Um Sistema Autônomo para a Predição de Ataques de DDoS em Redes Locais e Internet

Davi Brito[1], Anderson B. de Neira[2], Ligia F. Borges[1], Alex M. de Araújo[2], Michele Nogueira[1 2]

[1]Departamento de Ciência da Computação - Universidade Federal de Minas Gerais

[2]Departamento de Informática - Universidade Federal do Paraná

{davibrito,ligia.borges,michele}@dcc.ufmg.br

{abneira,amaraujo}@inf.ufpr.br

Resumo. *Diante da constante evolução dos ataques de negação de serviço distribuído (DDoS), torna-se necessário o desenvolvimento de técnicas de predição para confrontar essas ameaças. A dissimulação dos ataques e o alto volume de tráfego de rede dificultam o seu reconhecimento. Portanto é importante identificar a preparação dos ataques para aumentar o tempo hábil de combatê-los. Este artigo apresenta um sistema autoadaptativo para a predição dos ataques DDoS. O sistema define automaticamente a melhor configuração da rede neural para distinguir a preparação do ataque DDoS do tráfego normal. Os resultados indicam que o sistema consegue predizer um ataque DDoS em 29 minutos antes do seu início com acurácia superior à literatura (97,89%).*

Abstract. *Due to the constant evolution of distributed denial of service (DDoS) attacks, developing prediction techniques to confront these threats is necessary. The concealment of attacks and the high volume of network traffic make it difficult to recognize them. Therefore, it is important to identify the preparation of attacks to increase the time to combat them. This paper presents a self-adaptive system for predicting DDoS attacks. The system automatically defines the best neural network configuration to distinguish DDoS attack preparation from normal traffic. The results show that the system predicts a DDoS attack 29 minutes before its start with accuracy higher than the literature (97.89%).*

1. Introdução

Pessoas mal-intencionadas criam ataques cada vez mais elaborados explorando fraquezas nos sistemas e na rede para atingir suas vítimas. Existem vários tipos de ameaças cibernéticas, sendo o ataque de negação de serviço distribuído (do inglês, *Distributed Denial of Service* — DDoS) um dos mais custosos [Jyoti & Behal 2021]. A constante evolução e a escalada do volume dos ataques DDoS tornam rapidamente obsoletas as soluções de detecção de ataque DDoS. Tal como um ataque DDoS superou 60 milhões de requisições por segundo em menos de 30 segundos [Lakshmanan 2023]. Assim, realizar apenas a detecção dos ataques não é suficiente para evitar os danos causados por eles.

A Inteligência Artificial (IA) é uma área proeminente para o desenvolvimento de soluções de cibersegurança. O *Deep Learning* (DL) vem ganhando atenção devido à diversidade de arquiteturas com capacidade de se adaptar a diferentes ataques. Contudo a

implantação bem-sucedida de algoritmos de Aprendizado de Máquina (AM) requer grandes esforços de especialistas [Arp et al. 2020]. É necessário escolher um candidato entre um conjunto de diversas possíveis soluções, considerando as informações do ambiente de implantação. Além disso, realizar os ajustes dos parâmetros da solução para que ela obtenha um bom desempenho demanda tempo e conhecimento [Feurer et al. 2015]. Esse processo é ainda mais complexo em ambientes com desbalanceamento de classes. Os cenários com ataques cibernéticos são intrinsecamente desbalanceados, pois em um momento possuem mais instâncias de ataque e em outro, mais instâncias legítimas.

Várias soluções para a predição de ataque DDoS demandam a correta parametrização. Anuar *et al.* (2018) utilizaram a Rede Neural Artificial (do inglês, *Artificial Neural Network* — ANN) para prever ataques DDoS. Os autores utilizaram uma ANN com uma camada de entrada, uma intermediária e uma de saída. Contudo a arquitetura da ANN possui outros parâmetros, como a quantidade de neurônios e os pesos. Além do tempo gasto em avaliar todas as arquiteturas, é possível que a encontrada não seja ideal para outros cenários. Assim, identificar a arquitetura correta da ANN é um desafio que requer também ajustes dependendo do contexto. Ali e Al-Shaer (2013) propuseram a predição de ataques DDoS utilizando as cadeias de Markov para prever o próximo estado da rede usando o estado atual como entrada. Contudo configurar corretamente a cadeia de Markov para cada cenário não é trivial.

Para lidar com este problema, propõe-se um sistema para predizer ataques DDoS usando o DL e o gerenciamento autônomo da solução. A partir de esforços de especialistas em AM, a literatura definiu o processo denominado de *Automated Machine Learning* (AutoML), cujo objetivo é democratizar o uso de AM. O AutoML encontra e configura um algoritmo de AM que reduz os erros de classificação para um conjunto de dados [Arp et al. 2020]. O sistema utiliza técnicas de AutoML para construir e configurar redes neurais para realizar a predição. Assim, poupa-se tempo nessa tarefa, adquirindo, simultaneamente, resultados competitivos frente à literatura, no que tange ao tempo de processamento e à predição de ataques DDoS.

A proposta foi avaliada seguindo dois experimentos. O primeiro usa o conjunto de dados CTU-13, referente à rede local da Universidade Tcheca [Garcia et al. 2014]. O segundo experimento usa a base de dados *DDoS Evaluation Dataset* (CIC-DDoS2019) [Sharafaldin et al. 2019], onde a vítima e os atacantes estão em redes distintas ligadas pela Internet. No primeiro experimento, a proposta predisse o ataque 29 minutos e 51 segundos antes do seu lançamento, com acurácia de 97,89%. No segundo experimento, o ataque foi antecipado com 15 minutos e 21 segundos antes de o atacante lançá-lo, com acurácia de 85,61%. A análise mostrou que a proposta se adapta autonomamente aos *datasets* para realizar a predição dos ataques.

Este artigo contribui para a predição de ataques DDoS e demonstra a aplicabilidade do processo AutoML. Os resultados trazem avanços para o campo de pesquisa e beneficiam o gerenciamento da rede ao reduzir o tempo gasto com processos de seleção e configuração dos diversos componentes da arquitetura das redes neurais. Quando manual, esse processo demanda tempo e experiência. Portanto a proposta automatiza o procedimento considerando o comportamento da rede onde está inserido. Ademais, este trabalho democratiza a utilização de AM, em especial o DL, para a predição de ataques DDoS. Isso é benéfico pois a solução proposta alcança públicos não especialistas em AM.

