessante em alguns cenários. Na tarefa de detecção de *botnets*, detectar ataques com uma Revocação baixa não é interessante pois dados de ataques podem ser eventualmente classificados como sendo benignos, o que poderia causar danos ao dispositivo e à rede conectada. Os resultados obtidos apresentam Revocação em torno de 99% para informações de ataque, tanto para o conjunto de dados limitado quanto para o espaço de características completo.

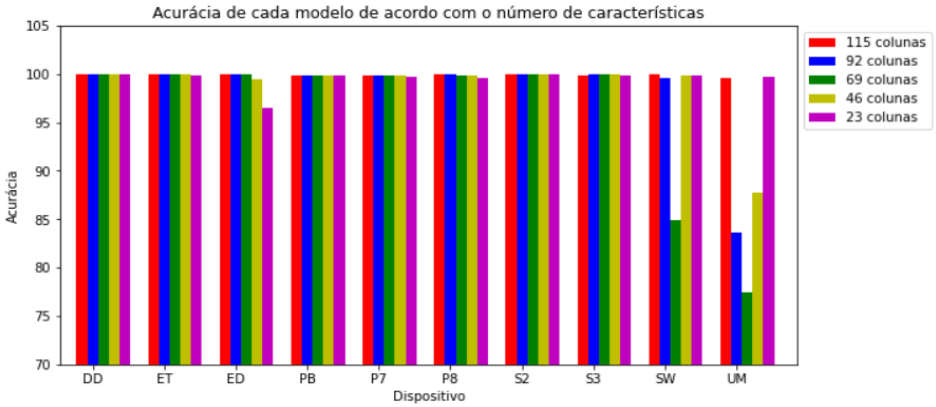

Figura 3. Valores de Acurácia (%) resultantes. Os dispositivos foram testados com diferentes quantidades de colunas para trazer um entendimento de como a métrica se comporta com a redução das características.

A Figura 3 mostra os valores de Acurácia obtidos em todos os dispositivos e no modelo único, considerando também a redução do espaço de características. Ele demonstra que é possível manter uma alta Acurácia, próxima a 100%, com os hiper-parâmetros corretos, mesmo com o número de colunas decrescendo. Este é um resultado interessante, pois mostra que a redução no espaço de características não afeta consideravelmente a Acurácia. Utilizar menos colunas para o treinamento e teste do modelo gera redução no tempo total das execuções e beneficia a etapa de detecção dos dados. Contudo, o modelo único apresenta valores altos somente em duas execuções: com 115 e 23 características, os extremos, um cenário que pode ser futuramente investigado. Os resultados da Acurácia também alcançam os obtidos por Meidan et al. (2018), Nomm and Bahsi (2018), Alqahtani et al. (2020) e Parra et al. (2020), e fornecem uma nova visão sobre o comportamento do modelo limitado.

Tabela 5. Intervalos da etapa de teste para cada dispositivo de acordo com o número de características do modelo. A última coluna indica a razão da média de tempo total dividida pelo tamanho do conjunto de teste.

Modelo	Tamanho (linhas)	Tempo de Detecção(s)					Razão (ms)
		115 c.	92 c.	69 c.	46 c.	23 c.	
DD	983,615	38.55	38.46	37.10	37.26	25.61	**0.03**
ET	826,697	34.96	30.68	32.10	26.75	30.88	**0.04**
ED	328,130	13.56	12.68	13.03	12.65	12.03	**0.04**
PB	976,010	40.70	35.90	37.34	36.88	35.76	**0.04**
P7	785,753	31.16	29.62	29.46	24.49	27.52	**0.04**
P8	767,932	33.31	31.87	29.54	25.54	28.24	**0.04**
S2	830,447	32.72	33.99	31.15	31.84	30.43	**0.04**
S3	837,157	33.83	31.93	33.22	26.01	28.10	**0.04**
SW	228,717	14.23	12.91	11.93	12.85	12.34	**0.06**
UM	6,673,458	262.79	248.82	245.50	233.73	226.85	**0.04**

Por fim, a Tabela 5 mostra o tempo gasto na etapa de detecção para cada quantidade de características e modelo (de 115 a 23 colunas). O tempo do passo de teste é mais

relevante de ser avaliado pois representa o momento exato em que a informação seria coletada e classificada em um cenário real. A coluna Razão indica que, no geral, o modelo gastaria cerca de 0.04 ms para classificar um pacote vindo da rede, um tempo menor do que o obtido por Meidan et al. (2018). Esse comportamento indica que o modelo proposto poderia ser implementado em um ecossistema IoT sem afetar a latência da rede.

6. Conclusão e Trabalhos Futuros

O trabalho implementa uma estratégia em Aprendizado Profundo para executar a tarefa de detecção de ataques a partir de informações de tráfego de dispositivos IoT. Este estudo é importante para descobrir novos métodos para melhorar a segurança dos dados recebidos por dispositivos IoT e garantir ambientes de alta performance aos usuários, com maior confiança e menor custo computacional. Os métodos de DL mostraram-se adequados para a tarefa, pois permitem melhores resultados que os métodos de Aprendizado de Máquina convencionais, já que possuem alto desempenho e lidam com informações complexas (Aversano et al., 2021).

