CENTRO UNIVERSITÁRIO FEI

MIGUEL FREZATTO

ANÁLISE DE CARACTERÍSTICAS DE NAVEGAÇÃO EM REDES PARA A DETECÇÃO DE INTRUSÃO COM BASE EM ALGORITMOS BIO-INSPIRADOS

São Bernardo do Campo

2023

MIGUEL FREZATTO

ANÁLISE DE CARACTERÍSTICAS DE NAVEGAÇÃO EM REDES PARA A DETECÇÃO DE INTRUSÃO COM BASE EM ALGORITMOS BIO-INSPIRADOS

Dissertação de Mestrado apresentada ao Centro Universitário FEI para obtenção do título de Mestre em Engenharia Elétrica. Orientado pelo Prof. Dr. Paulo Sérgio Silva Rodrigues.

São Bernardo do Campo

2023

Frezzato, Miguel.
Análise de características de navegação em redes para a detecção de intrusão com base em algoritmos bio-inspirados / Miguel Frezzato. São Bernardo do Campo, 2023.
70 p. : il.

Dissertação - Centro Universitário FEI.
Orientador: Prof. Dr. Paulo Sérgio Silva Rodrigues.

1. Sistema de detecção de intrusão. 2. Algoritmos bio-inspirados. 3. Segurança cibernética. I. Rodrigues, Paulo Sérgio Silva, orient. II. Título.

Elaborada pelo sistema de geração automática de ficha catalográfica da FEI com os dados fornecidos pelo(a) autor(a).

centro universitário FEI

APRESENTAÇÃO DE DISSERTAÇÃO
ATA DA BANCA EXAMINADORA

Programa de Pós-Graduação Stricto Sensu em Engenharia Elétrica

Mestrado

PGE-10

Aluno(a): Miguel Frezzato **Matrícula:** 120304-1

Título do Trabalho: ANÁLISE DE CARACTERÍSTICAS DE NAVEGAÇÃO EM REDES PARA A DETECÇÃO DE INTRUSÃO COM BASE EM ALGORITMOS BIO-INSPIRADOS

Área de Concentração: Processamento de Sinais e Imagens

Orientador(a): Prof. Dr. Paulo Sérgio Silva Rodrigues

Data da realização da defesa: 22/09/2023 **ORIGINAL ASSINADA**

Avaliação da Banca Examinadora:

O aluno respondeu adequadamente todas as perguntas da banca e foi aprovado por unanimidade.

A Banca Julgadora acima-assinada atribuiu ao aluno o seguinte resultado:

APROVADO ☒ REPROVADO ☐

MEMBROS DA BANCA EXAMINADORA

Prof. Dr. Paulo Sérgio Silva Rodrigues

Prof. Dr. Rodrigo Sanches Miani

Prof. Dr. Ricardo de Carvalho Destro

Aprovação do Coordenador do Programa de Pós-graduação

Prof. Dr. Carlos Eduardo Thomaz

AGRADECIMENTOS

A Deus, pela minha vida, e por me permitir ultrapassar todos os obstáculos encontrados ao longo da realização deste trabalho.

Ao meu orientador, Prof. Dr. Paulo Sergio Rodrigues, por ter acreditado no meu potencial e ter me incentivado do início ao fim do curso.

À minha noiva, Daniella Carioni, que esteve ao meu lado em todos os momentos me dando forças e foi a primeira leitora deste trabalho.

A todos os familiares, por sempre terem me apoiado e entenderem a minha ausência enquanto eu me dedicava ao curso de Mestrado.

Ao Centro Universitário FEI, pela infraestrutura, disponibilidade exímios profissionais e de recursos para o desenvolvimento do trabalho.

RESUMO

Com o constante aumento de usuários conectados à Internet, um grande volume de dados tem sido gerado a partir de várias redes. Em vista disso, a segurança cibernética está sendo cada vez mais afetada, havendo grande necessidade de estudos científicos na área. Sistemas de detecção de intrusão (IDS) cada vez mais robustos são constantemente desenvolvidos, visando proteger os dados que são trafegados nas redes. Estes sistemas analisam as características de fluxo de cada dispositivo na rede para identificar uma possível instrusão. Selecionar apenas as características que mais se relacionam com as intrusões influencia diretamente na velocidade da análise, além de auxiliar os classificadores a tomar decisões precisas ao identificar uma intrusão. Por outro lado, o desenvolvimento do aprendizado de máquina e de algoritmos de otimização inspirados na natureza têm impulsionado o avanço de diversas áreas tecnológicas. Assim, o presente trabalho apresenta uma metodologia de análise dessas características utilizando uma combinação de aprendizado de máquina e algoritmos bio-inspirados para detecção eficiente de intrusões na rede. Os resultados experimentais mostram que o método proposto aumenta a acurácia e a taxa de detecção do IDS, além de diminuir a taxa de falsos alarmes. Além disso, o método se mostrou competitivo com os principais trabalhos relacionados do estado da arte com desempenho semelhante ou superiror nas bases de dados NSL-KDD e UNSW-NB15.

Palavras-chave: Algoritmos bio-inspirados. Sistema de detecção de intrusão. Segurança cibernética.

ABSTRACT

With the constant increase of users connected to the internet, a large volume of data is generated from various networks. Because of this, cybersecurity is being increasingly affected, with a great need for scientific studies in the area. Intrusion detection systems (IDS) are increasingly robust and constantly being developed to protect data that is transmitted over networks. These systems analyze the flow features of each user on the network to identify a possible intrusion. Selecting only the features that are most related to the intrusions influences the speed of the analysis, in addition to helping the classifiers to make accurate decisions when identifying an intrusion. On the other hand, the development of machine learning and optimization algorithms inspired by nature has driven the advancement of several technological areas. Thus, this work presents a methodology for analyzing these characteristics using a combination of machine learning and bio-inspired algorithms for efficient detection of network intrusions. The experimental results show that the proposed method increases the accuracy and detection rate of the IDS and decreases the false alarm rate. In addition, the method proved to be competitive with the main state-of-the-art related works with similar or superior performance in the NSL-KDD and UNSW-NB15 datasets.

Keywords: Bio-inspired Algorithms. Intrusion deteccion systems. Cyber security.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

LISTA DE ABREVIATURAS

ALO	*Antlion Optimizer*.
CS	*Cuckoo Search*.
DoS	*Denial os Service*.
FNR	*False Negative Rate*.
FPR	*False Positive Rate*.
GWO	*Grey Wolf Optimizer*.
HIDS	*Host Intrusion Detection System*.
IDS	*Intrusion Detection System*.
KH	*Krill Herd*.
KNN	*K-Nearest Neighbors*.
NIDS	*Network Intrusion Detection System*.
TNR	*True Negative Rate*.
UNB	Universidade de New Brunswick.
UNSW Sydney	Universidade de Nova Gales do Sul.
WOA	*Whale Optimization Algorithm*.

SUMÁRIO

1 INTRODUÇÃO

Com o constante aumento de usuários conectados à internet, um grande volume de dados tem sido gerado a partir de várias redes. Uma era tecnológica se iniciou e diversos dispositivos estão conectados para colocar em prática os conceitos de cidades inteligentes, casas inteligentes, indústria 4.0, entre outros.

No ambiente de redes, vários métodos de ataque são atualizados constantemente, ocasionando um aumento da escala de impacto e da frequência dos ataques. Os problemas de segurança da rede estão se tornando cada vez mais sérios e, por isso, a segurança cibernética é um tema que está sendo frequentemente abordado.

Assim, alguns sistemas de segurança devem ser utilizados para prevenir os ataques e fornecer a confidencialidade, bem como a disponibilidade de recursos e integridade para as comunicações na Internet. Para determinar e restringir o tráfego malicioso na rede, o Sistema de Detecção de Intrusão (*Intrusion Detection System* (IDS)) tornou-se uma das soluções de segurança de rede mais importantes e mais utilizadas em redes (Liao et al. (2013)). IDSs cada vez mais robustos são constantemente desenvolvidos, visando proteger os dados que são trafegados nas redes, sem no entanto, interferir no seu desempenho.

Na literatura científica, técnicas de seleção de características estão entre as mais empregadas para reduzir conjuntos de dados com alta dimensionalidade, selecionando as características mais relevantes para o processo de construção do modelo do IDS, evitando que dados redundantes ou insignificantes interfiram no treinamento (Naseri e Gharehchopogh (2022) e Gharaee e Hosseinvand (2016)) . Através da técnica de seleção, um subconjunto do conjunto original de dados é usado para simplificar a construção do modelo, a fim de melhorar o tempo de execução, reduzindo o tempo de treinamento. Além disso, busca-se aumentar a performance do sistema, filtrando as características desnecessárias para a identificação de ataques.

A utilização de algotimos bio-inspirados para seleção de características é uma abordagem que tem sido gradualmente utilizada, pois estes algoritmos empregam um método simples, inspirado na natureza, para resolver problemas complexos (Wang e Chen (2020), Abualigah (2018) e Nadimi-Shahraki, Taghian e Mirjalili (2021)). Estes algoritmos buscam a melhor solução para objetivos determinados através da minimização ou maximização de um parâmetro (Diao e Shen (2015)).

O presente trabalho apresenta uma metodologia de análise de características de tráfego de rede, utilizando aprendizado de máquina para classificação e algoritmos bio-inspirados para seleção de características, a fim de detectar intrusões em redes. Foram escolhidos cinco algoritmos bio-inspirados de otimização (ALO, CS, GWO, KH e WOA) para seleção de características das bases de dados NSL-KDD e UNSW-NB15 e foi utilizado o KNN como classificador para criar modelos de IDS.

Para a base de dados NSL-KDD, os melhores resultados obtidos para as métricas de avaliação utilizadas foram por meio do modelo GWO-KNN com 97,82% de acurácia, 97,94%

de *F1-Score*, 98,24% de sensibilidade, 97,54% de precisão, 2,63% de FPR, 1,76% de FNR e 97,37% de TNR, selecionando 11 características das 41 da base de dados. Todos os outros modelos proposto obtiveram resultados similares, mas o que se destacou foi o WOA-KNN que utilizou apenas 5 características para obter tais resultados.