O trabalho proposto fornece à literatura as primeiras evidências de que os ataques DDoS podem ser preditos por modelos configurados autonomamente sem a dependência de dados rotulados para realizar o treinamento dos algoritmos. A utilização de dados rotulados é custosa e limita a generalização da solução frente a abundância de ataques DDoS existentes. A automatização da seleção da arquitetura de DL supera os resultados obtidos na literatura [Silva et al. 2022]. Em Silva *et al.* (2022), os autores obtiveram 91,42% de acurácia definindo manualmente todos os componentes da solução. No mesmo *dataset*, este trabalho obteve 97,53% de acurácia selecionando e configurando autonomamente a arquitetura de rede neural. Vale destacar que este trabalho é pioneiro na automatização da seleção de arquiteturas de DL para predizer ataques DDoS em redes locais e na Internet.

Este artigo procede da seguinte forma. A Seção 2 apresenta os trabalhos relacionados com o AutoML e com a predição de ataques DDoS. A Seção 3 detalha a proposta. A Seção 4 apresenta os resultados obtidos. Por fim, a Seção 5 conclui este trabalho.

2. Trabalhos Relacionados

As técnicas de AutoML ajudam as soluções a se adaptarem a diferentes tipos de ataques. Ioulianou *et al.* (2022) utilizaram o processo AutoML para combater ataques de roteamento do tipo *blackhole*. No ataque *blackhole*, o invasor controla o nó malicioso e determina o descarte de todos os dados que recebe. Ou seja, os nós legítimos tentam retransmitir os dados, mas, durante o ataque, o nó malicioso descarta todos os dados que recebe, prejudicando o correto funcionamento da rede [Ioulianou et al. 2022]. Os autores verificaram que a solução proposta consegue encontrar diferentes arquiteturas de DL para diferentes conjuntos de dados. Ioulianou *et al.* (2022) atingiram a acurácia de 97,91% classificando tráfego normal e malicioso para detectar o ataque *blackhole*. Contudo os autores não apresentam métricas complementares, como precisão e *recall*. Portanto não é possível avaliar se a solução proposta lida com o problema adequadamente.

A literatura mostra que o processo AutoML auxilia na detecção de ataques DDoS. Horsanali *et al.* (2021) propuseram um *framework* de AutoML próprio para a detecção desses ataques. Eles usaram seis algoritmos de AM: regressão logística, *k-nearest neighbors*, *Support Vector Machine*, *Naive Bayes*, *Decision Tree* e o *Random Forest*. Quando recebe dados para o treinamento, ele os pré-processa, seleciona as características, treina e testa os seis algoritmos. O *framework* de AutoML seleciona o algoritmo que obtiver a melhor acurácia. Contudo nesse contexto, usar a acurácia como função de avaliação quando as classes dos dados são desbalanceadas não é o ideal. A acurácia pode privilegiar modelos que classificam corretamente somente o tráfego normal, não identificando o tráfego de ataque. Para que a solução de detecção de ataque DDoS funcione, é necessário que o ataque esteja consumindo os recursos da vítima. Isso pode impactar negativamente na utilização dos usuários reais. Por fim, como a solução detecta o ataque apenas após o seu lançamento, o tempo para interrompê-lo pode não ser suficiente.

A predição de ataques DDoS é um problema de pesquisa pouco abordado na literatura. Em Santos *et al.* (2013), uma solução para monitorar textos de mídias sociais foi proposta para identificar um potencial alerta de ataque e antecipar problemas em redes de computadores. A solução filtra postagens relacionadas com ataques DDoS. Assim, os atacantes devem sinalizar o ataque, como, por exemplo, postar uma mensagem contendo discurso de ódio contra a vítima (*e.g.*, pessoa, organização privada ou pública) ou utili-

zar termos referentes ao ataque. Machaka *et al.* (2021) compararam três algoritmos de regressão para prever ataques DDoS. Os algoritmos comparados são *Logistic Regression* (LGR), *Support Vector Regression* e *Kernel Ridge Regression*. Para realizar a comparação, os autores utilizaram o tráfego de rede do conjunto de dados DARPA 1999 e, em alguns momentos, aumentaram manualmente o número de pacotes para introduzir tráfego de ataque DDoS. Os autores treinaram os algoritmos de regressão usando o número de pacotes contidos a cada 10 segundos de tráfego em 80% do conjunto de dados, e avaliaram a solução com os 20% restantes. O LGR obteve os melhores resultados, alcançando uma acurácia de 98,60% e prevendo ataques DDoS com 15 minutos de antecedência.

A literatura foca na predição de ataques criando soluções baseadas em algoritmos e modelos que não se adaptam a novos contextos (*i.e.*, mudanças que ocorrem na distribuição dos dados analisados). Isso causa a obsolescência prematura dessas soluções a cada ataque diferente, pois altera o padrão para o qual o modelo foi criado e calibrado. Esse problema advém da dificuldade de selecionar corretamente a arquitetura e os diversos parâmetros que os algoritmos de AM possuem, em especial os algoritmos de DL. Desta forma, este trabalho propõe uma abordagem de predição de ataques DDoS que seleciona e configura autonomamente as arquiteturas de redes neurais empregadas. Assim, o administrador de rede não precisa se preocupar com essas tarefas para realizar a predição.

3. Sistema Proposto

A Figura 1 ilustra o processo executado pelo sistema proposto. O processo é dividido em cinco etapas e utiliza o DL selecionado e configurado autonomamente para predizer ataques DDoS: (1) captura de tráfego de rede; (2) engenharia de características; (3) seleção e configuração autônoma do modelo; (4) predição do ataque; e (5) notificação aos administradores. Para o completo entendimento do sistema, a Subseção 3.1 descreve a captura de tráfego da rede. A Subseção 3.2 detalha o pré-processamento dos dados. A Subseção 3.3 especifica o treinamento para seleção autônoma do modelo AM. A Subseção 3.4 apresenta a identificação de sinais da orquestração de ataques DDoS. Finalmente, a Subseção 3.5 descreve como ocorre a notificação do iminente lançamento de um ataque.

