Detecção Eficiente de Anomalias em Redes de Data Centers Apoiada por Aprendizado de Máquina e Otimizador do Lobo Cinzento para Seleção de Atributos

Henrique G. Salvador[1], **Daniel M. Batista**[1], **Aldri Luiz Dos Santos**[2], **Michele Nogueira**[2]

[1]Depto. de Ciência da Computação, Universidade de São Paulo (USP)

[2]Depto de Ciência da Computação, Universidade Federal de Minas Gerais (UFMG)

{henriquesalvador,batista}@ime.usp.br, {aldri,michele}@dcc.ufmg.br

Abstract. *The high transmission rate in Data Center networks requires that such networks employ mechanisms that use few traffic features to detect anomalies, thus avoiding high consumption of resources for network monitoring. Furthermore, the detection must have high accuracy, avoiding false positives and false negatives. This paper proposes a solution with these characteristics based on the KNN algorithm for anomaly detection and the GWO optimizer for selecting the main traffic features. In experiments with the solution analyzing the UNSW-NB15 dataset, it was possible to observe a reduction of 86,21% in the number of features used to detect anomalies in the network, besides an improvement in the detection accuracy.*

Resumo. *A elevada taxa de transmissão em redes de Data Centers exige que tais redes empreguem mecanismos que utilizem poucos atributos do tráfego para detecção de anomalias, evitando assim um alto consumo de recursos para o monitoramento da rede. Além disso, é importante que a detecção possua uma alta precisão, evitando falsos positivos e falsos negativos. Este artigo propõe uma solução com essas características, apoiada no algoritmo KNN para a detecção de anomalias e no otimizador GWO para a seleção dos principais atributos do tráfego. Em experimentos com a solução analisando o dataset UNSW-NB15, foi possível observar uma redução de 86,21% na quantidade de atributos utilizados para detectar anomalias na rede, além de melhoria na acurácia da detecção.*

1. Introdução

Ao compararmos os registros de ataques de negação de serviço do primeiro trimestre de 2021 com os do primeiro trimestre de 2022 é possível observar um aumento de 4,5 vezes no número desses ataques, além de um aumento de duração na ordem de até 100 vezes [Gutnikov et al. 2022a] [Gutnikov et al. 2022b]. Ademais, os governos, as empresas e a população em geral estão cada vez mais conectados e dependentes da disponibilidade de serviços online, situação na qual surgem motivações diversas para que os ataques ocorram [NETSCOUT 2022]. Esse cenário exige soluções de segurança de redes que propiciem a identificação de ataques da forma mais precisa e eficiente possível (por eficiente, refere-se à redução na quantidade de atributos do tráfego que precisam ser analisados).

Os ataques de negação de serviço visam impedir que usuários legítimos consigam acessar determinado serviço, consumindo, de forma excessiva, recursos computacionais como processamento, memória e largura de banda, deixando o serviço lento ou

inacessível. Os ataques podem ter como motivação a intenção em gerar prejuízo financeiro ou ao intangível do alvo, confiabilidade ligada a marca/nome do alvo, e podem ser oriundos de concorrentes, de atacantes que visam receber destaque por suas habilidades ou, ainda, por ganhos financeiros [de Oliveira et al. 2021]. Os ataques de negação de serviço podem ter como fonte um único dispositivo (DoS - *Denial of Service*) ou vários dispositivos infectados (*bots*), orquestrados por um atacante (*bot master*) que utiliza de uma rede de bots (*botnet*) para lançar ataques de negação de serviços distribuídos (DDoS - *Distributed Denial of Service*) [Gasti et al. 2013].

Com o advento da computação em nuvem e da proliferação dos modelos de computação como serviço (SaaS, PaaS, IaaS, etc...), os centros de processamento de dados, ou Data Centers, passaram a desempenhar um papel importante para o funcionamento de toda a Internet. Se até os anos 1990 tais centros eram exclusividade de grandes bancos e centros de pesquisa, hoje eles são a base para serviços como plataformas de compartilhamento de vídeo e redes sociais online. A dependência dos Data Centers fez com que surgissem empresas especializadas em fornecer recursos desses ambientes para outras empresas disponibilizarem serviços na Internet. Logo, um mesmo Data Center em um dado momento pode estar fornecendo serviços de diversas empresas de forma independente. Essa centralização faz com que os Data Centers tornem-se alvos interessantes para atacantes, já que diversos serviços e empresas podem sofrer as consequências de um ataque bem planejado e direcionado para um único ponto da Internet.

Para detectar ataques direcionados a Data Centers, pode-se adotar algoritmos de Aprendizado de Máquina (*Machine Learning* - ML) com o intuito de identificar anomalias nos fluxos da rede. Os algoritmos de ML podem realizar o aprendizado supervisionado, em que se tem dados rotulados durante o treinamento, e aprendizado não supervisionado, no qual os rótulos não estão presentes durante o treinamento. Pode-se obter uma melhora de desempenho dos modelos, em termos de consumo de recursos computacionais, ao selecionar atributos por meio da utilização de otimizadores ou métricas como, por exemplo, a correlação entre atributos com o intuito de diminuir a quantidade necessária para gerar o modelo, bem como a quantidade de dados necessários para a realização da classificação dos fluxos no Data Center. Dado que o tráfego em Data Centers de grandes empresas flui na ordem de dezenas ou centenas de Gigabits por segundo, essa diminuição de atributos é extremamente importante para que tanto o monitoramento quanto o armazenamento do tráfego a ser analisado sejam realizados de forma mais eficiente.