Para a base de dados UNSW-NB15, assim como na base de dados NSL-KDD, o modelo que se destacou foi o GWO-KNN com 93,39% de acurácia, 94,93% de *F1-Score*, 93,40% de sensibilidade, 93,40% de precisão, 6,63% de FPR, 6,63% de FNR e 93,37% de TNR, selecionando 12 características das 42 da base de dados. Os outros modelos porpostos novamente obtiveram resultados similares ao principal citado.

Além disso, foi possivel identificar que as características que representam o volume de dados na rede, o tipo de serviço empregado nas conexões, o estado da conexão e sucesso ou não do login são as que mais influenciam o classificador proposto na detecção de intrusão na base de dados NSL-KDD, enquanto que as características de volume de dados na rede e as características relacionadas à conexão são as que mais tiveram influência no classificador KNN para a base de dados UNSW-NB15.

1.1 OBJETIVO

Este trabalho tem como objetivo analisar e correlacionar as principais características para detecção de intrusão em redes utilizando aprendizado de máquina e algoritmos bio-inpirados.

1.2 ESTRUTURA DO TRABALHO

As estruturas dos demais capítulos deste trabalho serão apresentadas nos parágrafos seguintes.

No Capítulo 2 serão apresentados os trabalhos relacionados disponíveis na literatura, com o objetivo de apresentar o cenário atual de pesquisa da área e o estado da arte.

No Capítulo 3 serão descritos de forma detalhada todos os conceitos utilizados e relacionados ao tema abordado, para que o leitor possa entender com clareza as técnicas que estão sendo empregadas na metodologia proposta e compreender os termos que serão descritos no decorrer do trabalho.

No Capítulo 4 será detalhada a metodologia que foi utilizada para o desenvolvimento deste trabalho, demonstrando as técnicas que foram utilizadas e os passos realizados para atingir o objetivo final.

No Capítulo 5 serão apresentados e discutidos os resultados obtidos, analisando o desempenho do IDS proposto e realizando uma comparação com outros trabalhos.

Por fim, no Capítulo 6, será demonstrada a conclusão do trabalho, sintetizando as principais observações retiradas de todo o seu desenvolvimento.

2 TRABALHOS RELACIONADOS

Diversos autores têm utilizado técnicas de extração de características e classificação para sistemas de detecção de intrusão (IDS). Um aprimoramento do algoritmo de busca cuco (CSA), denominado *Mutation Cuckoo Fuzzy* (MCF), foi proposto por Sarvari et al. (2020) para seleção de características em conjunto com redes neurais artificiais para classificação de IDS baseado em anomalias. Para a fase de seleção de características, foi utilizado o MCF, que integra o operador de mutação com busca de cuco e o agrupamento *Fuzzy C Means* (FCM). Através deste método, a eficiência da busca de cuco para detectar as principais características aumentou. O método de seleção proposto escolheu 22 de 41 características e, para avaliação, foi escolhida parte de uma base de dados comumente utilizada, chamada NSL-KDD.

Por outro lado, Safaldin, Otair e Abualigah (2021) propuseram um IDS aplicando o otimizador lobo cinzento (GWO) para selecionar características e máquina de vetores de suporte (SVM) para classificação de amostras de uma rede de sensores sem fio. A abordagem proposta atinge uma acurácia de 96%, taxa de falso positivo (FPR) de 0,03, taxa de detecção (DR) de 0,96 e tempo de execução de 69,6 h. A escolha de uma boa função de kernel é um desafio e se torna a principal desvantagem do classificador SVM. Além disso, o SVM leva mais tempo para treinar grandes conjuntos de dados e para armazenar todos os vetores de suporte, fazendo com que haja um grande consumo de memória.

Seguindo na mesma linha, Priya et al. (2020) usaram redes neurais profundas para desenvolver um IDS eficiente no ambiente de internet das coisas para classificar e prever ataques cibernéticos. O método proposto também reduz o número de características utilizadas no processo de classificação usando otimizador lobo cinzento (GWO), porém, ainda é utilizada análise de componentes principais (PCA). O conjunto de dados reduzido em seguida é classificado utilizando redes neurais profundas. Para validar a solução, os autores compararam o resultado com outros algoritmos de aprendizado de máquina usando as métricas de precisão, especificação e sensibilidade. Os autores afirmam que o modelo de rede neural profunda proposto tem um desempenho melhor do que as abordagens de aprendizado de máquina existentes, pois acarreta aumento de precisão de 15% e diminuição de tempo de 32%, proporcionando alertas mais rápidos em caso de detecção de intrusão.

No sistema de detecção de intrusão proposto por Riyaz e Ganapathy (2020), foram utilizados campos aleatórios condicionais e um algoritmo de seleção de característica baseado em coeficiente de correlação linear, para classificar características usando uma rede neural convolucional. Os autores relataram uma precisão de 98,88% utilizando o conjunto de dados KDD CUP 99 para testes.

No trabalho de Alazzam, Sharieh e Sabri (2020), foi proposto o algoritmo de otimização inspirado em pombos (PIO) para extração de características, bem como um novo método de binarização de um PIO contínuo. O algoritmo utilizado superou diversos algoritmos de seleção de características de trabalhos relacionados do estado da arte em termos de taxa de verdadeiro

positivo (TPR), taxa de falso positivo (FPR), precisão e F-score. Os testes foram efetuados usando as bases de dados NSL KDD e UNSWNB15. Foi possível verificar também que o método de similaridade de cossenos, proposto para binarizar o algoritmo, tem uma convergência mais rápida que o método sigmóide.

Outro método de extração de características foi proposto por Selvakumar e Muneeswaran (2019), utilizando um algoritmo baseado em vagalumes. Além disso, os autores propuseram um procedimento para diminuir a dimensionalidade dos dados utilizando o método *wrapper based* com rede Bayesiana, C4.5 e informação mútua (MI). Originalmente, a base de dados KDD CUP 99 possui 41 características, mas os resultados desta abordagem mostram que apenas 10 são suficientes para detectar intrusão de forma eficiente, diminuindo assim o custo computacional do classificador.

O método proposto por Nguyen e Kim (2020) consiste em uma rede neural convolucional otimizada usando algoritmo genético (GA-CNN). As características são selecionadas pela rede neural, cuja estrutura é escolhida de forma otimizada usando o GA. A avaliação do método usando a base de dados NSL-KDD, demonstrou que o GA-CNN alcançou 96,2% de precisão de detecção. No entanto, o processo de busca exaustivo do GA aumenta o consumo de tempo, além de ter limitações no manuseio de base de dados desequilibradas.

No trabalho de Salo, Nassif e Essex (2019), foi utilizado um IDS híbrido que combina as abordagens de seleção de característica de informação de ganho (IG) e análise de componentes principais (PCA) com um classificador de conjunto baseado em máquina de vetores de suporte (SVM), aprendizado baseado em instância (IBL) e multicamada Perceptron (MLP). Os autores sugerem que após uma análise comparativa realizada em vários conjuntos de dados IDS, o método IG-PCA apresenta melhor desempenho do que a maioria das abordagens existentes.

Zavrak e Iskefiyeli (2020) propuseram os métodos de autocodificador e autocodificador variacional para um IDS e compararam com o algoritmo de máquina de vetores de suporte de uma classe (OCSVM). Foi realizado um experimento utilizando a base de dados CICDS2017 para extrair as características baseadas em fluxo de rede para uma análise de desempenho dos métodos sob diferentes limites. Os resultados experimentais mostram que o valor da área sob a curva (AUC) obtido pelo autocodificador variacional é 0,7596, provando ser superior ao autocodificador e a OCSVM. No entanto, determinar um limite apropriado que forneça alta precisão de detecção ou baixa taxa de alarmes falsos é um desafio apresentado para o método.

No trabalho de Nancy et al. (2020), foi proposto um novo algoritmo chamado seleção de características dinâmica recursiva (DRFSA), combinando uma extensão do algoritmo árvore de decisão com rede neural convolucional para classificar um grande volume de dados de uma rede de sensores sem fio (WSN). Esse modelo oferece maior precisão de detecção de intrusão, taxa de entrega de pacotes e taxa de transferência, enquanto reduz o atraso da rede e a taxa de falsos negativos.

Seguindo no mesmo objetivo de criar um IDS, muitos projetos utilizam técnicas de aprendizado profundo para propor um sistema de detecção de intrusão eficaz. Jiang et al. (2020) pro-

puseram um sistema IDS eficiente combinando rede neural convolucional e memória longa de curto prazo bidirecional (BiLSTM) em uma hierarquia profunda. O problema de desequilíbrio de classe foi resolvido usando a técnica de sobreamostragem de minoria sintética (SMOTE) para aumentar as amostras minoritárias, auxiliando o modelo a aprender completamente as características. Os experimentos realizados para validar o método utilizaram as bases de dados NSL-KDD e UNSWNB15. A metodologia proposta alcançou maior desempenho em termos de precisão e taxa de detecção, além de ter melhorado a taxa de detecção de classes de dados minoritários. Contudo, esta taxa ainda é consideravelmente mais baixa quando comparada com outras classes de ataque e, devido à estrutura complexa, o tempo de treinamento é maior.

Um sistema de detecção de intrusão baseado em aprendizado profundo para ataques DDoS na agricultura 4.0 foi proposto por Ferrag et al. (2021). Para isso, foram utilizados três modelos, sendo eles rede neural convolucional, rede neural profunda e rede neural recorrente. O modelo IDS baseado na rede neural convolucional superou os métodos de IDS de aprendizado profundo do estado da arte, que foram testados usando as bases de dados CIC-DDoS2019 e TON IoT, registrando uma precisão de 99,95% para detecção de tráfego binário e 99,92% para detecção de tráfego multiclasse.

No trabalho de Choraś e Pawlicki (2021), foram investigados diferentes hiperparâmetros de uma rede neural artificial quando aplicados a bases de dados usuais, como NSL-KDD e CICIDS2017. Estes parâmetros incluem ativação, otimização, tamanho do lote, épocas, camadas e neurônios. A melhor acurácia alcançada foi de 99,9% quando os valores dos parâmetros foram ativação = tanh, otimização = adam, tamanho do lote = 100, épocas = 300, camadas = 1 e neurônios = 25. Contudo, a precisão diminuiu drasticamente com a alteração da função de ativação, indicando que o modelo da rede neural artificial é altamente sensível aos valores dos parâmetros estabelecidos.