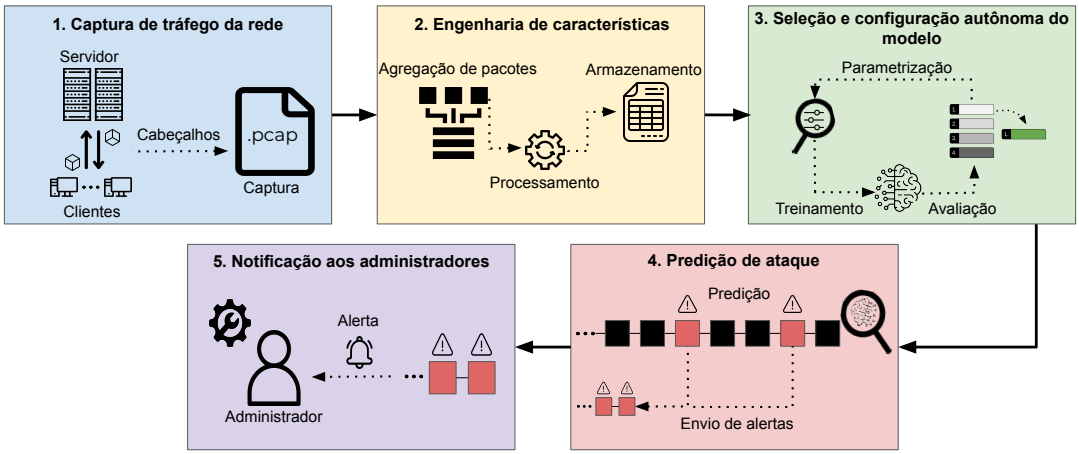


Figura 1. Arquitetura do sistema proposto

3.1. Captura de Tráfego da Rede

A coleta do tráfego de rede é a primeira ação realizada pelo sistema. Os dados coletados são utilizados para treinar o modelo de aprendizado profundo e para predizer os ataques. A captura de tráfego ocorre por uma ferramenta integrada ao *firewall* que intercepta a entrada e saída da rede do usuário e extrai os cabeçalhos (*headers*) dos pacotes transmitidos. Dependendo das características do tráfego de rede definidas pelo usuário (Subseção 3.2), o sistema pode não coletar dados de um protocolo específico. Caso o usuário defina que o sistema deve utilizar apenas características do tráfego de rede baseadas em *User Datagram Protocol* (UDP), o sistema pode desprezar os outros protocolos. Essa ação é importante para, quando possível, economizar recursos computacionais. A coleta do tráfego de rede independe da topologia de rede conforme demonstrado na Subseção 4.1. Os experimentos consideram ataques conduzidos em redes locais e ataques conduzidos onde a rede vítima está ligada à rede atacante pela Internet.

Os cabeçalhos coletados são, então, armazenados em um servidor dedicado. Os dados coletados são exportados para arquivos do tipo Captura de Pacotes (do inglês, *Packet Capture* — PCAP) para possibilitar a extração das características da rede (Subseção 3.2). O volume máximo da captura pode ser definido pela quantidade de pacotes ou por tempo da captura. Assim, cada PCAP contém um conjunto fixo de pacotes ou todos os pacotes trafegados em um limite de tempo. O limite é de 50 mil pacotes e o valor padrão para o tempo é um segundo. O administrador de rede tem a opção de alterar esses parâmetros da maneira que melhor lhe atenda. Cada PCAP é salvo com a data do início da coleta. Então, o sistema consegue ordenar as capturas e identificar as mais novas. Para melhorar o gerenciamento da capacidade de armazenamento, evitando a sobrecarga do servidor, é necessário a remoção dos dados antigos quando um limite de armazenamento é atingido. Pode-se definir o valor ideal baseado na disponibilidade de hardware. A ação padrão é começar a remover os dados antigos quando o espaço restante for inferior a 20%. Uma vez concluída a coleta, os dados são pré-processados na segunda etapa.

3.2. Engenharia de Características

A primeira ação da Etapa 2 é a extração das características do tráfego de rede. O sistema monitora o servidor de arquivos em busca de novas capturas e extrai os atributos dos cabeçalhos dos pacotes (*e.g.*, tamanho em *bytes* e a quantidade de pacotes). A definição dos atributos do tráfego de rede também pode ser customizada. Contudo o sistema foi projetado para ser independente dos atributos empregados pelos administradores de rede. O imperativo para a abordagem funcionar é que os atributos devem ser afetados pela preparação do ataque. O atacante pode realizar testes antes de iniciar o ataque [Jaber et al. 2017]. Atributos como a quantidade de dispositivos únicos trocando pacotes e quantidade de pacotes trafegados podem ser influenciados pela preparação dos ataques. Feng *et al.* (2018) identificaram quarenta atributos representativos para a detecção de comunicação de Comando e Controle (C&C), *i.e.*, mensagens enviadas pelo atacante ao *BotMaster* (computador com código malicioso) e do *BotMaster* aos dispositivos infectados (*bots*). Os atributos apresentados pelos autores podem ser usados para a predição de ataques DDoS, uma vez que a comunicação de C&C ocorre antes do ataque. Este trabalho não realiza a seleção de atributos, pois o DL não exige sua seleção manual.

Para realizar a extração das características do tráfego de rede, o sistema agrega as capturas por unidade de tempo (*e.g.*, minutos, segundos e milissegundos). Definir o

valor adequado de agregação das capturas é fundamental. Valores pequenos não são significativos para realizar a predição do ataque devido à quantidade reduzida de informação disponível. Por outro lado, os valores grandes resultam em maior tempo de processamento, prejudicando a predição do ataque DDoS. O valor padrão adotado pelo sistema é um segundo, podendo ser alterado pelo usuário. A agregação mantém as características do tráfego de rede ordenadas sequencialmente ao longo do tempo (*i.e.*, uma série temporal para cada atributo). Após a extração das características do tráfego de rede, o sistema lida com dados em escalas diferentes para evitar a degradação do desempenho. Para isso, o sistema disponibiliza a padronização, a estratégia *RobustScaler* e o dimensionamento Mín-Máx. A padronização (Eq. 1) dimensiona os atributos removendo a média das observações dividido pelo desvio padrão das observações.

$$Padronização = \frac{(X(j) - \bar{x})}{s}, \tag{1}$$

em que $X(j)$ representa o valor de cada amostra do atributo analisado, o termo $\bar{x}$ representa a média e o termo s desvio padrão. A estratégia *RobustScaler* (Eq. 2) utiliza a mediana do conjunto de atributos para diminuir a observação analisada. O resultado dessa operação é dividido pelo valor da amplitude interquartil, representada pela subtração do terceiro com o primeiro quartil.

$$RobustScaler = \frac{X(j) - M_d}{Q_3(x) - Q_1(x)}, \tag{2}$$

onde $X(j)$ é o valor de cada amostra do atributo analisado, o termo M_d representa a mediana, o $Q_1(x)$ e o $Q_3(x)$ representam o primeiro e o terceiro quartis respectivamente. A Mín-Máx é a última estratégia do sistema para dimensionamento dos dados (Eq. 3). A estratégia Mín-Máx dimensiona os valores dos atributos entre um intervalo pré-definido. O valor do intervalo é entre zero e um ou, de forma absoluta, os valores reais da base.

$$M\acute{i}n - M\acute{a}x = \frac{X(j) - min_A}{max_A - min_A}(max'_A - min'_A) + min'_A, \tag{3}$$

em que $X(j)$ representa o valor de cada amostra do atributo analisado, o min_A e o max_A são respectivamente o menor e o maior valor observado para o atributo. O min'_A e o max'_A representam o novo intervalo. Por fim, o administrador de rede pode optar por não utilizar nenhuma estratégia de redimensionamento dos dados.