Este artigo apresenta uma solução para a detecção de anomalias nos fluxos da rede de um Data Center, valendo-se de algoritmos de ML e do Otimizador do Lobo Cinzento para realizar a seleção de atributos. O otimizador faz uso da estratégia de caça dos Lobos Cinzentos para encontrar a melhor solução de um problema. O uso deste otimizador deve-se aos bons resultados obtidos por ele na detecção de anomalias em data centers [Garg et al. 2019], com a observação de que é possível melhorar a sua função de fitness, além de avaliar o seu desempenho com datasets mais recentes, dois avanços realizados neste artigo em relação ao trabalho de [Garg et al. 2019]. A redução da quantidade de atributos para a geração dos modelos de detecção de anomalias ocorre de modo a não impactar significativamente na acurácia dos modelos, preservando a capacidade de classificar adequadamente os fluxos da rede do Data Center. Na realização

dos experimentos adotou-se os datasets KDD'99 [Bay et al. 2000][1] e o UNSW-NB15 [Moustafa and Slay 2015] e os resultados obtidos comprovaram a eficácia da proposta. Por exemplo, com o dataset UNSW-NB15, houve uma redução de 86,21% na quantidade de atributos usados para detectar anomalias na rede, e melhoria na acurácia da detecção.

O restante deste artigo está estruturado da seguinte maneira: a Seção 2 apresenta os trabalhos relacionados. A Seção 3 descreve o otimizador utilizado para seleção de atributos. A Seção 4 descreve o funcionamento do sistema proposto. A Seção 5 detalha a metodologia de avaliação de desempenho e os resultados alcançados. Por fim, a Seção 6 apresenta as conclusões e os trabalhos futuros.

2. Trabalhos Relacionados

Nos últimos anos, uma série de trabalhos na literatura abordaram a detecção de anomalias em redes de computadores, boa parte deles apoiadas em algoritmos de ML [Özgür and Erdem 2016][Al-Daweri et al. 2020], como os trabalhos de [Tan et al. 2020], [Biswas et al. 2021] e [Haider et al. 2020]. Vale observar que quando se trata de soluções voltadas para Data Centers, tem-se um cenário crítico e desafiador [Dong et al. 2021].

Em [Silva et al. 2022], os autores propõem uma solução que utiliza o algoritmo LSTM (*Long Short-Term Memory*) sobre o dataset CTU-13 para realizar a detecção antecipada de ataques DDoS. Os autores utilizam uma janela de análise com tempo de 1 segundo, podendo variar em até 10% sobre o tempo total analisado, que considera janelas de até 568s. Essa técnica é denominada janelas deslizantes. Para identificar a mudança de status da rede, adotou-se as medidas estatística de Skewness e de Kurtosis. Essas métricas são utilizadas, originalmente, em monitoramento de mudanças climáticas e ecológicas, baseando-se em conceitos de séries temporais. A solução atingiu antecipação de 46 e de 29 minutos em seus experimentos para detecção de ataques DDoS. Esta solução procura identificar anomalias no fluxo da rede para sinalizar possíveis ataques. Os autores [Leevy et al. 2021] apresentam uma solução para identificar ataques em redes, avaliando os modelos com métricas como a AUC (*Area under the Receiver Operating Characteristic Curve*) e a AUPRC (*Area Under the Precision-Recall Curve*). Os algoritmos de ML utilizados que geraram modelos que apresentaram melhor desempenho ao detectar anomalias que podem indicar ataques foram: CatBoost, LightGBM e XGBoost.

Em [Tan et al. 2020], os autores utilizam uma estratégia para detecção de anomalias dividida em duas fases: (1) Mecanismo de gatilho (*trigger*); e (2) A aplicação de algoritmos de ML, K-means e KNN (*K-Nearest Neighbor*), para gerar modelos que realizam a classificação dos fluxos em malignos e benignos. O mecanismo de gatilhos utiliza-se da capacidade de processamentos dos switches para processar e calcular as estatísticas da rede e sinalizar fluxos suspeitos, de acordo com um limiar (*threshold*) e assimetria dos fluxos. A sinalização gerada pelos switches segue para o controlador da rede, que então aplica os modelos para classificar os fluxos suspeitos em normais ou maliciosos. Os fluxos maliciosos ou malignos são bloqueados pelo controlador. Essa estratégia reduz o processamento no controlador, tendo em vista que a triagem é realizada nos switches, seguindo para o controlador apenas os fluxos suspeitos, auxiliando na melhoria da eficiência da detecção e de defesa do controlador da rede.

[1]Mesmo sendo antigo, este dataset foi utilizado para que fosse possível comparar a nossa solução com a solução apresentada por [Garg et al. 2019] pois, naquele trabalho, este dataset foi utilizado.

Em [Garg et al. 2019], os autores apresentam um sistema híbrido para a detecção de anomalias em Nuvens, que adota Rede Neural Convolucionais (*Convolutional Neural Network* - CNN) para classificar os fluxos da rede e o algoritmo Otimizador do Lobo Cinzento (*Grey Wolf Optimizer* - GWO) para seleção de atributos. Dentre os melhores resultados, tem-se a aplicação da solução sobre a base de dados KDD'99, em que se obteve a redução de 41 para 34 atributos e com acurácia de 98,42%. Em [Biswas et al. 2021], os autores definiram uma amostragem dos fluxos que trafegam na estrutura do Data Center, considerando apenas os fluxos internos. Essa solução prevê uma situação na qual um nó interno ataca outro nó do mesmo Data Center. Neste cenário, o monitoramento considera o volume de dados que trafegam na estrutura interna para o monitoramento de ataques no Data Center e seu impacto no tráfego da rede. Os autores propõem um *framework* para monitoramento do fluxo interno da rede, determinando se a amostragem é adequada ao monitoramento. O *framework* usa um algoritmo de clustering hierárquico para agrupar os nós de acordo com uma matriz de similaridade, agrupando os nós conforme a relação entre pacotes recebidos e os enviados (neutro, atacante ou vítima). Além disso, a solução adota Rede Neural Artificial (*Artificial Neural Network* - ANN) para encontrar a relação entre a taxa de detecção de anomalias e a taxa de amostragem de fluxos utilizados na geração dos modelos e na detecção das anomalias nos fluxos internos.