Na abordagem de Yang e Wang (2019), foi proposto um IDS que pré-processa os dados de tráfego da rede sem fio e modela o tráfego relacionado à intrusão usando uma versão aprimorada da rede neural convolucional (CNN), denominada ICNN. Este método extrai características de forma autônoma e aplica o método de descida de gradiente estocástico para otimizar os parâmetros da rede. Ao realizar os experimentos utilizando a base de dados KDD CUP 99, os autores demonstraram que o método pode melhorar a precisão, taxa de positivos verdadeiros (TPR) e taxa de falsos verdadeiros (FPR), quando comparados aos métodos DBN e LeNet-5.

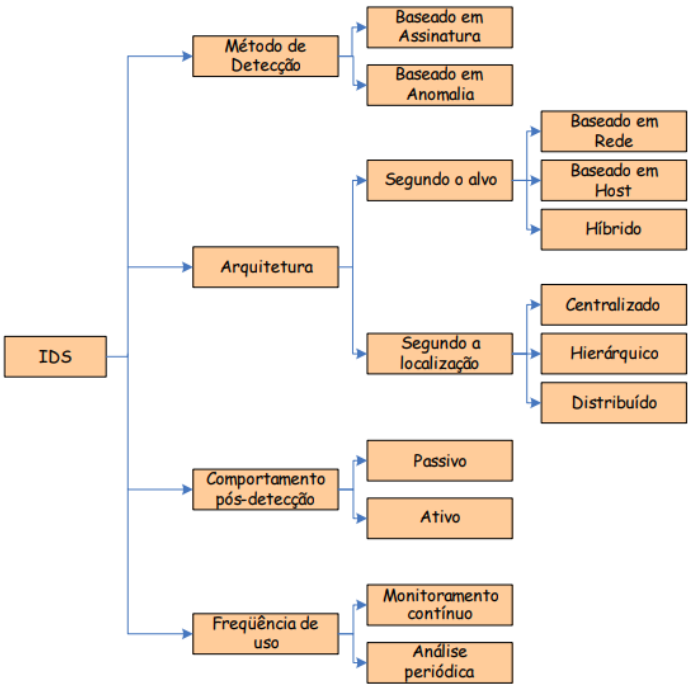
3 CONCEITOS FUNDAMENTAIS

Neste capítulo serão apresentados os conceitos fundamentais utilizados na metodolodia empregada neste trabalho.

3.1 SISTEMAS DE DETECÇÃO DE INTRUSÃO

Sistemas de Detecção de intrusão (IDS - *Intrusion Detection System*), são um mecanismo utilizado em redes no contexto de segurança para relatar um evento suspeito ou impedir a propagação de ações maliciosas na rede. Os IDSs podem ser classificados de acordo com o método de detecção, a arquitetura, o comportamento pós-detecção e frequência de uso, conforme é apresentado na Figura 1.

Figura 1 – Categorias de Classificação de um IDS



Fonte: Sá Silva (2008).

3.1.1 Método de Detecção

O IDS pode ser baseado em assinatura (abuso ou baseado em conhecimento prévio) ou em anomalia (baseado em comportamento). Anomalias são desvios do comportamento normal de uso, enquanto assinaturas são padrões conhecidos de ataques.

O método de detecção por assinatura consiste na identificação de padrões de ataques em bases de dados conhecidas, referentes a atividades hostis ocorridas no passado. Sistemas deste

tipo são altamente eficientes na identificação de ataques, contudo não possuem muita assertividade na identificação de novas ameaças à rede.

O método de detecção por anomalia busca dados não usuais em um conjunto de dados, aplicando medidas matemáticas, estatísticas ou de inteligência artificial para comparar atividades recorrentes de usuários, rede ou sistemas baseado no conhecimento histórico armazenado sobre estes elementos. Os sistemas baseados em anomalias costumam ser computacionalmente mais caros, pois precisam realizar treinamentos extensivos de dados históricos e armazenam uma grande quantidade de dados.

3.1.2 Arquitetura

O IDS pode ser classificado pela sua arquitetura, segundo a localização ou segundo o alvo. Em termos de localização, o IDS pode ter arquitetura centralizada, distribuída ou hierárquica.

3.1.2.1 Arquitetura Segundo Localização

Um sistema de arquitetura centralizada possui seus módulos de detecção instalados em um único *host* e tem como principal vantagem a facilidade no desenvolvimento, instalação e configuração.

Por outro lado, o sistema de arquitetura distribuída possui seus módulos de detecção distribuídos em várias maquinas pela rede, que se comunicam de modo cooperativo através de troca de mensagens para garantir uma redundância inerente. As vantagens desta arquitetura são a robustez e a facilidade de crescimento modular, possibilitando agregar novos mecanismos ao sistema de acordo com a necessidade, retirando de um único ponto o custo e a responsabilidade de todo o processamento, além de ter maior abrangência de detecção, com módulos espalhados pelos mais diferentes pontos do sistema.

A arquitetura hierárquica consiste na distribuição parcial dos componentes do IDS de modo que os módulos de detecção distribuídos fiquem subordinados a alguns módulos gerentes e se comuniquem segundo uma coordenação centralizada, ou seja, os módulos obedecem a uma estrutura hierárquica, com a interação entre os módulos do sistema sendo regida por relações de subordinação. A detecção de falhas no módulos do sistema é facilitada através da estrutura hierárquica, porém a vulnerabilidade do sistema aumenta, principalmente pela criação de pontos únicos de falha, representados pelos módulos mais altos da cadeia hierárquica. Se um desses módulos falhar, todo o sistema pode ficar indisponível, contrapondo-se à principal motivação no uso de sistemas distribuídos.

3.1.2.2 Arquitetura Segundo Alvo

O IDS classificado pelo tipo de alvo a ser monitorado pode ser baseado em *host*, em rede ou híbrido.

Sistemas baseados em host (*Host Intrusion Detection System* - HIDS) utilizam dados coletados na própria máquina ou arquivos de log para monitorar os sistemas, permitindo determinar exatamente quais usuários e processos estão realizando operações maliciosas no sistema. Este tipo de sistema é capaz de monitorar acessos a sistemas e alterações em arquivos de sistemas críticos, uso da CPU e memória, programas que estão sendo executados, modificações nos privilégios dos usuários, processos, entre outros eventos. Estas informações são examinadas em busca de padrões de ataques ou de desvios do comportamento normal de hosts ou usuários.

Sistemas baseados em rede (*Network Intrusion Detection System* - NIDS) realizam o monitoramento do sistema através da captura e análise de cabeçalhos e conteúdo de pacotes de rede, que podem ser comparados com padrões de ataques conhecidos para verificação de algum desvio do comportamento normal da rede. As fontes de informação usadas por este tipo de sistema variam desde dados de gerenciamento até pacotes de rede carregando protocolos.

Em busca de equilíbrio entre desempenho, simplicidade, abrangência e robustez, algumas implementações mais avançadas de IDS coletam diferentes dados de *hosts* e do tráfego de rede, mesclando mecanismos centralizados e mecanismos distribuídos. Grande parte das ferramentas de detecção de intrusão explora o melhor das arquiteturas baseadas em *host* e rede, adotando soluções híbridas.

3.1.3 Comportamento Pós-Detecção

Os Sistema de Detecção de Intrusão baseados no comportamento pós-detecção podem ser ativos ou passivos. O sistema ativo é aquele que é projetado para realizar um bloqueio das conexões caso identifique um pacote malicioso. Este tipo de sistema realiza ações de forma reativa após detectar intrusos na rede. Já o sistema passivo examina as informações da rede e reporta em caso de detecção de intrusão, sem tomar nenhuma ação reativa.

3.1.4 Frequência de Uso

Os IDSs baseados na frequência de uso podem ser separados em monitoramento contínuo e análise periódica. Estes sistemas realizam análises em tempo real das amostras ou análises periódicas em instantes pré-definidos, respectivamente.

3.2 ALGORITMOS BIO-INSPIRADOS

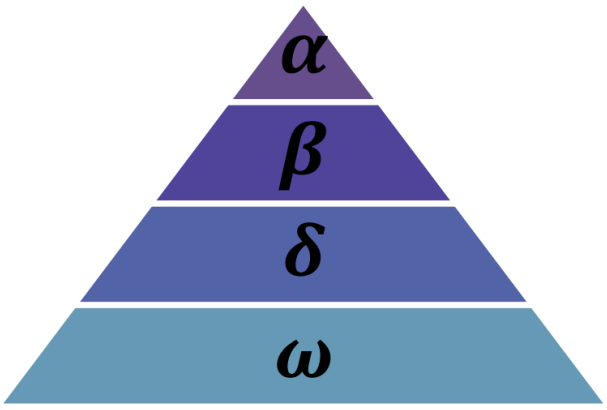
Os algoritmos bio-inspirados simulam diferentes processos biológicos observados na natureza para resolver problemas complexos de otimização de forma eficiente (Molina et al. (2020)). A estratégia de utilizar este tipo de algoritmo para extração de características de redes para detecção de intrusão já tem sido utilizada em outros trabalhos como Sarvari et al. (2020), Safaldin, Otair e Abualigah (2021) e Alazzam, Sharieh e Sabri (2020), contudo, ainda é um campo promissor por conta da quantidade de algoritmos existentes que podem ser aplicados e testados. Além disso, existem diversas possibilidades de combinações com outras técnicas de aprendizado de máquina e aprendizado profundo para detectar invasões de maneira eficiente.

Os algoritmos bio-inspirados deste trabalho foram escolhidos pelos resultados promissores obtidos para problemas complexos, mesmo que em outras áreas como mostra Nadimi-Shahraki, Taghian e Mirjalili (2021), Yang e Deb (2010), Mirjalili, Jangir e Saremi (2016), Wang e Chen (2020) e Abualigah (2018). Desta forma, os algoritmos escolhidos foram ALO (Mirjalili, Jangir e Saremi (2016)), CS (Yang e Deb (2010)), GWO (Nadimi-Shahraki, Taghian e Mirjalili (2021)), KH (Abualigah (2018)) e WOA (Wang e Chen (2020)), que serão descritos em detalhes adiante.