O sistema transforma as características em sinais de alerta precoces. Para isso são aplicadas as medidas estatísticas de *Skewness*, *Kurtosis* e coeficiente de variação. Essas medidas estatísticas baseiam-se no conceito de séries temporais (*i.e.*, constituída de observações realizadas sequencialmente no tempo). *Kurtosis* é uma medida que caracteriza o achatamento da curva de uma distribuição. Essa métrica indica o quanto uma variável se encontra nas caudas da distribuição. O cálculo da *Kurtosis* segue:

$$Kurtosis = \frac{(N_t - 1)}{(N_t - 2)(N_t - 3)}(N_t - 1)\hat{y} + 6, \tag{4}$$

$$\hat{y} = \frac{N_t \sum (x_t - \bar{x})^4}{[\sum (x_t - \bar{x})^2]^2} \tag{5}$$

Skewness (Eq. 6) é uma medida de assimetria de uma distribuição de probabilidade idealmente simétrica. Essa métrica indica o quanto a distribuição de probabilidade de uma variável aleatória desvia da sua distribuição normal.

$$Skewness = \frac{N \sum_{t=1}^{N} (x_t - \bar{x})^3}{(N-1)(N-2)s^3}, \tag{6}$$

O termo x_t refere-se a cada amostra da série temporal, N é a quantidade total de itens observados, $\bar{x}$ é a média aritmética simples e o termo s representa o desvio padrão. O Coeficiente de Variação (CV) é utilizado para analisar a dispersão em termos relativos ao seu valor médio. O CV permite comparar séries de valores que apresentam unidades de medida distintas. Para isso, o desvio padrão dos dados analisados (σ) é dividido pela média dos dados ($\bar{x}$), onde $\bar{x} \neq 0$ [Bedeian and Mossholder 2000]. O CV é utilizado como indicador de diversidade em relação à média dos conjuntos de dados. Assim, o limite inferior de CV ($cv = 0$) indica uniformidade completa do conjunto.

As oscilações e as perturbações afetam a distribuição de dados e geram uma variação. Um exemplo de variação é a transição crítica que ocorre quando o sistema transita entre pontos de equilíbrio. O cálculo estatístico sobre séries temporais de atributos auxilia na identificação de sinais que antecedem transições críticas [Bury et al. 2021]. Dessa forma, as características do tráfego de rede são transformadas em sinais por meio da *Skewness*, *Kurtosis* e CV, visando identificar transições críticas durante a orquestração dos ataques e assim predizer o seu lançamento. Na transição de estado é possível observar um aumento ou redução na *Skewness* de uma série temporal. Similarmente, as oscilações fazem com que o estado de um sistema atinja valores extremos próximos a uma transição, levando a um aumento na *Kurtosis* de uma série temporal anterior à transição. O CV também pode ser utilizado como um sinal precoce de alerta, pois aumentos no CV podem indicar a ocorrência de uma transição crítica [Carpenter & Brock 2006, Dakos et al. 2012]. Dessa forma, a proposta prediz ataques com base em tráfego de rede. Considera-se o evento futuro de ocorrência de ataque quando as características estatísticas (*Kurtosis*, *Skewness* e CV) variam, forçando o sistema a apresentar erros na reconstrução.

3.3. Seleção Autônoma do AM

Na Etapa 3, o sistema treina uma rede neural autonomamente para a predição de ataques DDoS. Existe uma vasta diversidade de algoritmos e arquitetura de AM com características únicas que podem se adaptar a diferentes volumes de dados. Contudo selecionar e configurar a arquitetura adequada não é uma tarefa trivial. Visando democratizar o uso da aprendizagem de máquina, especialistas reuniram esforços para desenvolver o processo denominado de AutoML. O AutoML visa encontrar e configurar um algoritmo de AM para minimizar erros de classificação para a base de dados selecionada pelo usuário do *framework* [Feurer et al. 2015]. Dado o potencial de automatização do AutoML, esse recurso é utilizado para evitar erros no processo de seleção, configuração e treinamento do modelo capaz de predizer os ataques DDoS sem a interação humana. O AutoML sugere o algoritmo mais adequado para o contexto da rede, determinando o conjunto de algoritmos de AM ou a arquitetura de rede neural mais adequada. Assim, são utilizados os dados pré-processados para o treinamento da rede neural com seleção de camadas.

A Figura 2 mostra o funcionamento dos *frameworks* de AutoML. No Passo 1, é definido o espaço de pesquisa com um conjunto de arquiteturas de redes neurais [Lam & Abbas 2020].As arquiteturas candidatas podem variar entre diferentes estruturas (*i.e.*, números de camadas ocultas, pesos e quantidade de neurônios). Cada *framework* de AutoML define seu espaço de busca. O Passo 2 configurara um subconjunto ou todos os algoritmos de AM candidatos. Este passo usa algum processo de otimização para o *framework* de AutoML avaliar as combinações de configuração dos algoritmos e arquitetura. No passo 3, o *framework* de AutoML treina e testa algoritmos configurados usando o

conjunto de dados selecionado pelo usuário do *framework*. A acurácia é um critério para avaliar algoritmos de AM. No entanto, é comum a escolha de outros critérios, como precisão, *recall* ou *F1-score* para privilegiar modelos mais equilibrados.