Em [Sharma and Mukherjee 2012] é proposta uma solução que adota o algoritmo NBTree para detectar anomalias e adotaram o dataset KDD´99 para medir a eficácia da solução proposta. A solução atingiu 99,05% de acurácia e utilizou todos os 41 atributos disponíveis. Em [Guo et al. 2016] os autores adotaram os algoritmos KNN e a Detecção de Anomalias baseados na mudança de centróides do cluster (*Anomaly-Detection method based on the change of cluster center* - ADBCC), que possui a organização do cluster conforme a distância euclidiana das instâncias para o centróide, similar ao K-Means. Nesta abordagem, os autores realizaram a detecção de anomalias nos fluxos da rede em duas fases, uma com KNN e a segunda com ADBCC, utilizando todos os atributos disponíveis e obtendo uma acurácia de 93,29%.

A Tabela 1 apresenta uma síntese dos principais trabalhos relacionados, os algoritmos adotados em cada solução, a acurácia e se a solução adota a seleção de atributos para gerar os modelos de detecção de ataques. Em [Biswas et al. 2021], os autores procuram otimizar os recursos do Data Center ao realizar a amostragem dos fluxos a serem analisados. No entanto, a solução envia para análise um conjunto de atributos que poderia ser otimizado. A sua redução implicaria diretamente no aumento da amostragem que o *framework* seria capaz de analisar consumindo os mesmos recursos ou menos. Condição similar ao que corre em [Sharma and Mukherjee 2012] e [Guo et al. 2016]. Já em [Garg et al. 2019] pode-se melhorar a função fitness utilizada pelo otimizador, além da utilização de outros algoritmos para se obter um subconjunto de atributos que ofereça um impacto reduzido na acurácia dos modelos gerados para detecção de anomalias.

3. GWO - *Grey Wolf Optimizer*

Esta seção detalha o funcionamento do otimizador GWO (*Grey Wolf Optimizer* / Otimizador do Lobo Cinzento) e sua aplicabilidade para a seleção de atributos de um dataset, visando reduzir a quantidade de atributos necessários para a geração dos modelos de detecção de anomalias nos fluxos de rede. Essa redução da quantidade de atributos

Tabela 1. Comparação entre as soluções

Trabalho	Algoritmos	Acurácia	Seleção de atributos
[Silva et al. 2022]	Janelas deslizantes, Skewness e Kurtos.	91,00%	Não. CTU-13. Modelo gerado abordando todos os atributos.
[Leevy et al. 2021]	XGBoost	98,54%	Não. Bot-IoT. Modelo gerado abordando todos os atributos.
[Tan et al. 2020]	KNN e K-Means	98,85%	Não. KDD'99 (NSL-KDD). Modelo gerado abordando todos os atributos.
[Garg et al. 2019]	CNN + GWO	98,42%	Sim. KDD'99. Dos 41 atributos 34 foram selecionados.
[Biswas et al. 2021]	Amostragem, ANN	93,38%	Não. O trabalho reduz a quantidade de fluxos analisados, amostragem.
[Sharma and Mukherjee 2012]	NBTree	99,05%	Não. Adotou todos os 41 atributos para construir o modelo para realizar a respectiva classificação dos fluxos.
[Guo et al. 2016]	KNN e ABDCC	93,29%	Não. Adotou todos os 41 atributos para construir o modelo para realizar a respectiva classificação dos fluxos.

deve ocorrer de tal sorte que os modelos gerados não sofram impactos significativos na acurácia e na sua capacidade de classificar adequadamente os fluxos.

O otimizador GWO possui como inspiração comportamentos observados na natureza [Mirjalili et al. 2014]. Nesse caso, o GWO utiliza-se da estratégia dos lobos cinzentos para encontrar e abater presas. Além disso, o GWO faz uso da hierarquia dos lobos dentro da alcateia, na qual os lobos são classificados como Alfa, Beta, Delta e Ômega, do mais hábil para o menos hábil. O GWO possui duas fases para encontrar a melhor solução possível com base em sua heurística. Na primeira fase, os agentes (lobos) saem em busca de suas presas, possíveis regiões onde podem encontrar o melhor resultado de acordo com a função fitness, realizando uma busca exploratória por uma presa. Nessa busca, atribui-se os três melhores resultados aos lobos Alfa, Beta e Delta, respectivamente; os demais resultados são considerados lobos Ômegas.

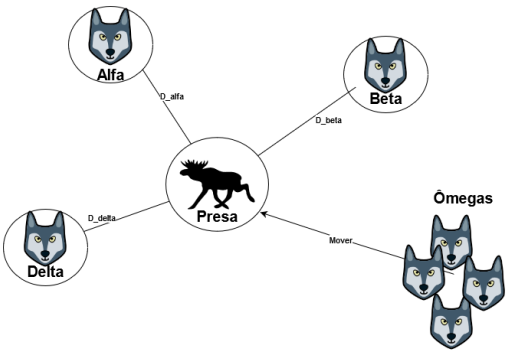

Figura 1. Movimentação dos Lobos para encontrar a presa.

Na segunda fase, tendo como base os líderes (lobos Alfa, Beta e Delta), inicia-se uma aproximação para cercar e abater a presa, ou seja, define-se a área entre as posições dos líderes como sendo a possível área onde a presa se encontra (a melhor solução possível). Nesta fase, todos os lobos são reposicionados, tendo os líderes como referência, como pode ser observado na Figura 1. Em seguida, os resultados encontrados são testados em relação aos lobos Alfa, Beta e Delta. Caso surja algum lobo com resultado melhor, tendo como métrica uma função fitness, haverá uma sucessão na hierarquia da alcateia, atualizando-se os lobos Alfa, Beta e Delta, para que eles contenham os três melhores resultados encontrados até a presente iteração.

Ao final do processo, o GWO retorna como resultado o lobo Alfa, o agente que

possui o melhor resultado encontrado pelo algoritmo. No contexto deste trabalho, o lobo Alfa contém o subconjunto com a menor quantidade possível de atributos (características / *features*) e com o menor impacto possível na capacidade de classificação de fluxos.