A estratégia de DL proposta é o *Variational Autoencoder*, modelo escolhido pois pode ser usado de maneira semi-supervisionada, de modo a reduzir o custo da etapa de treinamento, e permitir uma rápida etapa de teste, como demonstrado na Seção 5. As VAEs também são levemente menos complexas do que algumas outras metodologias de DL, como CNNs e RNNs, as quais usualmente requerem um grande número de camadas conectadas para gerar bons resultados. Elas também podem alcançar melhores respostas do que os Autoencoders regulares.

Os experimentos conduzidos no conjunto de dados N-BaIoT conseguem alcançar e até superar alguns resultados obtidos na literatura (Meidan et al., 2018; Parra et al., 2020; Nomm & Bahsi, 2018; Alqahtani et al., 2020). Os experimentos foram construídos de maneira a considerar um modelo por dispositivo e um modelo único com informações de todos os dispositivos e espaço de características reduzido. Tais abordagens são interessantes para entender o *tradeoff* ao reduzir levemente o desempenho do algoritmo, mas ganhar melhores resultados no tempo de detecção e simplificar a geração do modelo.

Em geral, o modelo VAE proposto se provou uma boa opção para executar tarefas de detecção em dispositivos IoT, a julgar pela simplicidade do modelo e pelos resultados obtidos. Como trabalhos futuros, seria interessante analisar a otimização dos hiperparâmetros de forma automática, aplicar e avaliar o modelo em conjuntos de dados diferentes e utilizar algum método de seleção de características, com a finalidade de usar as colunas que fornecem informações mais relevantes. Os resultados do tempo de detecção ainda corroboram com a investigação de aplicar o modelo VAE em dispositivos de borda, os quais possuem recursos computacionais reduzidos e requerem baixa latência.

Referências

Alqahtani, M., Mathkour, H., & Ismail, M. M. B. (2020). Iot botnet attack detection based on optimized extreme gradient boosting and feature selection. *Sensors (Switzerland)*, *20*, 1-21.

Alsoufi, M. A., Razak, S., Siraj, M. M., Nafea, I., Ghaleb, F. A., Saeed, F., & Nasser, M. (2021). Anomaly-based intrusion detection systems in iot using deep learning: A systematic literature review. *Applied Sciences (Switzerland)*, *11*.

An, J., & Cho, S. (2015). Variational Autoencoder based Anomaly Detection using Reconstruction Probability. *Special Lecture on IE*, *2(1)*, 1-18.

Aversano, L., Bernardi, M. L., Cimitile, M., & Pecori, R. (2021). A systematic review on deep learning approaches for iot. *Computer Science Review*, *40*.

Borges, J. B., Medeiros, J. P. S., Barbosa, L. P. A., Ramos, H. S., & Loureiro, A. A. (2022). Iot botnet detection based on anomalies of multiscale time series dynamics. *IEEE Transactions on Knowledge and Data Engineering*.

Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A. C., & Bengio, Y. (2015). A recurrent latent variable model for sequential data. In *Advances in neural information processing systems* (Vol. 28).

Gron, A. (2019). *Hands-on Machine Learning with Scikit-Learn, Keras and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Inc.

Hussain, F., Hussain, R., Hassan, S. A., & Hossain, E. (2020). Machine Learning in IoT Security: Current Solutions and Future Challenges. *IEEE Communications Surveys and Tutorials*, *22*, 1686-1721.

Kim, J., Sim, A., Kim, J., & Wu, K. (2020). Botnet detection using recurrent variational autoencoder. In *Globecom 2020 - ieee global communications conference* (p. 1-6).

Kingma, D. P., & Welling, M. (2014). Auto-Encoding Variational Bayes. *International Conference on Learning Representations*.

Kingma, D. P., & Welling, M. (2019). An Introduction to Variational Autoencoders. *Foundations and Trends in Machine Learning*.

Kolias, C., Kambourakis, G., Stavrou, A., & Voas, J. (2017). DDoS in the IoT: Mirai and Other Botnets. *Computer*, *50*, 80-84.

Laguduva, V. R., Islam, S. A., Aakur, S., Katkoori, S., & Karam, R. (2019). Machine Learning Based IoT Edge Node Security Attack and Countermeasures. *Proceedings of IEEE Computer Society Annual Symposium on VLSI, ISVLSI*, 670-675.

Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A., & Lloret, J. (2017). Conditional variational autoencoder for prediction and feature recovery applied to intrusion detection in iot. *Sensors*, *17*.

Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Breitenbacher, D., Shabtai, A., & Elovici, Y. (2018). N-baiot: Network-based detection of iot botnet attacks using deep autoencoders. *IEEE Pervasive Comput.*, *17*, 12-22.

Mirsky, Y., Doitshman, T., Elovici, Y., & Shabtai, A. (2018). Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection. In *Proc. of network and distributed system security symposium* (pp. 1–15). Reston, VA: Internet Society.

Nomm, S., & Bahsi, H. (2018). Unsupervised anomaly based botnet detection in iot networks. *Proceedings - 17th IEEE International Conference on Machine Learning and Application*, 1048-1053.

Parra, G. D. L. T., Rad, P., Choo, K.-K. R., & Beebe, N. (2020). Detecting internet of things attacks using distributed deep learning. *Journal of Network and Computer Applications*, *163*.

Song, Y., Hyun, S., & Cheong, Y.-G. (2021). Analysis of autoencoders for network intrusion detection. *Sensors*, *21*.

Yang, Y., Zheng, K., Wu, C., & Yang, Y. (2019). Improving the classification effectiveness of intrusion detection by using improved conditional variational autoencoder and deep neural network. *Sensors*, *19*.