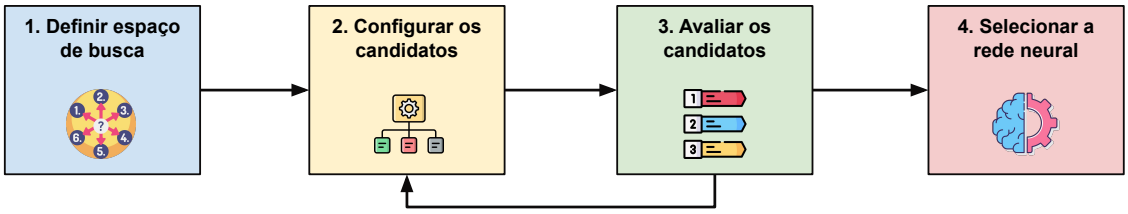


Figura 2. Operação geral dos *frameworks* de AutoML

Ao final do passo de avaliação, o *framework* de AutoML retorna ao Passo 2 para que os algoritmos de AM recebam novas configurações para maximizar os critérios de avaliação (resultados). O ciclo entre os Passos 2 e 3 se repete até que o *framework* de AutoML encontre um critério de parada. Este critério pode ser o tempo de execução ou o número de iterações. No Passo 4, o *framework* de AutoML escolhe a arquitetura de AM que maximiza os critérios de avaliação. Para restringir o espaço de busca e diminuir o tempo para selecionar e configurar a arquitetura da rede neural, definiu-se que apenas seriam avaliadas as redes neurais do tipo *Long Short-Term Memory* (LSTM) *Autoencoder*.

A LSTM é uma rede neural que se diferencia das outras pois diminui o problema de dissipação do gradiente. O algoritmo *backpropagation* é utilizado para o treinamento de redes neurais, por atualizar os pesos dos neurônios baseado no erro da saída do modelo. Contudo ao longo do tempo de treinamento, o algoritmo *backpropagation* acaba dissipando o valor do erro e assim não consegue atualizar os pesos das camadas, principalmente as iniciais. O LSTM evita o problema de dissipação do gradiente por meio de Carrosséis de Erro Constante (CECs). Os CECs são aplicados em células especiais para criar um fluxo de erro constante. O *backpropagation* obtém acesso a essas células especiais por intermédio de um portão multiplicativo, que aprendem quando devem conceder acesso [Staudemeyer & Morris 2019]. O *Autoencoder* consiste em uma rede neural composta por camadas de entrada e saída, um espaço latente, uma rede neural codificadora e uma rede neural decodificadora. Esta rede neural visa compreender como codificar e decodificar os dados selecionados pelo usuário da rede. Para isso, o *Autoencoder* comprime os dados no espaço latente, para em seguida decodificar os dados de saída do modelo. O *Autoencoder* compara a saída processada com os dados originais do usuário. O erro entre o predito e o real é utilizado para atualizar os pesos da rede [Nguyen et al. 2021].

Este trabalho assume uma rede LSTM *Autoencoder*. Assim, tanto o codificador quanto o decodificador são a rede LSTM [Nguyen et al. 2021]. A principal vantagem da rede é que o treinamento pode ser conduzido sem a utilização de dados rotulados e somente com dados. Isso facilita a adoção da proposta em ambientes reais por diminuir o custo da aquisição dos rótulos dos dados. Assim, o administrador de rede inicia a coleta do funcionamento da rede e o sistema identifica mudanças com comportamento dos sinais que representem a ocorrência de um ataque DDoS futuro. Este trabalho selecionou o LSTM *Autoencoder* para facilitar a utilização do sistema proposto em ambientes reais. Assim, o objetivo da Etapa 4 é automatizar a configuração de cada aspecto da arquitetura do LSTM *Autocoder* para se adaptar a diferentes ataques DDoS sem a interação humana.

3.4. Predição de ataques DDoS

Na Etapa 4, a rede neural configurada autonomamente processa os dados das novas coletas de rede e tenta reconstruí-los. Para isso, a rede neural comprime os sinais (*i.e.*, *Skewness*, *Kurtosis* e CV) nas primeiras camadas do LSTM *Autoencoder*. Na sequência, a rede neural decodifica esses sinais para reconstruí-los. O sistema compara os valores reais do *Skewness*, *Kurtosis* e CV sobre o tráfego de rede com os valores dos sinais reconstruídos e usa a diferença absoluta entre eles para obter o valor do erro de reconstrução. O erro é calculado para o *Skewness*, *Kurtosis* e CV, individualmente. O sistema calcula a média aritmética dos erros para obter apenas um indicador de erro total. Esse processo é repetido sempre que existirem novas coletas de rede.

A predição do ataque DDoS acontece quando o erro total de reconstrução for maior que um limiar previamente definido. O sistema define o limiar com base no percentil do erro de reconstrução no treinamento. O percentil indica os valores da amostra que são iguais ou menores a um determinado valor. O sistema define 97,9% como o valor padrão para o percentil, isso significa que o limiar é definido com base no valor do erro da menor amostra fora do grupo do percentil 97,9. Esse valor padrão foi definido empiricamente com base em vários testes realizados antes da avaliação (Seção 4). Contudo o percentil pode ser modificado a critério do administrador de rede para um controle mais fino das predições. Para um sistema mais rígido, basta usar um percentil maior. Para maior flexibilidade do sistema e maior quantidade de alerta, basta reduzir o percentil. Contudo pode-se optar pela não utilização de limiares, assim o sistema informa o erro de reconstrução e o usuário decide quais ações tomar.

3.5. Notificação de Ataque

O sistema utiliza a saída da etapa de identificação de sinais de alerta para emitir notificações sobre a ocorrência de possíveis ataques DDoS (Etapa 5). Para esse processo, pode ser utilizado uma interface de programação de aplicativos web para a transferência de dados, que podem, inclusive, ser utilizados como entrada para automatizar outra solução de cibersegurança ou ainda para envio de mensagens de texto instantâneos para os administradores da rede. Além disso, sempre que o sistema predizer um ataque, ele pode notificar a equipe responsável por meio de e-mail ou mensagens.

4. Avaliação

A avaliação do sistema proposto segue dois experimentos. Para a avaliação, os *datasets* utilizados rotulam os *bots* e o início de um ataque DDoS. A rotulagem é importante pois permite verificar se o sistema proposto consegue identificar sinais da preparação dos ataques. Para a avaliação, os *datasets* devem apresentar alguma comunicação dos *bots* ou alguma ação anterior ao ataque realizada pelo atacante, como a infecção dos dispositivos ou algum teste de ataque. A marcação do início do ataque é necessária, pois a avaliação utiliza o tráfego de rede anterior ao início do ataque. Este trabalho usou a captura 51 do *dataset* CTU-13 e o *dataset* CIC-DDoS2019, pois eles suprem os requisitos acima.