4. Solução Proposta

A solução adota o otimizador GWO [Mirjalili et al. 2014] para realizar a seleção de atributos, a fim de construir modelos de detecção de anomalias mais eficientes no uso de recursos computacionais como processamento, memória RAM e recursos de redes. Isso ocorre pois utiliza-se apenas os atributos necessários à geração dos modelos de detecção de anomalias, além de transmitir pela rede apenas os dados referentes aos atributos essenciais de cada fluxo para a sua classificação. Para a construção do modelo de detecção de anomalias, adotou-se o algoritmo K-Vizinhos Mais Próximos (*K-Nearest Neighbor* - KNN), uma técnica de Aprendizado de Máquina Supervisionado. O KNN define, de acordo com as instâncias mais próximas, a qual classe determinado fluxo pertence.

No pseudocódigo listado na Listagem 1, tem-se como entrada o número máximo de iterações, o número de agentes (lobos), o conjunto de dados (*dataset*) referente às conexões com o Data Center e respectivo número de atributos. Ao iniciar o método GWO_F1, da linha 2 à linha 7 realiza-se a criação da população inicial dos lobos, atribuindo a cada lobo um conjunto de atributos selecionados aleatoriamente a partir de uma distribuição uniforme. Em seguida, um conjunto de dados auxiliar é criado a partir do dataset de entrada, contendo apenas os atributos selecionados. A partir desse conjunto de dados auxiliar, divide-se os dados em dois subconjuntos, um para treinar (*train*) o modelo do KNN e outro (*test*) para avaliar a qualidade do modelo gerado. Para avaliação dos resultados encontrados por cada lobo, utilizou-se como métrica o F1-Score por ser mais apropriada para avaliação de modelos treinados a partir de dados desbalanceados. Por fim, na linha 7, a função fitness é calculada e o GWO tem como objetivo encontrar o menor valor possível para essa função.

Após iniciar a população de lobos, os três melhores lobos, isto é, os que possuem os valores menores para a função fitness, são considerados os lobos alfa (Xa), beta (Xb) e delta (Xc), respectivamente. Uma vez criada a população inicial e selecionados os três melhores lobos, inicia-se um laço com o intuito de melhorar os resultados obtidos com a população inicial. O laço realiza o número de iterações informado como um dos parâmetros de entrada do método. Dentro do laço, atualiza-se a posição (conjunto de atributos) de cada um dos lobos, tendo como referência a posição dos labos alfa, beta e delta. Em seguida, atualiza-se a variável a, reduzindo o seu valor a cada iteração e de maneira uniforme. A variável a inicia-se com o valor 2 e vai até 0, reduzindo, assim, o grau de liberdade na variação das posições (conjunto de atributos) dos lobos (agentes). Na linha 15, atualiza-se os vetores de deslocamento dos lobos e, então, cria-se novos modelos e realiza-se o cálculo dos respectivos resultados da função fitness. Na linha 17, caso outros lobos obtenham melhores resultados para a função fitness, atualiza-se os lobos $alfa$, $beta$ e $delta$, mantendo-os com os três melhores resultados obtidos até então.

Ao final da execução do método, retorna-se o lobo alfa, ou seja, o agente que obteve a melhor solução encontrada pelo algoritmo GWO, de acordo com a função fitness. O lobo alfa conterá o modelo gerado conforme os atributos selecionados, a pontuação obtida pelo F1-Score e o resultado da Função Fitness. A solução proposta possui uma

complexidade de tempo de $O(i \cdot a \cdot n \cdot d \cdot k)$, onde i representa o número máximo de iterações e a o número de agentes do GWO; n o número de registros e d a quantidade de atributos do conjunto de dados; e k o número de vizinhos definido no KNN. Uma vez encontrado, pela solução proposta, o modelo com melhor resultado para função fitness, esse modelo conterá o menor número de atributos possível e que possua um impacto reduzido em relação à qualidade do modelo.

```
GWO_F1(Max_iteration, n_agents, dataset, n_features, a=2.0)
    for i=0 to n_agents do
        X[i].select_features[] = rand.uniform[n_features]
        dataset_aux = dataset(X[i].select_features)
        X[i].KNN = KNN(dataset_aux.train)
        X[i].F1 = F1_Score(X[i].KNN, dataset_aux.test)
        X[i].fit = -0.9*X[i].F1 - 0.1/len(X[i].select_features)
    Xa = best search agent
    Xb = second best search agent
    Xc = third best search agent
    while ( t < max_iteration) do
        for i=0 to n_agents do
            update X[i] //update agents
        a = a - (2.0*t / max_iteration)
        Update A, C
        Calculate the fitness of all search agents
        Update Xa, Xb, Xc
        t = t+1
    return Xa
```

Algoritmo 1. Implementação do GWO com KNN e F1-Score.

A solução proposta pode ser aplicada tanto para detecção de anomalias que possuem como origem fontes externas ao Data Center, quanto anomalias com origem em fontes internas. Para exemplificar a aplicação da solução proposta, as anomalias têm como fontes os nós internos ao Data Center (Figura 2), considerando o cenário de um Data Center no qual ocorre a interoperação entre os nós de A até I, e o monitoramento da rede será executado em nós que estejam livres para processamento. Conforme observado na Figura 2, tem-se um cenário onde os nós D e G, na cor cinza, estão livres e serão alocados para monitoramento da rede. Os nós E e H, em vermelho, representam os nós que realizam ataques ao nó I, sinalizados por setas vermelhas e linhas tracejadas. As setas azuis com linhas sólidas indicam um fluxo normal.

Para que o nó D analise o fluxo de dados entre A e B,por exemplo, esses dados devem seguir de $S2- > S1- > S3- > D$, o que consume parte significante da banda para realização da análise dos fluxos. Desta forma, torna-se importante reduzir o impacto na capacidade de transmissão de dados no Data Center ao realizar o monitoramento da rede. Uma forma de reduzir esse impacto é transmitir um conjunto mínimo e necessário de dados para realizar a classificação dos fluxos da rede de forma adequada.