3.2.1 *Grey Wolf Optimizer* (GWO)

O algoritmo GWO, proposto por Mirjalili, Mirjalili e Lewis (2014), se inspira no comportamento de caça dos lobos cinzentos. Este algoritmo simula a hierarquia de liderança destes lobos na natureza, dividindo-os em quatro grupos, conforme mostra a Figura 2.

Figura 2 – Hierarquia de liderança dos lobos cinzentos



Fonte: Mirjalili, Mirjalili e Lewis (2014).

sendo:

a) *Alpha*: Líder do grupo.
b) *Beta*: Segundo em comando.

c) *Delta*: Terceiro em comando.
d) *Omega*: Lobos no menor nível da hierarquia.

Outro aspecto que é inspirado no comportamento dos lobos cinzentos é a caça em grupo que pode ser separado nas seguintes etapas:

a) Busca, rastreamento e aproximação da presa.
b) Perseguição e cerco até a presa parar de se mover.
c) Ataque à presa.

Cada solução do problema de otimização é considerada um lobo cinzento, sendo a solução com a melhor aptidão considerada o lobo *alpha* (α), a segunda melhor o lobo *beta* (β) e a terceira melhor o lobo *delta* (δ). As demais soluções são consideradas como *omega* (ω). A caça dos lobos (otimização) é conduzida por α, β e δ e é seguida pelos lobos ω.

As Equações 1 e 2 representam o modelo matemático do cerco da presa pelos lobos durante a caça.

$$\vec{D} = |\vec{C} \cdot \vec{X}_p(t) - \vec{X}(t)| \tag{1}$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \cdot \vec{D} \tag{2}$$

sendo $\vec{D}$ o vetor de distância do lobo para a presa, t a iteração atual, $\vec{A}$ e $\vec{C}$ são vetores de coeficientes, $\vec{X}_p$ é o vetor de posição da presa e $\vec{X}$ o vetor de posição do lobo cinzento. $\vec{A}$ e $\vec{C}$ são calculados de acordo com as Equações 3 e 4.

$$\vec{A} = 2\vec{a} \cdot \vec{r1} - \vec{a} \tag{3}$$

$$\vec{C} = 2 \cdot \vec{r2} \tag{4}$$

onde $\vec{a}$ é um vetor de componentes que decrescem linearmente de 2 até 0 ao longo das iterações e $\vec{r1}$ e $\vec{r2}$ são vetores aleatórios no intervalo de [0,1].

Simulando matematicamente o comportamento de caça dos lobos, assume-se que α, β e δ possuem um conhecimento melhor sobre a possível localização da presa. Então, as três melhores soluções são armazenadas e os lobos ω (outros agentes de busca) precisam atualizar suas posições de acordo com os três melhores agentes de busca encontrados. As Equações 5 a 7 expressam este comportamento.

$$\vec{D}_\alpha = |\vec{C}_1 \cdot X_\alpha \vec{-} \vec{X}|, \vec{D}_\beta = |\vec{C}_2 \cdot X_\beta \vec{-} \vec{X}|, \vec{D}_\delta = |\vec{C}_3 \cdot X_\delta \vec{-} \vec{X}| \tag{5}$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 \cdot (\vec{D}_\alpha), \vec{X}_2 = \vec{X}_\beta - \vec{A}_2 \cdot (\vec{D}_\beta), \vec{X}_3 = \vec{X}_\delta - \vec{A}_3 \cdot (\vec{D}_\delta) \tag{6}$$

$$X(\vec{t+1}) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{7}$$

Para modelar matematicamente a aproximação da presa, o alcance de $\vec{A}$ diminui conforme o valor de $\vec{a}$ diminui. Desta forma, quando $|\vec{A}| < 1$, o lobo é forçado atacar a presa, direcionando o seu movimento à ela. Porém, quando $|\vec{A}| \geq 1$, o lobo é forçado e divergir desta presa para buscar uma outra presa que resulte em uma solução melhor. O pseudocódigo para a implementação do GWO é mostrado no Algoritmo 1.

Algoritmo 1 – Pseudocódigo do algoritmo GWO

```
Inicializar a população de lobos cinzentos
Inicializar a⃗, A⃗ e C⃗
Calcular a aptidão de todos os agentes de busca
X_α = melhor agente de busca
X_β = segundo melhor agente de busca
X_δ = terceiro melhor agente de busca
while t < número máximo de iterações do
    for para cada agente de busca do
        Atualizar a posição do agente de busca atual de acordo com a Equação (7)
    end
    Atualizar a⃗, A⃗ e C⃗
    Calcular a aptidão de todos os agentes de busca
    Atualizar X_α, X_β e X_δ
    t = t+1
end
Retornar X_α
```

3.2.2 *Cuckoo Search* via *Lévy Flights* (CS)

O *Cuckoo Search* (CS) é um algortimo de otimização inspirado no comportamento parasita da ninhada de algumas espécies de cuco, combinado com o padrão de voos de Lévy encontrado em alguns pássaros. Este algoritmo foi proposto por Yang e Deb (2009) e segue três regras:

a) Cada cuco deposita um ovo aleatoriamente em um ninho escolhido;
b) Os melhores ninhos, que possuem os ovos de maior qualidade, são levados para a próxima geração;
c) O número de ninhos disponíveis é fixo e o ovo colocado por um cuco tem a propabilidade de $p_a \in [0,1]$ de ser descoberto pelo pássaro hospedeiro. Neste caso, o pássaro hospedeiro pode jogar o ovo fora ou abandonar o ninho e construir um novo ninho.

Sendo assim, as três regras citadas do CS são representadas pelo pseudocódigo apresentado pelo Algoritmo 2.

Algoritmo 2 – Pseudocódigo do algoritmo CS

```
Gerar uma população inicial de n ninhos
while t < número máximo de iterações do
    Selecionar um cuco aleatoriamente pelos voos de Lévy e avaliar a sua aptidão F_i
    Selecionar o ovo de um ninho aleatório e avaliar a sua aptidão F_j
    if F_i > F_j then
        Substituir j pela nova solução i
    end
    Uma fração (p_a) dos piores ninhos são abandonados e novos são construídos
    As soluções são ranqueadas e a melhor solução atual é encontrada
end
Retornar o melhor ninho
```

Cada ovo do ninho representa uma solução candidata e um ovo do pássaro cuco representa uma nova solução. O principal objetivo é utilizar potenciais soluções melhores para substituir uma solução pior do ninho. Apenas um ovo é considerado por ninho.

Novas soluções x^{t+1} para um cuco i são geradas realizando voos de Lévy, de acordo com a Equação 8.

$$x_i^{(t+1)} = x_i^{(t)} + \alpha \oplus \text{Lévy}(\lambda), \tag{8}$$

onde $\alpha > 0$ é o tamanho do passo.

O voo de Lévy é responsável por fonecer uma caminhada aleatória pelo espaço de busca, enquanto o tamanho aleatório do passo é definido pela distribuição de Lévy, mostrada na Equação 9, que possui uma variação infinita e média infinita. É possível encontrar algumas soluções próximas da ideal utilizando o voo de Lévy, agilizando a busca local, contudo, parte das novas soluções geradas estão em locais distantes da solução ideal, garantindo que o algoritmo não fique preso em soluções ótimas locais.

$$\text{Lévy } u = t^{-\lambda}, \qquad (1 < \lambda \leq 3), \tag{9}$$

3.2.3 *Krill Herd*(KH)

A técnica KH foi desenvolvido por Gandomi e Alavi (2012) como um algorítimo para problemas de otimização, baseado no comportamento de pastoreio de enxames de krill.

A predação remove indivíduos, leva à redução da densidade média de krill e afasta o enxame de krill do local de alimentação. Este processo é considerado a fase de inicialização no algoritmo KH. No sistema natural, supõe-se que a aptidão de cada indivíduo seja uma combinação da distância do alimento e da maior densidade do enxame de krill. Portanto, a aptidão (distâncias imaginárias) é o valor da função objetivo. A posição dependente do tempo de um indivíduo krill na superfície 2D é regida pelas seguintes três ações principais:

a) Movimento induzido por outros indivíduos krill;
b) Atividade de alimentação; e
c) Difusão aleatória.

As três ações mencionadas acima podem ser representadas matematicamente pela Equação (10).

$$\frac{dX_i}{dt} = N_i + F_i + D_i, \tag{10}$$

onde N_i é o movimento induzido por outro krill; F_i é o movimento de alimentação e D_i é a difusão física dos i-ésimos indivíduos de krill.

3.2.3.1 Movimento induzido por outro krill

De acordo com argumentos teóricos Gandomi e Alavi (2012), os indivíduos de krill tentam manter uma alta densidade e se mover devido a seus efeitos mútuos. A direção do movimento induzido, α_i, é estimada a partir da densidade local do enxame (efeito local), uma densidade alvo do enxame (efeito alvo) e uma densidade repulsiva do enxame (efeito repulsivo). Para um indivíduo krill, esse movimento pode ser definido como:

$$N_i^{novo} = N^{max}\alpha_i + \omega_n N_i^{velho} \tag{11}$$

onde,

$$\alpha_i = \alpha_i^{local} + \alpha_i^{alvo} \tag{12}$$

e N^{max} é a velocidade máxima induzida, ω_n é o peso de inércia do movimento induzido no intervalo [0, 1], N_i^{velho} é o último movimento induzido, α_i^{local} é o efeito local produzido pelos vizinhos e α_i^{alvo} é o alvo efeito de direção produzido pelo melhor indivíduo krill.