Os experimentos agrupam os pacotes dos *datasets* a cada um segundo para extraírem os atributos. Utilizou-se o intervalo de um segundo para obter resultados mais precisos. Contou-se quantos pacotes a rede trafegou em cada segundo. Escolheu-se a quantidade de pacotes, por ser um dos atributos mais relevantes identificados

em [Feng et al. 2018]. Quando os atacantes testam os ataques, a quantidade de pacotes trafegados varia. A quantidade de endereços de IP (do inglês, *Internet Protocol* - IP) na origem e no destino dos pacotes são os outros atributos empregados. Para definir a quantidade de IPs de origem, contou-se quantos IPs únicos enviaram pacotes utilizando o campo endereço de origem do pacote IP. A quantidade de IPs de destino baseia-se na contagem de IPs únicos existentes no campo de destino do pacote IP. Esses atributos foram selecionados, pois a falsificação de IPs é uma prática comum em ataques DDoS [Jyoti & Behal 2021]. A quantidade de IPs que enviam pacotes antes do ataque apresenta potencial como atributo pois a preparação do ataque causa variações nesse atributo.

A execução do sistema consiste em calcular os sinais precoces de alerta *Kurtosis*, *Skewness* e CV (Subseção 3.2), utilizando uma janela deslizante de tamanho fixo. Calculou-se um sinal para cada atributo do tráfego de rede para a seleção ser executada em tempo hábil pelo sistema. Portanto o *Kurtosis* foi calculado para o atributo quantidade de IPs de origem, o *Skewness* foi calculado para a quantidade de IPs de destino e o CV foi calculado para o total de pacotes. Todos os resultados do sistema estão disponíveis online[1]. A combinação de sinais precoces de alerta com as características foi responsável por maximizar os resultados da predição de ataques DDoS dentre vários testes. Porém trabalhos futuros criarão maneiras para definir essas combinações automaticamente. O sistema utiliza o conceito de janela deslizante de tamanho fixo para eliminar tendências errôneas e avaliar a solução proposta ao longo do tempo [Bury et al. 2020]. A literatura ainda não é unânime sobre esse valor, por exemplo, Bury *et al.* (2020) utilizaram 40% do *dataset* para o tamanho da janela. Este trabalho definiu 5% com o intuito de maximizar o tempo de predição dos ataques DDoS. Ao utilizar um valor alto a predição é atrasada, uma vez que o sistema utiliza mais tempo para realizar as análises.

Após a engenharia de caraterísticas em cada *dataset*, executou-se o Autokeras [Jin et al. 2019] para identificar a melhor rede neural do tipo LSTM *Autoencoder* para cada *dataset*. O Autokeras utiliza uma otimização Bayesiana para encontrar a melhor arquitetura de rede neural baseada nos dados inseridos pelo usuário. O Autokeras implementa os passos da Figura 2 por meio de um algoritmo de otimização de função de aquisição estruturado em árvore e um *kernel* de rede neural. O objetivo é que o Autokeras possa analisar o espaço de busca eficientemente. O processo de seleção e configuração da arquitetura da rede LSTM *Autoencoder* foi executado na versão gratuita do Google Colab.A avaliação utiliza a acurácia, precisão e o *recall*. Para definir essas métricas deve-se medir: (i) total de verdadeiros positivos VP; (ii) total de falsos positivos (FP); (iii) total de falsos negativos (FN); (iv) total de verdadeiros negativos (VN) e (v) total de observações (N). A acurácia avalia as classificações do sistema (Eq. 7). Como há poucos sinais de preparação para o ataque, pois os invasores ocultam suas ações, é necessário complementar a análise com a precisão e o *recall*. A precisão indica a relação entre as observações rotuladas pelo sistema para um tipo específico e quantas são do tipo assumido (Eq. 8). O *recall* apresenta a relação entre todas as observações esperadas do tipo específico e quantas observações desse tipo o sistema classificou corretamente (Eq. 9). Como é possível obter a precisão e o *recall* para as classes positivas e negativas, e a quantidade de amostras varia muito devido ao desbalanceamento inerente ao ataque DDoS, assume-se a precisão e o *recall* médios ponderados pela quantidade de amostras de cada classe.

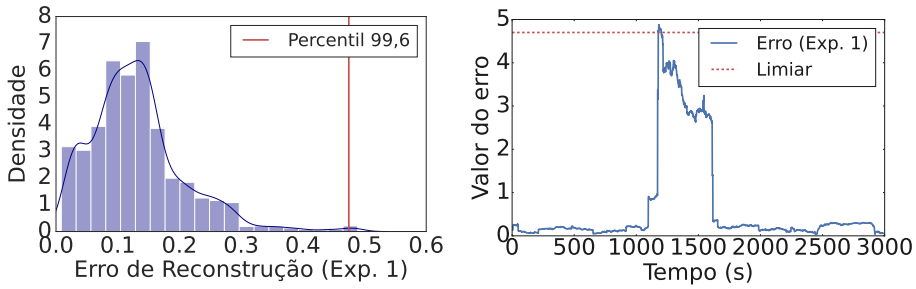
[1] `github.com/daviembrito/pred-ddos-automl`

$$Acurácia = \frac{VP + VN}{N} \quad (7) \qquad Precisão = \frac{VP}{VP + FP} \quad (8) \qquad Recall = \frac{VP}{VP + FN} \quad (9)$$

4.1. Experimentos

O *Experimento 1* utilizou o tráfego de rede coletado na rede local de uma universidade e disponibilizada no *dataset* CTU-13. Essa captura possui 8803 segundos, 10 *bots*, 41 GB, 46.997.342 pacotes e ataques do tipo inundação de Internet Control Message Protocol (ICMP) e UDP. Para criar o *dataset*, os pesquisadores capturaram dados reais da universidade, infectando os *bots* no segundo 2643 e lançando os ataques no segundo 5632 da captura. Para executar o Autokeras e configurar a rede LSTM *Autoencoder*, utilizou-se 30% do tráfego de rede total, contendo apenas tráfego prévio ao ataque (*i.e.*, do segundo 0 até o segundo 2642). O teste utilizou o tráfego restante até o momento anterior ao início do ataque (*i.e.*, do segundo 2643 até o segundo 5631). Para demonstrar o poder de customização do sistema, o Experimento 1 usa um percentil alto definido em 99,6%. Isso é, o limiar é definido com o valor do erro da menor amostra fora das 99,6% menores amostras.