Com a adoção da solução proposta, pretende-se reduzir de forma significativa a quantidade de dados a serem transmitidos até o nó responsável pelo monitoramento da rede, uma vez que se pretende selecionar um conjunto mínimo de atributos para gerar os modelos de detecção de anomalias e que possua impacto reduzido na qualidade dos modelos de detecção de anomalias na rede do Data Center. Assim, apenas os dados referentes aos atributos selecionados seriam transmitidos até o nó de monitoramento para

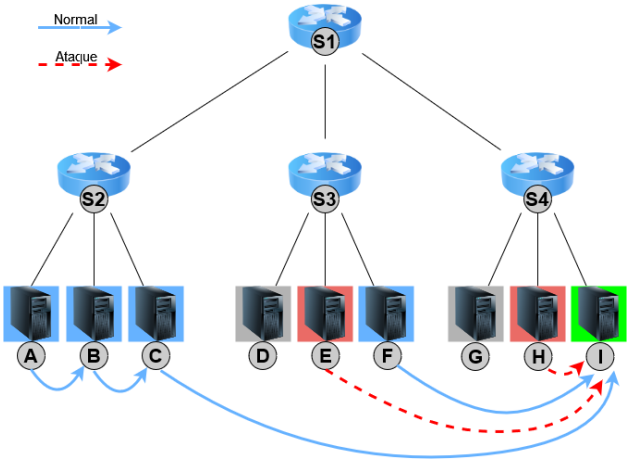

Figura 2. Arquitetura de implantação da solução em um Data Center.

que o modelo possa determinar se o fluxo analisado representa um fluxo normal ou uma anomalia, o que poderia indicar um ataque ao Data Center ou a um de seus nós.

5. Avaliação de Desempenho

Esta seção apresenta a metodologia adotada para a realização dos experimentos e os resultados alcançados. Para a realização dos experimentos, foram utilizados os dados do dataset KDD'99 [Bay et al. 2000][Özgür and Erdem 2016] e do dataset UNSW-NB15 [Moustafa and Slay 2015]. A solução foi implementada em Python, adotando especificamente as bibliotecas scikit-learn e pandas. Para a execução dos experimentos foi utilizado o Google Colab. A solução pode ser acessada em: shorturl.at/cluJX.

O dataset KDD'99 trata-se de um conjunto de dados extraído do DARPA'98 [Lippmann et al. 2000] e contém 4,9 milhões de registros já rotulados e 41 atributos. Desse conjunto de dados, foi extraída uma amostragem aleatória contendo 49.402 registros para a realização dos experimentos. Na Tabela 2 observa-se a relação de atributos disponíveis no dataset KDD'99. Para a realização dos experimentos optou-se pela divisão (*Split*) com 75% dos dados para treinamento e 25% para testes, conforme também adotado por [Garg et al. 2019], por ser mais rápido para execução de treinamento do modelo e com diferença menor que 1% em relação a acurácia obtida com a adoção do Cross-Validation (CV). O CV com 5 *folds* para o dataset obteve um desvio padrão de 0,19% entre os resultados dos *folds* e um tempo total 4,5 vezes maior para execução em relação ao teste por Split.

Embora seja utilizado para avaliar diversos mecanismos de classificação de tráfego, principalmente voltados para detecção de ataques, o dataset KDD'99 é antigo e por conta disso também utilizamos o dataset UNSW-NB15, um dataset mais recente, na análise de desempenho da nossa proposta. O dataset UNSW-NB15 trata-se de um conjunto de dados gerado a partir de um gerador de tráfego denominado IXIA. O tráfego é composto de conexões TCP em três servidores, sendo que dois deles estão conectados a um roteador com três clientes. Um terceiro servidor se conecta a um segundo roteador, que possui também três clientes. Os dois roteadores estão conectados entre si, mas com um firewall entre eles. O dataset é composto por fluxo normais (87,35%) e fluxos anor-

Tabela 2. Lista de atributos que compõe o dataset KDD'99.

#	Atributo	#	Atributo	#	Atributo
0	duration	14	su_attempted	28	same_srv_rate
1	protocol_type	15	num_root	29	diff_srv_rate
2	service	16	num_file_creations	30	srv_diff_host_rate
3	flag	17	num_shells	31	dst_host_count
4	src_bytes	18	num_access_files	32	dst_host_srv_count
5	dst_bytes	19	num_outbound_cmds	33	dst_host_same_srv_rate
6	land	20	is_host_login	34	dst_host_diff_srv_rate
7	wrong_fragment	21	is_guest_login	35	dst_host_same_src_port_rate
8	urgent	22	count	36	dst_host_srv_diff_host_rate
9	hot	23	srv_count	37	dst_host_serror_rate
10	num_failed_logins	24	serror_rate	38	dst_host_srv_serror_rate
11	logged_in	25	srv_serror_rate	39	dst_host_rerror_rate
12	num_compromised	26	rerror_rate	40	dst_host_srv_rerror_rate
13	root_shell	27	srv_rerror_rate	41	label

mais (12,65%), ou ataques. Os fluxos contidos nesse dataset já se encontram rotulados. O dataset possui um total de 2,5mi de registros e 45 atributos, do quais 175 mil registros foram selecionados para a realização dos experimentos, além de realizar a transformação dos atributos, criando colunas binárias. Após a transformação, o dataset ficou com 58 atributos. Ainda na etapa de pré-processamento, realiza-se a exclusão de registros com dados nulos e dos atributos 'id' e 'label'. A amostragem final ficou com 81.173 registros, destinando-se 75% para o conjunto de treinamento e 25% para o de testes. Na Tabela 3, observa-se a relação de atributos disponíveis no dataset UNSW-NB15.

Tabela 3. Lista de atributos que compõe o dataset UNSW-NB15.