Portanto, α_i^{local} pode ser calculado como:

$$\alpha_i^{local} = \sum_{j=1}^{NN} \hat{K}_{ij}\hat{X}_{ij} \tag{13}$$

$$\hat{X}_{ij} = \frac{X_j - X_i}{||X_j - X_i|| + \epsilon} \tag{14}$$

$$\hat{K}_{ij} = \frac{K_i - K_j}{K^{pior} - K^{melhor}} \tag{15}$$

K^{pior} e K^{melhor} são, respectivamente, o krill com melhor e pior aptidão; K_i e k_j representam a aptidão de i-éismo e j-ésimo krill; K_j é a aptidão do J-ésimo vizinho ($j = 1, 2, ..., NN$); X representa as posições relacionadas e NN é o número de vizinhos.

Adicionalmente, $\alpha_i^a lvo$ pode ser descrito como pela Equação (16).

$$\alpha_i^{alvo} = C^{melhor} \hat{K}_{i,melhor} \hat{X}_{i,melhor} \tag{16}$$

onde, C^melhor é o coeficiente efetivo do indivíduo krill com a melhor aptidão para o i-ésimo indivíduo krill.

3.2.3.2 Movimento de alimentação

A segunda ação pode ser representada em duas partes: localização do alimento e sua experiência anterior. Para o krill i, pode ser expresso abaixo pela Equação (17).

$$F_i = V_f \beta_i + \omega_f F_i^{velho} \tag{17}$$

onde

$$\beta_i = \beta_i^{alimento} + \beta_i^{melhor} \tag{18}$$

e V_f é a velocidade de alimentação, ω_f é o peso de inércia em [0,1], F_i^{velho} é o último movimento de alimentação, $\beta_i^{alimento}$ é o alimento atrativo e β_i^{melhor} é o efeito da melhor aptidão do i-ésimo indivíduo krill até o momento.

3.2.3.3 Difusão física

A difusão física dos indivíduos de krill é considerada um processo aleatório. Este movimento pode ser expresso em termos de uma velocidade de difusão máxima e um vetor direcional aleatório. A Equação (19) demonstra esta formulação.

$$D_i = D^{max} \delta \tag{19}$$

onde D^{max} é a velocidade máxima de difusão e δ é um vetor aleatório em [-1, 1].

A partir da junção das partes descritas anteriormente para a técnica do KH, é possível demonstrar o Algoritmo 3 através do pseudocódigo.

3.2.4 *Whale Optimization* (WOA)

O WOA foi proposto por Mirjalili e Lewis (2016) para resolução de problemas de otimização, baseado no mecanismo de caça das baleias jubarte na natureza.

As baleias jubarte caçam cardumes de krill ou pequenos peixes perto da superfície nadando ao redor deles dentro de um círculo cada vez menor e criando bolhas distintas ao longo de um círculo ou caminho em forma de ’9’, como mostra a Figura 3. O processo de caça pode ser separado em três partes:

a) Cercando a presa;
b) Método de ataque rede de bolhas; e
c) Procurar presa.

Algoritmo 3 – Pseudocódigo do KH

Define o tamanho da população A e iterações I_{max}
Inicialização aleatória
  Contador de iterações $I = 1$;
  Inicializa a população A
  Onde, A = $\{A_1, A_2, A_3, ... A_k\}$ são conjuntos de energia calculados que significam indivíduos de krill. Define a velocidade de alimentação, a velocidade máxima de difusão e a velocidade induzida máxima.
Avaliação de aptidão
  Avalia cada indivíduo krill de acordo com sua posição.
while $I < I_{max}$ **do**
    Ordena a população/krill do melhor para o pior.
    for $i = 1 : S$ *(todos os krills)* **do**
      Executa o seguinte cálculo de movimento.
      Movimento induzido por outros indivíduos de krill
      Atividade de alimentação
      Difusão física
      Implementa os operadores genéticos.
      Atualiza a posição do indivíduo krill no espaço de busca.
      Avalie cada indivíduo de krill de acordo com sua posição.
    end
    Ordena a população/krill do melhor para o pior e encontra o atual melhor.
    $I_{max} = I + 1$
end
Avalia a melhor solução de krill.

3.2.4.1 Cercando a presa

As baleias jubarte podem reconhecer a localização das presas e cercá-las. Como a disposição ótima no espaço de busca ainda não é conhecida em um primeiro momento, o algoritmo assume que a presa-alvo atual é a melhor solução ou está próxima do ótimo. Após a definição do melhor agente de busca, os outros agentes de busca tentarão atualizar suas posições em relação a este. Este comportamento pode ser representado pelas Equações 20 e 21.

$$\vec{D} = |\vec{C}\vec{X}^*(t) - \vec{X}(t)| \tag{20}$$

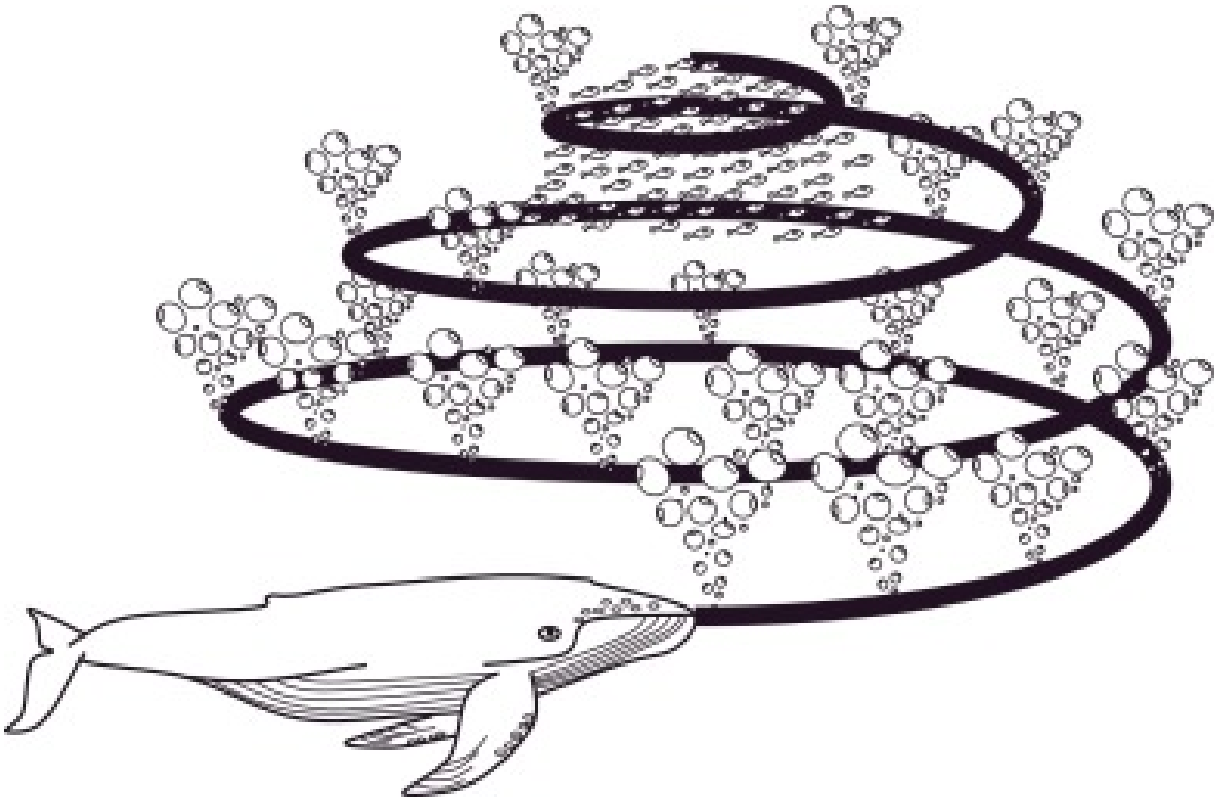
$$\vec{X}(t+1) = \vec{X}^*(t) - \vec{A} \cdot \vec{D} \tag{21}$$

onde t indica a iteração atual, $\vec{A}$ e $\vec{C}$ são vetores de coeficientes, $\vec{X}^*$ é o vetor de posição da presa e $\vec{X}$ indica o vetor de posição de uma baleia.

Os vetores $\vec{A}$ e $\vec{C}$ são representados de acordo com as Equações 22 e 23.

$$\vec{A} = 2\vec{a} \cdot \vec{r} - \vec{a} \tag{22}$$

Figura 3 – Movimento de caça da baleia



Fonte: Mirjalili e Lewis (2016)

$$\vec{C} = 2 \cdot \vec{r} \tag{23}$$

onde os componentes de $\vec{a}$ diminuem linearmente de 2 para 0 ao longo das iterações e $\vec{r}$ é um vetor aleatório em [0,1].

3.2.4.2 Método de ataque rede de bolhas

Para modelar matematicamente o comportamento da rede de bolhas das baleias jubarte, duas abordagens são propostas:

a) Mecanismo de encolhimento: Este comportamento é obtido diminuindo o valor de $\vec{a}$ na Equação 22. O intervalo de flutuação de $\vec{A}$ também é reduzido em $\vec{a}$. Desta forma, $\vec{A}$ é um valor aleatório no intervalo $[-a,a]$, onde a diminui de 2 para 0 ao longo das iterações. Definindo valores aleatórios para $\vec{A}$ em $[-1,1]$, a nova posição de um agente de busca pode ser definida em qualquer lugar entre a posição original do agente e a posição do melhor agente atual.

b) Posição de atualização espiral: Esta abordagem primeiro calcula a distância entre a baleia localizada em (X,Y) e a presa localizada em $(X*,Y*)$. Cria-se uma equação espiral entre a posição da baleia e da presa para imitar o movimento em forma de hélice das baleias jubarte (Equação 24).

$$\vec{X}(t+1) = \vec{D'} \cdot e^{bl} \cdot cos(2\pi l) + \vec{X^*}(t) \tag{24}$$

onde $\vec{D'} = |\vec{X^*}(t) - \vec{X}(t)$ e indica a distância da iésima baleia com relação à presa (considerada a melhor solução até este ponto). A constante b que define a forma da espiral logarítmica e l é um número aleatório em [-1,1].