A Figura 3(a) apresenta o histograma do erro na base de treinamento. Seguindo essa abordagem, o limiar definido para esse experimento é de 0,47. Porém vale ressaltar que o administrador de rede pode determinar como o sistema emitirá o alerta. Para exemplificar isso, este trabalho customizou o limiar multiplicando-o por 10. Isso resulta em um limiar mais rígido e mais customizado que o limiar padrão do sistema. Utilizando o limiar 4,7 representado pela linha vermelha na Figura 3(b), o sistema errou a classificação de apenas 63 dos 2990 segundos da base de teste. Isso garante uma acurácia de 97,89%, uma precisão média ponderada de 97,4% e um *recall* médio ponderado de 97,9% para identificar os segundos com mais de 2 pacotes enviados por *bots* (Tabela 1). O sistema proposto predisse o ataque DDoS 29 minutos e 51 segundos antes do seu início.

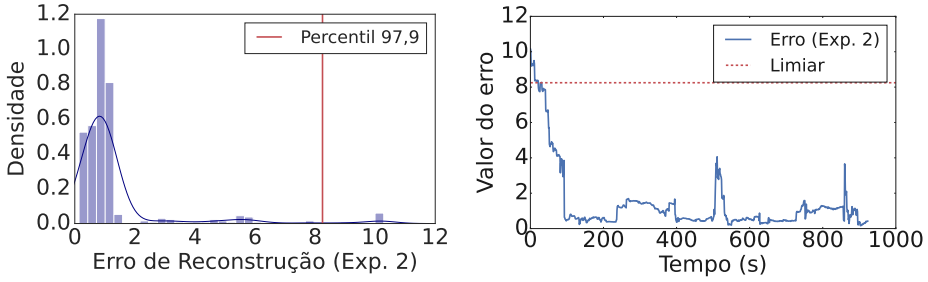


(a) Distribuição do erro no treinamento (b) Variação do erro no teste

Figura 3. Resultados Experimento 1

O *Experimento 2* aplicou o *dataset* CIC-DDoS2019, onde a rede vítima está conectada ao atacante pela Internet. Essa captura possui 19 ataques DDoS lançados pelos pesquisadores em dois dias. O *dataset* possui 61.407.883 de pacotes e 27 GB de dados referente aos ataques e dados reais. A avaliação do sistema concentrou-se na predição do primeiro ataque DDoS realizado. O ataque começa no segundo 1484 da captura e possui duração de 540 segundos. O experimento utilizou 28% do tráfego total da captura para realizar o treinamento do modelo (*i.e.*, do segundo 0 até o segundo 559) e o restante de tráfego prévio ao ataque para avaliação (*i.e.*, do segundo 560 até o segundo 1483). O Experimento 2 usa a configuração padrão do limiar definido na Subseção 3.4. Isso implica

em um limiar mais baixo e menos customizado. Neste caso, o valor é de 97,9%. Isso significa que o limiar é definido com o valor do erro da menor amostra fora do grupo 97,9% menor. A Figura 4(a) apresenta o histograma do erro na base de treinamento. O limiar foi definido como 8,3. Usando esse limiar (Figura 4(b)), a arquitetura predisse o ataque DDoS com 15 minutos e 21 segundos de antecedência, classificando corretamente 791 de 924 segundos com mais de dois pacotes enviados por *bots*. A acurácia foi de 85,39%, a precisão média ponderada de 78,7% e o *recall* médio ponderado foi de 85,4% (Tabela 1).



(a) Distribuição do erro no treinamento (b) Variação do erro no teste

Figura 4. Resultados Experimento 2

4.2. Discussão dos resultados

O sistema proposto predisse o ataque DDoS com acurácia superior a 85%. A solução notifica sobre a ocorrência de um ataque DDoS minutos antes de o atacante lançá-lo. Isso viabiliza ao administrador de rede maior tempo para proceder com as contramedidas necessárias e assim evitar prejuízos causados pelos ataques. Os resultados mostram que o sistema autonomamente seleciona e configura a melhor arquitetura de uma rede neural do tipo LSTM *Autoencoder* diferente para cada experimento realizado. Essa autonomia reduz o consumo de tempo gasto no processo de configuração da rede neural. Mesmo com o grande desbalanceamento de classes onde existe mais tráfego normal do que tráfego de ataque, o sistema conseguiu obter métricas que superam trabalhos da literatura.

O sistema proposto oferece quatro avanços para a literatura. O primeiro é o aumento no tempo de predição em relação à literatura. A proposta predisse o ataque com 29 minutos e 51 segundos de antecedência na captura 51 do CTU-13, o mesmo ataque foi predito somente com 5 minutos e 41 segundos de antecedência em [Rahal et al. 2020]. A segunda evolução foi a redução de erros de predição. A proposta de Silva *et al.* (2022) classificou erroneamente 257 segundos do tráfego de rede da captura 51 do *dataset* CTU-13, enquanto a proposta deste trabalho errou 63 segundos no mesmo contexto (linha 3, Tabela 1). Os ganhos estão relacionados com a configuração autônoma do LSTM Autoencoder realizada pelo sistema proposto. Uma configuração especializada para cada contexto permite ao sistema extrair o melhor do AM. Além disso, as características utilizadas evidenciam a preparação dos ataques durante a análise das séries temporais. Isso resulta no aumento do tempo de predição associado com a redução de erros.

Tabela 1. Resultados dos experimentos e comparação com a literatura

Experimento	VP	FP	FN	VN	Acurácia	Erro Total
Experimento 1	2926	24	39	1	97,89 %	63
Experimento 2	783	18	117	6	85,39%	133
[Silva et al. 2022]	2737	1	256	2	91,42%	257

O terceiro avanço está relacionado com a adaptação. Projetou-se um sistema para ser adaptável a diversos cenários e os resultados suportam essa generalização. O sistema proposto foi capaz de se adaptar para a predição de ataques em diferentes *datasets*. Todos eles possuem características distintas, como tipos de ataques, botnets, topologia e tamanho. Trabalhos anteriores focaram somente na captura 51 do *dataset* CTU-13 [Silva et al. 2022]. O sistema indica os erros de predição conforme os dados são processados, assim como mostra as Figuras 3 e 4 para melhor entendimento dos fatores que o levaram a realizar a predição do ataque DDoS. Por fim, todos esses avanços independem de dados rotulados. Assim, os resultados não se limitam a uma família de *botnet*, ataques DDoS específicos ou a um determinado perfil estatístico. Trabalhos futuros investigarão soluções para reduzir o consumo de tempo e recursos. O sistema utilizou aproximadamente 42 e 28 minutos para configurar e treinar as redes neurais LSTM *Autoencoder* nos Experimentos 1 e 2. A literatura indica que esse processo pode levar mais tempo, entre 3 e 24 horas [Lam & Abbas 2020]. A proposta apresenta menor tempo de processamento que a literatura pois utiliza poucas características. Trabalhos futuros abrangerão outras redes neurais (*e.g., Gated Recurrent Unit*) e seus desempenhos nos contextos apresentados.