#	Atributo	#	Atributo	#	Atributo
0	id	15	dloss	30	response_body_len
1	dur	16	sinpkt	31	ct_srv_src
2	proto	17	dinpkt	32	ct_state_ttl
3	service	18	sjit	33	ct_dst_ltm
4	state	19	djit	34	ct_src_dport_ltm
5	spkts	20	swin	35	ct_dst_sport_ltm
6	dpkts	21	stcpb	36	ct_dst_src_ltm
7	sbytes	22	dtcpb	37	is_ftp_login
8	dbytes	23	dwin	38	ct_ftp_cmd
9	rate	24	tcprtt	39	ct_flw_http_mthd
10	sttl	25	synack	40	ct_src_ltm
11	dttl	26	ackdat	41	ct_srv_dst
12	sload	27	smean	42	is_sm_ips_ports
13	dload	28	dmean	43	attack_cat
14	sloss	29	trans_depth	44	label

Uma vez selecionadas as bases de dados para realizar os experimentos, tem-se a necessidade de adequações para que seja possível aplicar os algoritmos de ML. As principais adequações foram: Transformação de Atributos, Seleção de uma amostragem dos fluxos contidos no dataset, além de estudos estatísticos.

Para adequar o dataset KDD'99 aos requisitos de entrada do algoritmo KNN, os atributos "protocol_type", "service", "flag"e "label"foram transformadas de cadeia de caracteres para categorias, utilizando as bibliotecas do Pandas [McKinney et al. 2010]. No pré-processamento, e para a solução proposta, é importante reduzir a quantidade de atributos, visando extrair as redundâncias (atributos com alta correlação) para melhorar a eficiência do modelo gerado. Isso impacta diretamente no volume de dados que irá trafegar pela rede do Data Center para o processo de monitoramento e classificação dos fluxos da rede. No entanto, para este trabalho, optou-se em manter todos os atributos para

que o GWO realizasse a seleção, gerando modelos que utilizassem o menor conjunto de atributos possível sem impactar significativamente a acurácia.

Para avaliação da eficácia dos modelos de detecção de ataques ao Data Center gerados pela solução proposta neste artigo, utilizou-se a Acurácia, dada pela soma dos acertos obtidos pelo modelo (verdadeiro positivo - TP e verdadeiro negativo - TN) dividido pelo total de instâncias classificadas, dado pelo somatório de verdadeiro positivo - TP, verdadeiro negativo - TN, falso positivo - FP e falso negativo - FN, conforme observado na Equação 4. Na função fitness adotada pelo GWO para otimizar a seleção de atributos, utilizou-se o F1-Score, dado pela Equação 6, composta pela Precisão (Equação 3) e Recall (Equação 1).

$$Recall = \frac{TP}{TP + FN} \tag{1}$$

$$FPR = \frac{FP}{FP + TN} \tag{2}$$

$$Precisao = \frac{TP}{TP + FP} \tag{3}$$

$$Acuracia = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

$$F1 - Score = 2 * \frac{Precisao * Recall}{Precisao + Recall} \tag{5}$$

Para medir os resultados obtidos pela solução proposta, realizou-se um experimento criando modelos apenas com o algoritmo KNN, sem o otimizador GWO, e outros dois que utilizam o otimizador e que adotam métricas diferentes para compor a função fitness. Desses, um experimento adota o F1-score e o outro a Acurácia para construir a função fitness, no qual se tem por objetivo melhorar a Acurácia.

Em ambos os experimentos com o otimizador GWO, considerou-se o número de atributos empregados para construir os modelos, privilegiando soluções com a menor quantidade de atributos. Pode-se observar as Equações 6 e 7 as funções fitness adotadas.

$$0,9 * f1_score + 0,1 * \left(\frac{1}{len(selected_features)}\right) \tag{6}$$

A Figura 3 apresenta um comparativo dos resultados obtidos ao aplicar a solução proposta sobre o dataset KDD'99. Nessa figura pode-se observar os resultados ao gerar o modelo apenas com o algoritmo KNN; o resultado obtido pela solução proposta que combina o KNN com o GWO e utiliza o F1-Score como métrica na função fitness ($KNN + GWO_F1$), conforme a Equação 6; e, por último, o resultado da solução que utiliza o KNN combinado com GWO e a Acurácia para compor a função fitness ($KNN + GWO_Accuracy$), conforme a Equação 7.

$$0,9 * acuracia + 0,1 * \left(\frac{1}{len(selected_features)}\right) \tag{7}$$

Na avaliação da taxa de erro de cada modelo, adotou-se como métrica a acurácia do modelo. A taxa de erro é o complementar da acurácia $(1 - acuracia)$. Assim, observa-se que ao usar o KNN sem o otimizador para seleção de atributos, o modelo atingiu a taxa de erro de 0,50% e fez uso dos 41 atributos para gerar o modelo; por consequência precisará dos dados dos 41 atributos da conexão para poder classificá-la de forma adequada.

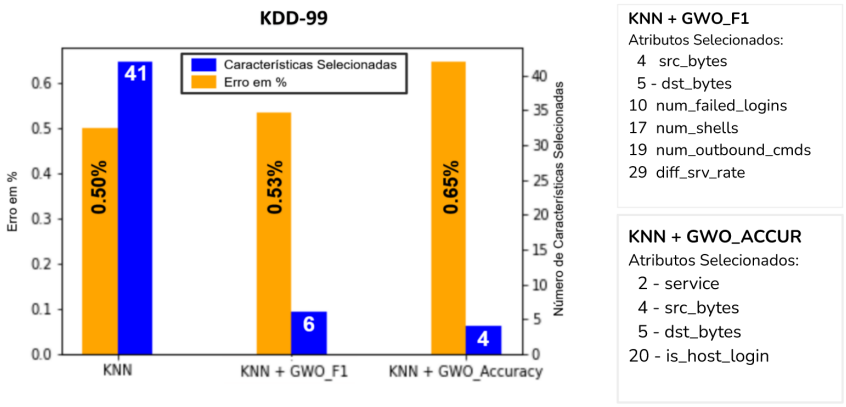

Figura 3. KDD'99 - Comparação KNN e KNN+GWO para selecionar os atributos.

A nossa proposta obteve um erro de 0,53% mas reduziu em 85,37% a quantidade de atributos necessários para gerar o modelo de detecção de anomalias na rede selecionados pelo GWO ao adotar o F1_score para compor a função fitness. Por fim, observa-se o resultado da solução que adota a acurácia para compor a função fitness, que obteve um erro de 0,65% e selecionou 4 atributos para gerar o modelo, redução de 90,24% na quantidade de atributos.