As baleias jubarte nadam ao redor da presa dentro de um círculo cada vez menor e ao longo de um caminho em espiral de forma simultanea. Para modelar esse comportamento simultâneo, assume-se que há uma probabilidade de 50% de escolher entre o mecanismo de encolhimento ou o modelo espiral para atualizar a posição das baleias durante a otimização. O modelo matemático é mostrado na Equação 25.

$$\vec{X}(t+1) = \begin{cases} \vec{X^*}(t) - \vec{A} \cdot \vec{D} & p < 0{,}5 \\ \vec{D'} \cdot e^{bl} \cdot cos(2\pi l) + \vec{X^*}(t) & p \geq 0{,}5 \end{cases} \tag{25}$$

onde p é um número aleatório em [0,1].

3.2.4.3 Procurar presa

Além do método da rede de bolhas, as baleias jubarte procuram presas aleatoriamente. A mesma abordagem baseada na variação do vetor $\vec{A}$ e $\vec{a}$ pode ser utilizada para procurar presas.

As baleias jubarte procuram de forma aleatória de acordo com a posição umas das outras. Desta forma, é possível utilizar $\vec{A}$ com valores aleatórios maiores que 1 ou menores que -1 para forçar o agente de busca a se afastar de uma baleia de referência. Portanto, a posição de um agente de busca na fase de exploração de acordo com um agente de busca escolhido é atualizada aleatoriamente ao invés de utilizar o melhor agente de busca. Este mecanismo e $|\vec{A}| > 1$ enfatizam a exploração e permitem que o algoritmo WOA realize uma busca global, demonstrada no modelo matemático das Equações 26 e 26.

$$\vec{D} = |C' \cdot \vec{X_{rand}} - \vec{X}| \tag{26}$$

$$\vec{X}(t+1) = \vec{X_{rand}} - \vec{A} \cdot \vec{D} \tag{27}$$

onde $\vec{X_{rand}}$ é um vetor de posição aleatória (uma baleia aleatória).

O pseudocódigo do WOA é apresentado no Algoritmo 4.

3.2.5 Ant Lion Optimizer (ALO)

O ALO é um algoritmo proposto por Mirjalili (2015) com o objetivo de resolver diversos problemas de otimização, que modela matematicamente a interação de formigas e formigas-leão na natureza.

Algoritmo 4 – Pseudocódigo do WOA

```
Inicializa a população de baleias X_i(i = 1, 2, ..., n)
Calcula a aptidão de cada agente de pesquisa.
X* = melhor agente de pesquisa
while (t < número máximo de iterações) do
    for Cada agente de pesquisa do
        Atualiza a, A, C, l e p
        if (p < 0,5) then
            if (|A| < 1) then
                Atualiza a posição do agente de pesquisa atual pela Equação 20
            end
            Seleciona um agente de pesquisa aleatório (X_rand)
            Atualiza a posição do agente de pesquisa atual pela Equação 27
        end
        Atualiza a posição do agente de pesquisa atual pela Equação 24
    end
    Verifica se algum agente de pesquisa ultrapassa o espaço de pesquisa e altera-o.
    Calcula a aptidão de cada agente de busca.
    Atualiza X* se houver uma solução melhor.
    t = t + 1
end
Retorna X*
```

O mecanismo de caça das formigas-leão consiste em preparar pequenas armadilhas em forma de cone para capturar formigas. Formigas-leão sentam-se sob o fosso e esperam que a presa seja capturada (Figura 4). Depois de consumir a carne da presa, as formigas-leão jogam as sobras fora da cova e preparam a cova para a próxima caçada. Foi observado que formigas-leão tendem a cavar um buraco maior quando estão com fome e esta é exatamente a principal inspiração para o algoritmo ALO.

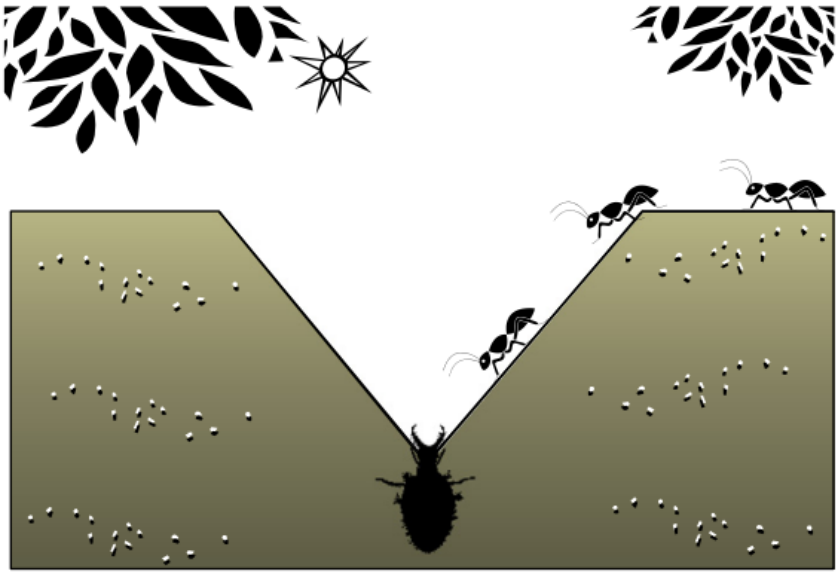
A principal responsabilidade das formigas é explorar o espaço de busca. Elas são obrigadas a se mover pelo espaço de busca usando um passeio aleatório. O passeio aleatório original utilizado no algoritmo ALO para simular o passeio aleatório das formigas é dado pela Equação 28.

$$x(t) = [0, cumsum(2r(t_1) - 1), cumsum(2r(t_2) - 1), ..., cumsum(2r(t_n) - 1)] \tag{28}$$

$$r(t) = \begin{cases} 1 & rand\ 0{,}5 \\ 0 & rand \leq 0{,}5 \end{cases} \tag{29}$$

onde *cumsum* calcula a soma cumulativa, n é o número máximo de iteração, t mostra o passo do passeio aleatório, r(t) é uma função estocástica e *rand* é um número aleatório gerado com distribuição uniforme no intervalo de [0, 1].

Figura 4 – Armadilha em formato de cone das formigas-leão



Fonte: Mirjalili (2015)

A fim de manter o passeio aleatório nos limites do espaço de busca e evitar que as formigas ultrapassem, os passeios aleatórios devem ser normalizados usando a Equação 30.

$$X_i^t = \frac{(X_i^t - a_i) \times (d_i - c_i^t)}{(d_i^t - a_i)} + c_i^t \tag{30}$$

onde c_i^t é o mínimo da i-ésima variável da t-ésima iteração, d_i^t indica o máximo da i-ésima variável da t-ésima iteração, a_i é o mínimo do passeio aleatório da i-ésima variável e b_i é o máximo do passeio aleatório da i-ésima variável.

O ALO simula o aprisionamento de formigas nos poços das formigas-leão alterando os passeios aleatórios em torno das formigas-leão, demonstrado pelas Equações 31 e 32.

$$c_i^t = Antlion_j^t + c^t \tag{31}$$

$$d_i^t = Antlion_j^t + d^t \tag{32}$$

onde c_t é o mínimo de todas as variáveis na t-ésima iteração, d_t indica o vetor incluindo o máximo de todas as variáveis na t-ésima iteração, c_i^t é o mínimo de todas as variáveis para a i-ésima formiga, d_i^t é o máximo de todas as variáveis para a i-ésimo formiga e $Antliont_j$ mostra a posição da j-ésimo formiga-leão selecionada na t-ésima iteração.

Na natureza, as formigas-leão maiores constroem poços maiores para aumentar suas chances de sobrevivência. Sendo assim, o ALO utiliza um operador de roleta que seleciona formigas-leão com base em seu valor de aptidão. A roleta auxilia as mais aptas a atrair mais formigas. Para imitar as formigas deslizando em direção às formigas-leão, os limites dos passeios aleatórios devem ser diminuídos adaptativamente utilizando as Equações 33 e 34.

$$c^t = \frac{c^t}{I} \tag{33}$$

$$d^t = \frac{d^t}{I} \tag{34}$$

onde I é uma taxa, c_t é o mínimo de todas variáveis na t-ésima iteração, d_t indica o vetor incluindo todos os máximos de todas as variáveis na t-ésima iteração.

A próxima etapa do algoritmo é pegar a formiga e reconstruir o poço. A Equação 35 simula este comportamento.

$$Antlion_j^t = Ant_i^t \quad if \quad f(Ant_i^t) > f(Antlion_j^t) \tag{35}$$

onde t mostra a iteração atual, $Antliont_j$ mostra a posição da j-ésima formiga-leão selecionada na t-ésima iteração e $Antliont_i$ indica a posição da i-ésimo formiga na t-ésima iteração.

Por fim, o último operador do ALO é o elitismo, no qual é armazenada a formiga-leão mais apta formada durante a otimização. Esta é a única formiga-leão capaz de impactar todas as formigas. Isso significa que os passeios aleatórios das formigas-leão gravitam em torno de uma formiga-leão selecionada (escolhida usando a roleta) e a formiga-leão de elite. A Equação 36 modela o elitismo.

$$Ant_i^t = \frac{R_A^t + R_E^t}{2} \tag{36}$$

O pseudocódigo do ALO é apresentado no Algoritmo 5.

Algoritmo 5 – Pseudocódigo do algoritmo ALO.

```
Inicializa a população de formigas e formigas leão aleatoriamente
Calcula a aptidão das formigas e das formigas leão
Encontra a melhor formiga leão e assume como ótimo
while Condição final não satisfeita do
    for cada formiga do
        Seleciona a formiga leão utilizando a roleta
        Atualiza c e d usando as Equações 33 e 34
        Cria um passeio aleatória e a normaliza usando as Equações 28 e 30
        Atualiza a posição da formiga usando a Equação 36
    end
    Calcula a aptidão de todas as formigas Substitui uma formiga leão com sua
      formiga correspondente mais apta pela Equação 35
    Atualiza a elite se uma formiga leão ficar mais apta que a elite
end
```

3.3 CLASSIFICAÇÃO DE DADOS

3.3.1 K-vizinhos mais próximos (KNN)

O K-vizinhos mais próximos (KNN) é uma técnica não paramétrica de aprendizado supervisionado que usa a proximidade para fazer classificações ou previsões sobre o agrupamento de um ponto de dados individual (Cover e Hart (1967)).