5. Conclusão

Este artigo apresenta um sistema pioneiro na automatização da predição de ataques DDoS em redes locais e na Internet. O sistema processa o tráfego de rede por meio de uma rede neural LSTM *Autoencoder*. Usando o AutoML, o sistema configura a arquitetura da rede neural para os diferentes cenários, proporcionando a predição de ataques sem a interação humana. Além disso, o sistema não usa dados rotulados para treinar a rede neural. Os resultados da avaliação indicam que o sistema consegue predizer um ataque DDoS com 29 minutos e 51 segundos antes do seu lançamento, com acurácia de 97,89%.

Agradecimentos

Os autores agradecem o apoio da UFMG e auxílio financeiro da FAPESP processos, #2022/06802-0 e #2022/06840-0 da CAPES, processo, #88887.501287/2020-00 e #88887.509309/2020-00.

Referências

Ali, M. Q. and Al-Shaer, E. (2013). Configuration-based IDS for advanced metering infrastructure. In *SIGSAC*, page 451–462, USA. ACM.

Anuar, S., Ahmad, N. A., Sahibuddin, S., Ariffin, A., Saupi, A., Zamani, N. A., Jeffry, Y., and Efendy, F. (2018). Modeling malware prediction using artificial neural network. In *SOMET*, volume 303, pages 240–248, SPAIN. IOS Press.

Arp, D., Quiring, E., Pendlebury, F., Warnecke, A., Pierazzi, F., Wressnegger, C., Cavallaro, L., and Rieck, K. (2020). Dos and donts of machine learning in computer security.

Bedeian, A. G. and Mossholder, K. W. (2000). On the use of the coefficient of variation as a measure of diversity. *ORM*, 3(3):285–297.

Bury, T. M., Bauch, C. T., and Anand, M. (2020). Detecting and distinguishing tipping points using spectral early warning signals. *J. R. Soc.*, 17(170).

Bury, T. M., Sujith, R. I., Pavithran, I., Scheffer, M., Lenton, T. M., Anand, M., and Bauch, C. T. (2021). Deep learning for early warning signals of tipping points. *PNAS*, 118(39).

Carpenter, S. R. and Brock, W. A. (2006). Rising variance: a leading indicator of ecological transition. *Ecology Letters*, 9(3):311–318.

Dakos, V., Carpenter, S. R., Brock, W. A., Ellison, A. M., Guttal, V., Ives, A. R., Kéfi, S., Livina, V., Seekell, D. A., van Nes, E. H., and Scheffer, M. (2012). Methods for detecting early warnings of critical transitions in time series illustrated using simulated ecological data. *PLOS ONE*, 7(7):1–20.

Feng, Y., Akiyama, H., Lu, L., and Sakurai, K. (2018). Feature selection for machine learning-based early detection of distributed cyber attacks. In *DASC*, pages 173–180, Greece. IEEE.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J. T., Blum, M., and Hutter, F. (2015). Efficient and robust automated machine learning. In *NIPS*, page 2755–2763, USA. MIT Press.

Garcia, S., Grill, M., Stiborek, J., and Zunino, A. (2014). An empirical comparison of botnet detection methods. *Computers & Security*, 45:100–123.

Horsanali, E., Yigit, Y., Secinti, G., Karameseoglu, A., and Canberk, B. (2021). Network-aware AutoML framework for software-defined sensor networks. In *DCOSS*, pages 451–457. IEEE.

Ioulianou, P., Vasilakis, V., and Shahandashti, S. F. (2022). ML-based detection of blackhole and rank attacks in RPL networks.

Jaber, A. N., Zolkipli, M. F., Majid, M. A., and Anwar, S. (2017). Methods for preventing distributed denial of service attacks in cloud computing. *Advanced Science Letters*, 23(6):5282–5285.

Jin, H., Song, Q., and Hu, X. (2019). Auto-keras: An efficient neural architecture search system. In *ACM SIGKDD*, pages 1946–1956. ACM.

Jyoti, N. and Behal, S. (2021). A meta-evaluation of machine learning techniques for detection of DDoS attacks. In *INDIACom*, pages 522–526, New Delhi, India. IEEE.

Lakshmanan, R. (2023). Massive HTTP DDoS attack hits record high of 71 million requests/second. https://thehackernews.com/2023/02/massive-http-ddos-attack-hits-record.html.

Lam, J. and Abbas, R. (2020). Machine learning based anomaly detection for 5G networks. *arXiv*, -(-):12.

Machaka, P., Ajayi, O., Maluleke, H., Kahenga, F., Bagula, A., and Kyamakya, K. (2021). Modelling DDoS attacks in IoT networks using machine learning.

Nguyen, H., Tran, K., Thomassey, S., and Hamad, M. (2021). Forecasting and anomaly detection approaches using LSTM and LSTM Autoencoder techniques with the applications in supply chain management. *IJIM*, 57:1–38.

Rahal, B. M., Santos, A., and Nogueira, M. (2020). A distributed architecture for DDoS prediction and bot detection. *IEEE Access*, 8:159756–159772.

Santos, L. A. F., Campiolo, R., Gerosa, M. A., and Batista, D. M. (2013). Extração de alertas de segurança postados em mensagens de redes sociais. In *SBRC*, pages 791–804, Brasil.

Sharafaldin, I., Lashkari, A. H., Hakak, S., and Ghorbani, A. A. (2019). Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy. In *ICCST*.

Silva, G. M., Neira, A., and Nogueira, M. (2022). Aprendizado profundo para a predição de ataques de negação de serviço distribuído. In *SBRC*, pages 475–488, Brasil. SBC.

Staudemeyer, R. C. and Morris, E. R. (2019). Understanding LSTM - a tutorial into long short-term memory recurrent neural networks. *CoRR*, abs/1909.09586.