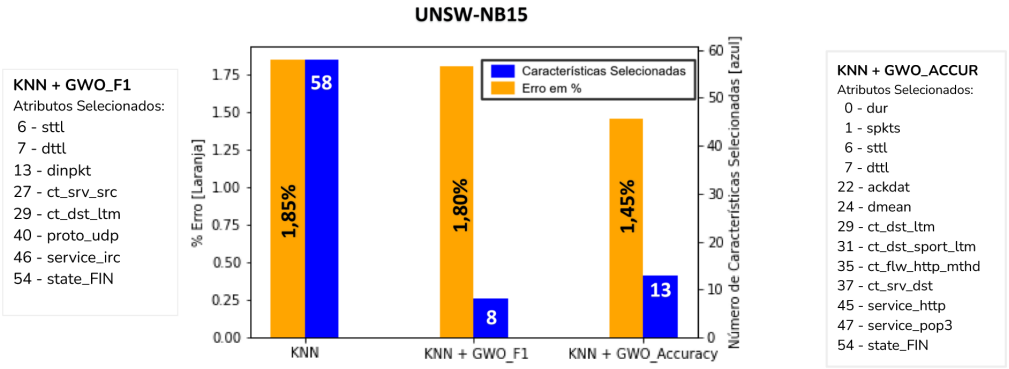

Figura 4. UNSW-NB15 - Comparação KNN e KNN+GWO para selecionar os atributos.

Na Figura 4 observa-se os resultados obtidos ao aplicar a solução proposta sobre o dataset UNSW-NB15. Ao aplicar o algoritmo KNN sem o otimizador, o modelo utiliza todos os 58 atributos, com um erro de 1,85%. Já quando se aplica o KNN com o GWO e a acurácia como métrica da função fitness, há a redução de 77,58% na quantidade de atributos necessários (total de 13 atributos) e erro de 1,45%. Por fim, ao se utilizar o KNN com o otimizador GWO e o F1-Score para compor a função fitness, foi gerado um modelo de

Tabela 4. Comparação entre a solução proposta e a [Garg et al. 2019].

	Solução Proposta	**[Garg et al. 2019]**
Dataset	KDD'99 e UNSW-NB15	KDD'99, DARPA'98 e dados sintéticos
Função Fitness	$$Y * F1 + \frac{1-Y}{len(SF)} \quad (8)$$ **Onde**: Y é uma constante entre 0 e 1, Para implementação foi adotado 0.9; F1 é a pontuação do F1-Score; SF representam os atributos selecionados para criação do modelo.	$$Y * \frac{F'}{F} + (1-Y) * \frac{E'}{E} \quad (9)$$ **Onde**: Y é uma constate entre 0 e 1 F' = nº de Atributos Selecionados; F = nº total de Atributos do Dataset; E' = erro do modelo criado com os Atributos Selecionados; E = erro do modelo criado com todos os Atributos. **Obs**: o valor adotado para Y não foi descrito na referência
A variável de convergência a	Implementado conforme o trabalho original que propõe o GWO [Mirjalili et al. 2014], assumindo valores de 2.0 a 0.0.	Segue a seguinte função: $$P_m = 0.5e^{-10*\frac{t}{T}} + 0.01 \quad (10)$$ **Onde**: t representa a iteração atual e T o número total de iterações.
Considerações	(1) Adequações, pré-processamento, necessárias para utilizar os datasets KDD99 e UNSW-NB15. (2) Ajustes da função fitness, adotando como parâmetro o F1-score.	(1) Código-fonte não disponível; (2) Alguns pontos do pseudocódigo não foram descritos de forma adequada, imprecisa ou apresentam erro, o que dificultou a reprodução dos resultados.

detecção de anomalias com a redução da quantidade de atributos necessários em 86,21%, usando o total de 8 atributos, e com um erro de 1,80% ao classificar os fluxos na rede.

Assim, conclui-se que, ao adotar o F1-Score para compor a função fitness, o GWO conseguiu selecionar um conjunto de atributos mais adequado para a criação de um modelo que possui precisão próxima ao modelo sem o otimizador (diferença de 6,45%) e com redução de 85,37% no número de atributos necessários, no experimento que utilizou o dataset KDD'99. Já ao adotar a função fitnesss proposta por [Garg et al. 2019], apesar de selecionar um número menor de atributos (4), o erro foi 29,03% maior em relação ao modelo sem o otimizador.

Ao empregar o dataset UNSW-NB15, a métrica aplicada por [Garg et al. 2019] apresenta ligeiramente melhora na acurácia, porém selecionou o total de 13 atributos, redução de 77,58%. Já a solução proposta neste artigo reduziu em 86,21% o número de atributos necessários para classificar os fluxos na rede, 8 atributos no total. Além de obter uma taxa de erro ligeiramente melhor que o obtido com o KNN sem otimizador. Por outro lado, ao utilizar apenas 13,8% dos atributos do dataset para detectar as anomalias na rede, a solução contribui para a diminuição significativa da quantidade de dados que precisarão trafegar na rede do Data Center para realizar o monitoramento da rede.

Na Tabela 4 estão as principais diferenças entre a solução proposta neste artigo e a solução proposta por [Garg et al. 2019], sendo que este trabalho apresentou uma solução robusta capaz de se ajustar a datasets mais recentes como o UNSW-NB15. Além disso, a função fitness ajustada adota o F1-Score no lugar do Erro (*1 - Acurácia*) para encontrar a melhor solução com o GWO. A preservação da variável a conforme a solução original contribuiu para um melhor desempenho dos resultados obtidos.