Uma amostra é classificada pela pluralidade dos votos de seus vizinhos, com esta amostra sendo atribuída à classe mais comum entre seus k vizinhos mais próximos, sendo k um inteiro positivo. Desta forma, é calculada a distância do dado a ser classificado com relação a todos os outros dados conhecidos. Para isso, são utilizadas técnicas como distância Euclidiana, distância Manhattan, distância Minkowski, ditância Hamming entre outros.

As etapas abaixo mostram o procedimento para implementação do KNN.

a) Recebe amostra não classificada;
b) Mede a distância da amostra com relação a todos os outros dados já classificados;
c) Obtem as k menores distâncias;
d) Determina a classe que mais apareceu entre as k menores distâncias obtidas; e
e) Classifica a amostra com a classe determinada.

3.3.2 Validação Cruzada *k-Fold*

A validação cruzada *k-Fold* é uma técnica empregada para avaliar a capacidade de generalização de um modelo, utilizando um conjunto de dados. Portanto, esta técnica avalia o quanto o modelo treinado consegue predizer dados que não foram utilizados no treinamento Kohavi (1995)).

Para a aplicação desta técnica, é realizada uma divisão do conjunto total de dados em k subconjuntos do mesmo tamanho mutuamente exclusivos. Em seguida, um dos subconjuntos é utilizado para a etapa de testes enquanto os outros subconjuntos são utilizados para a etapa de treinamento. Este processo é repetido k vezes até que todos os subconjuntos tenham sido utilizados para a validação do modelo. Ao final da validação de todos os subconjuntos, utiliza-se a acurácia média obtida destes subconjuntos para o modelo proposto.

As etapas abaixo mostram o procedimento da validação cruzada *k-fold*.

a) Embaralhamento da base de dados;
b) Divisão da base de dados em k conjuntos;
c) Em cada conjunto criado da base de dados:
   - Seleciona o conjunto atual para validação;
   - Seleciona o restante dos conjuntos para treinamento;
   - Treina um modelo com nova base de treinamento e efetua uma avaliação com a nova base de testes; e

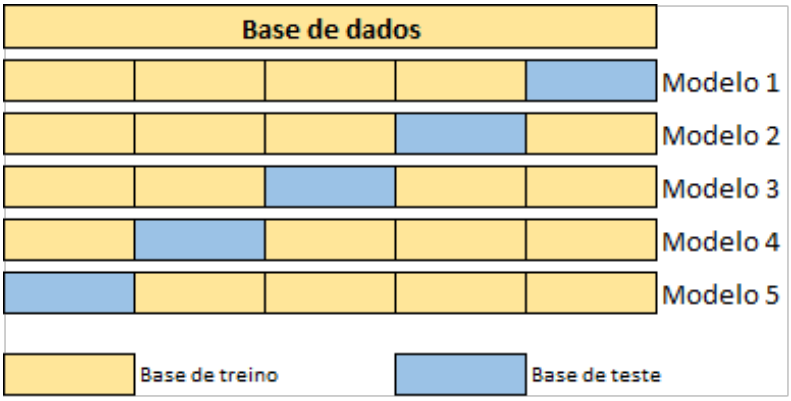
– Armazena os resultados obtidos pela avaliação do modelo.

d) Calcula a precisão do modelo a partir dos resultados obtidos de cada conjunto; e

e) Inicia novamente o processo com o próximo conjunto ou calcula-se a acurácia média.

A Figura 5 apresenta a aplicação do método *k-fold* para a divisão de uma base de dados com o valor de $k = 5$.

Figura 5 – Exemplo da divisão *k-fold* com $k = 5$.



Fonte: Autor.

3.3.3 Normalização Min-Max

A normalização é uma etapa de pré-processamento fundamental em qualquer tipo de declaração de problema (Patro e Sahu (2015)). A normalização assume um papel importante que pode reduzir o poder computacional necessário em um processamento por meio da redução da escala de dados antes de serem usados em etapas posteriores.

A técnica que fornece transformação linear na faixa original de dados é chamada normalização Min-Mix. Esta normalização costuma ser simples e muito usual, pois pode ajustar especificamente os dados em um limite pré-definido. A normalização Min-Max é dada pela Equação 37,

$$a' = \left(\frac{a - \text{Min valor de} A}{\text{Max valor de} A - \text{min valor de} A} \right) \tag{37}$$

onde a é um valor do vetor A e a' é o novo valor encontrado.

3.4 MÉTRICAS DE AVALIAÇÃO

Nesta seção serão apresentadas as métricas utilizadas para avaliação da qualidade dos modelos treinados utilizando as técnicas de extração de características e a função de custo utilizada.

3.4.1 Matriz de confusão

De acordo com DrEng (2001), a matriz de confusão é uma tabela que permite a visualização do desempenho de um algoritmo de classificação. Esta tabela possui quatro parâmetros que são obtidos através da comparação de um valor previsto e do valor encontrado. Os parâmetros são chamados de *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) e *False Negative* (FN) e são descritos abaixo.

a) Verdadeiro Positivo (TP): Item corretamente classificado quanto à classe de interesse;
b) Verdadeiro negativo (TN): Item corretamente classificado como não sendo da classe de interesse;
c) Falso positivo (FP): Item erroneamente classificado quanto à classe de interesse; e
d) Falso negativo (FN): Item erroneamente classificado como não sendo da classe de interesse.

Com os parâmetros obtidos na matriz de confusão, é possível extrair métricas de avaliação para o desempenho de um classificador. As métricas utilizadas neste trabalho são apresentadas a seguir.

3.4.1.1 Acurácia

A acurácia representa a proporção das amostras que foram corretamente classificadas pelo classificador dentre todas as amostras, sendo definida pela Equação (38).

$$\text{Acurácia} = \frac{TP + TN}{TP + TN + FP + FN} \tag{38}$$

3.4.1.2 sensibilidade

A sensibilidade ou taxa de detecção representa a proporção das amostras que foram corretamente classificadas como positivo pelo classificador dentre as amostras de verdadeiro positivo e falso negativo, representada pela Equação (39).

$$\text{sensibilidade} = \frac{TP}{TP + FN} \tag{39}$$

3.4.1.3 Precisão

A precisão representa a proporção das amostras que foram corretamente classificadas como positivo pelo classificador dentre todas as amostras classsificadas como positivo, como mostra a Equação (40).

$$\text{Precisão} = \frac{TP}{TP + FP} \tag{40}$$

3.4.1.4 F1-Score

O *F1-Score* é uma medida que combina a precisão e a sensibilidade, calculando a média harmônica entre elas. Por se tratar de uma média harmônica, o *F1-Score* sempre acompanha o menor valor, ou seja, se a precisão ou a sensibilidade tiverem um valor baixo, o *F1-Score* também terá. Por outro lado, se o valor do *F1-Score* for alto, o modelo possui alta capacidade de predição. A Equação (41) demonstra o comportamento descrito.

$$\text{F1-Score} = 2 * \frac{\text{Precisão} * \text{sensibilidade}}{\text{Precisão} + \text{sensibilidade}} \tag{41}$$

3.4.1.5 Taxa de falsos positivos (FPR)

A taxa de falsos positivos ou taxa de alarmes falsos representa a proporção das amostras que foram erroneamente classificadas como positivo pelo classificador dentre as amostras de verdadeiro negativo e falso positivo, representada pela Equação (42).

$$\text{FPR} = \frac{FP}{TN + FP} \tag{42}$$

3.4.1.6 Taxa de falsos negativos (FNR)

A taxa de falsos negativos representa a proporção das amostras que foram erroneamente classificadas como negativo pelo classificador dentre as amostras de verdadeiro positivo e falso negativo, representada pela Equação (43).

$$\text{FNR} = \frac{FN}{TP + FN} \tag{43}$$

3.4.1.7 Taxa de verdadeiros negativos (TNR)

A taxa de verdadeiros negativos representa a proporção das amostras que foram corretamente classificadas como negativo pelo classificador dentre as amostras de verdadeiro negativo e falso positivo, representada pela Equação (44).

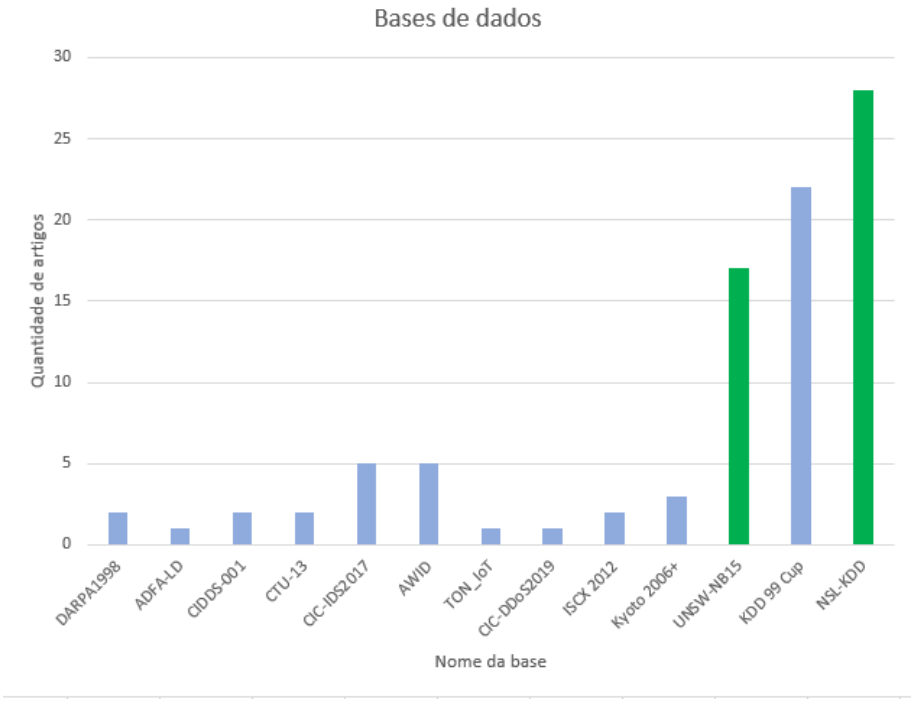
$$\text{TNR} = \frac{TN}{TN + FP} \tag{44}$$

3.5 BASES DE DADOS

Esta seção apresenta os detalhes das bases de dados utilizadas neste trabalho. Para selecionar estas bases, foi feito um levantamento dos principais trabalhos relacionados para identificar quais eram as bases mais utilizadas para criar IDSs, como mostra a Figura 6. Após o levantamento realizado foi identificado que as bases de dados mais utilizadas são a NSL-KDD, KDD Cup 99 e UNSW-NB15.