6. Conclusão e Trabalhos Futuros

A solução apresentada adotou o Otimizador do Lobo Cinzento (GWO) para a seleção de atributos e do algoritmo KNN, com aprendizado supervisionado, para criar o modelo de detecção de anomalias no fluxo de rede de um Data Center. No GWO, adotou-se como

métricas da função fitness o F1-Score do modelo gerado e a quantidade de atributos necessários para a construção deste modelo. Após a realização dos experimentos com os datasets KDD´99 e UNSW-NB15, a solução proposta obteve um erro de 0,53% ao classificar os fluxos da rede do Data Center e se utilizou de apenas 6 atributos dentre os 41 disponíveis no dataset KDD'99, redução de 85,37%. Além disso, ao aplicar a solução sobre o dataset UNSW-NB15, foram necessários apenas 8 atributos dos 58 disponíveis para realizar a detecção de anomalias, redução de 86,21% na quantidade de atributos necessários para a geração do modelo e a detecção adequada de anomalia nos fluxos de rede no Data Center. Além de apresentar a melhoria na taxa de erro de 1,85% para 1,80%. A redução da quantidade de atributos para geração dos modelos e, por conseguinte, para a classificação dos fluxos na rede, tem impacto direto no uso da banda na rede do Data Center, reduzindo significativamente a transferência de dados ao adequado monitoramento da rede. Como trabalhos futuros, pretende-se melhorar a geração da população de lobos iniciais, reduzindo os atributos redundantes já na geração da população inicial, além de realizar ajustes e melhorias na função fitness. Em relação à detecção de anomalias nos fluxos de rede em Data Centers, pretende-se realizar experimentos com algoritmos de Deep Learning e aplicar nossa solução em outros datasets para verificar a robustez da solução.

Agradecimentos

Esta pesquisa é parte do projeto de pesquisa temático MENTORED: da modelagem à experimentação - predizendo e detectando ataques DDoS e zero-day, um projeto selecionado na chamada pública MCTIC/CGI/FAPESP 2018 (proc. 18/23098-0). Ela também é parte do INCT da Internet do Futuro para Cidades Inteligentes, financiado por CNPq (proc. 465446/2014-0), Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001 e FAPESP (procs. 14/50937-1 e 15/24485-9).

Referências

Al-Daweri, M. S., Zainol Ariffin, K. A., Abdullah, S., and Md. Senan, M. F. E. (2020). An analysis of the KDD99 and UNSW-NB15 datasets for the intrusion detection system. *Symmetry*, 12(10):1666.

Bay, S. D., Kibler, D., Pazzani, M. J., and Smyth, P. (2000). The UCI KDD archive of large data sets for data mining research and experimentation. *ACM SIGKDD explorations newsletter*, 2(2):81–85.

Biswas, R., Kim, S., and Wu, J. (2021). Sampling rate distribution for flow monitoring and DDoS detection in datacenter. *IEEE Transactions on Information Forensics and Security*, 16:2524–2534.

de Oliveira, G. W., Porto, J. R., Prates Jr, N. G., dos Santos, A. L., Nogueira, M., and Batista, D. M. (2021). *Virtualização de Funções de Rede na IoT: Um Panorama do Gerenciamento de Desempenho x Segurança*, chapter 3, pages 101–145. Sociedade Brasileira de Computação. Minicursos do XXXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, 1 edition.

Dong, H., Munir, A., Tout, H., and Ganjali, Y. (2021). Next-Generation Data Center Network Enabled by Machine Learning: Review, Challenges, and Opportunities. *IEEE Access*.

Garg, S., Kaur, K., Kumar, N., Kaddoum, G., Zomaya, A. Y., and Ranjan, R. (2019). A hybrid deep learning-based model for anomaly detection in cloud datacenter networks. *IEEE Transactions on Network and Service Management*, 16(3):924–935.

Gasti, P., Tsudik, G., Uzun, E., and Zhang, L. (2013). DoS and DDoS in Named Data Networking. In *2013 22nd International Conference on Computer Communication and Networks (ICCCN)*, pages 1–7.

Guo, C., Ping, Y., Liu, N., and Luo, S.-S. (2016). A two-level hybrid approach for intrusion detection. *Neurocomputing*, 214:391–400.

Gutnikov, A., Kupreev, O., and Shmelev, Y. (2022a). DDoS attacks in Q1 2022. url: https://securelist.com/ddos-attacks-in-q1-2022/106358/.

Gutnikov, A., Kupreev, O., and Shmelev, Y. (2022b). DDoS attacks in Q3 2022. url: https://securelist.com/ddos-attacks-in-q1-2022/106358/.

Haider, S., Akhunzada, A., Mustafa, I., Patel, T. B., Fernandez, A., Choo, K.-K. R., and Iqbal, J. (2020). A deep CNN ensemble framework for efficient DDoS attack detection in software defined networks. *Ieee Access*, 8:53972–53983.

Leevy, J. L., Hancock, J., Khoshgoftaar, T. M., and Peterson, J. (2021). Detecting information theft attacks in the bot-iot dataset. In *2021 20th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 807–812. IEEE.

Lippmann, R., Haines, J. W., Fried, D. J., Korba, J., and Das, K. (2000). Analysis and results of the 1999 DARPA off-line intrusion detection evaluation. In *International Workshop on Recent Advances in Intrusion Detection*, pages 162–182. Springer.

McKinney, W. et al. (2010). Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, volume 445, pages 51–56. Austin, TX.

Mirjalili, S., Mirjalili, S. M., and Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software*, 69:46–61.

Moustafa, N. and Slay, J. (2015). UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *2015 military communications and information systems conference (MilCIS)*, pages 1–6. IEEE.

NETSCOUT (2022). DDoS Attacks Tied to Worldwide Geopolitical Issues and Events. url: https://www.netscout.com/threatreport.

Özgür, A. and Erdem, H. (2016). A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015. *PeerJ Preprints*.

Sharma, N. and Mukherjee, S. (2012). A novel multi-classifier layered approach to improve minority attack detection in IDS. *Procedia Technology*, 6:913–921.

Silva, G. L. F. M., Neira, A. B. d., Nogueira, M., et al. (2022). Aprendizado Profundo para a Predição de Ataques de Negação de Serviço Distribuído. In *Anais do XL Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 475–488. SBC.

Tan, L., Pan, Y., Wu, J., Zhou, J., Jiang, H., and Deng, Y. (2020). A new framework for DDoS attack detection and defense in SDN environment. *IEEE Access*, 8:161908–161919.