Decidiu-se utilizar duas bases de dados para este trabalho, contudo, ao analisar as três bases citadas com mais detalhes, observou-se que a base NSL-KDD é uma versão atualizada da base KDD Cup 99. Além disso, os trabalhos mais recentes que focam na elaboração de um IDS utilizam a versão atualizada da base (NSL-KDD). Sendo assim, a fim de obter uma variação maior de dados, foram selecionadas as bases NSL-KDD e UNSW-NB15.

Figura 6 – Levantamento das bases de dados utilizadas para criação de IDS



Fonte: Autor.

3.5.1 NSL-KDD

A base NSL-KDD, disponibilizada pela Universidade de New Brunswick (UNB) [1], é uma base de dados que contém registros de tráfego de dados de usuários na internet e foi proposta como uma versão revisada da conhecida base de dados KDD Cup 99 Tavallaee et al. (2009). A KDD Cup 99 possui alguns elementos desnecessários nos conjuntos de teste e treinamento.

[1] https://www.unb.ca/cic/datasets/nsl.html

Esses elementos redundantes têm um efeito negativo significativo no desempenho dos sistemas de detecção de intrusão IDS.

Desta forma, o conjunto de dados NSL-KDD é estruturado sem elementos desnecessários e repetições. A base NSL-KDD possui 43 colunas, sendo separadas em características (1 a 41), categoria (42) e grau de dificuldade (43). As Tabela 2 mostra o signficado de cada uma das colunas desta base de dados.

As amostras são separadas entre as categorias "ataque"ou "normal"e os ataques são divididos em quatro grupos, sendo eles DoS, Probe, R2L e U2R, explicados mais abaixo nesta Seção 3.5.1.

Por fim, a Tabela 1 mostra a relação de amostras encontradas na base de dados em questão.

Tabela 1 – Descrição das amostras do NSL-KDD

Classe	**Treino**	**Teste**
Normal	67343	9711
Dos	45927	7458
Probe	11656	2421
U2R	52	200
R2L	995	2754
Total	125973	22544

Tabela 2 – Descrição da base NSL-KDD

Coluna	**Nome**	**Tipo**	**Descrição**
1	Duration	Numérico	Duração da conexão
2	Protocol_type	Nominal	Protocolo utilizado na conexão
3	Service	Nominal	Serviço de rede de destino usado
4	Flag	Nominal	Status da conexão—Normal ou Erro
5	Src_bytes	Numérico	Número de bytes de dados transferidos da origem ao destino em uma única conexão
6	Dst_bytes	Numérico	Número de bytes de dados transferidos do destino para a origem em uma única conexão
7	Land	Binário	Se os endereços IP de origem e destino e os números de porta forem iguais, essa variável assume o valor 1 senão 0
8	Wrong_fragment	Numérico	Número total de fragmentos errados nesta conexão
			Continua na próxima página

Tabela 2 – continuação da página anterior

Coluna	Nome	Tipo	Descrição
9	Urgent	Numérico	Número de pacotes urgentes nesta conexão. Pacotes urgentes são pacotes com o bit urgente ativado
10	Hot	Numérico	Número de indicadores "quentes"no conteúdo, como: entrar em um diretório do sistema, criar programas e executar programas
11	Num_failed_logins	Numérico	Contagem de tentativas de login com falha
12	Logged_in	Binário	Status de Login: 1 se logado com sucesso; 0 caso contrário
13	Num_compromised	Numérico	Número de condições "comprometidas"
14	Root_shell	Binário	1 se o "root shell"for obtido; 0 caso contrário
15	Su_attempted	Binário	1 se o comando ‘su root’ for tentado ou usado; 0 caso contrário
16	Num_root	Numérico	Número de acessos ‘root’ ou número de operações realizadas como root na conexão
17	Num_file_creations	Numérico	Número de operações de criação de arquivo na conexão
18	Num_shells	Numérico	Número de prompts do shell
19	Num_access_files	Numérico	Número de operações em arquivos de controle de acesso
20	Num_outbo und_cmds	Numérico	Número de comandos de saída em uma sessão de FTP
21	Is_hot_login	Binário	1 se o login pertencer à lista ‘hot’, ou seja, root ou admin; senão 0
22	Is_guest_login	Binário	1 se o login for um login ‘convidado’; 0 caso contrário
23	Count	Numérico	Número de conexões com o mesmo host de destino da conexão atual nos últimos dois segundos

Continua na próxima página

Tabela 2 – continuação da página anterior

Coluna	Nome	Tipo	Descrição
24	Srv_count	Numérico	Número de conexões para o mesmo serviço (número da porta) da conexão atual nos últimos dois segundos
25	Serror_rate	Numérico	A porcentagem de conexões que ativaram o sinalizador (4) s0, s1, s2 ou s3, entre as conexões agregadas na contagem (23)
26	Srv_serror_rate	Numérico	A porcentagem de conexões que ativaram o sinalizador (4) s0, s1, s2 ou s3, entre as conexões agregadas em srv_count (24)
27	Rerror_rate	Numérico	A porcentagem de conexões que ativaram o sinalizador (4) REJ, entre as conexões agregadas na contagem (23)
28	Srv_rerror_rate	Numérico	A porcentagem de conexões que ativaram o sinalizador (4) REJ, entre as conexões agregadas em srv_count (24)
29	Same_srv_rate	Numérico	A porcentagem de conexões que foram para o mesmo serviço, entre as conexões agregadas na contagem (23)
30	Diff_srv_rate	Numérico	A porcentagem de conexões que foram para serviços diferentes, entre as conexões agregadas na contagem (23)
31	Srv_diff_host_rate	Numérico	A porcentagem de conexões que eram para diferentes máquinas de destino entre as conexões agregadas em srv_count (24)
32	Dst_host_count	Numérico	Número de conexões com o mesmo endereço IP do host de destino
33	Dst_host_srv_count	Numérico	Número de conexões com o mesmo número de porta

Continua na próxima página

Tabela 2 – continuação da página anterior

Coluna	Nome	Tipo	Descrição
34	Dst_host_same_srv_rate	Numérico	A porcentagem de conexões que foram para o mesmo serviço, entre as conexões agregadas em dst_host_count (32)
35	Dst_host_diff_ srv_rate	Numérico	A porcentagem de conexões que foram para serviços diferentes, entre as conexões agregadas em dst_host_count (32)
36	Dst_host_same_src_port_rate	Numérico	A porcentagem de conexões que estavam na mesma porta de origem, entre as conexões agregadas em dst_host_srv_count (33)
37	Dst_host_srv_diff_host_rate	Numérico	A porcentagem de conexões que foram para diferentes máquinas de destino, entre as conexões agregadas em dst_host_srv_count (33)
38	Dst_host_serror_rate	Numérico	A porcentagem de conexões que ativaram o sinalizador (4) s0, s1, s2 ou s3, entre as conexões agregadas em dst_host_count (32)
39	Dst_host_srv_s error_rate	Numérico	A porcentagem de conexões que ativaram o sinalizador (4) s0, s1, s2 ou s3, entre as conexões agregadas em dst_host_srv_count (33)
40	Dst_host_erro r_rate	Numérico	A porcentagem de conexões que ativaram o sinalizador (4) REJ, entre as conexões agregadas em dst_host_count (32)
41	Dst_host_srv_r error_rate	Numérico	A porcentagem de conexões que ativaram o sinalizador (4) REJ, entre as conexões agregadas em dst_host_srv_count (33)
42	Classificação da amostra	Nominal	Classificação da amostra entre ataque ou normal
43	Grau de dificuldade	Numérico	-

3.5.2 UNSW-NB15

A base de dados UNSW-NB15 foi proposta em 2015 pela Universidade de Nova Gales do Sul (UNSW Sydney)[2] para que sistemas de detecção de intrusão fossem desenvolvidos. O subconjunto desta base utilizado neste trabalho contém aproximadamente 250 mil amostras separadas entre teste e treino, como mostra a Tabela 3.

A base UNSW-NB15 possui 45 colunas, sendo separadas pelo identificador da amostra (1), características de navegação (2 a 43), tipo de ataque (44) e classe da amostra (45). A Tabela 4 mostra a descrição e organização desta base de dados.

Diferentemente da base NSL-KDD que possui 4 tipo de ataques em suas amostras, a base UNSW-NB15 apresenta 9 tipos de ataques, sendo eles Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode e Worms que estão explicados mais abaixo nesta seção.

Tabela 3 – Descrição das amostras do UNSW-NB15

Classe	**Treino**	**Teste**
Normal	56000	37000
Analysis	2000	677
Backdoor	1746	583
DoS	12264	4089
Exploits	33393	11132
Fuzzers	18184	6062
Generic	40000	18871
Reconnaissance	10491	3496
Shellcode	1133	378
Worms	130	44
Total	175341	82332

Tabela 4 – Descrição da base UNSW-NB15

Coluna	**Nome**	**Tipo**	**Descrição**
1	ID	Numérico	Número de identificação da amostra
2	dur	Nominal	Duração total do registro
3	proto	Numérico	Protocolo de transação
4	service	Nominal	Serviço (http, ftp, ssh, dns ...)
5	state	Numérico	O estado e seu protocolo dependente, por exemplo ACC, CLO, senão (-)
Continua na próxima página			

[2] https://research.unsw.edu.au/projects/unsw-nb15-dataset

Tabela 4 – continuação da página anterior

Coluna	Nome	Tipo	Descrição
6	spkts	Nominal	Contagem de pacotes de origem para destino
7	dpkts	Nominal	Contagem de pacotes de destino para origem
8	sbytes	Numérico	Bytes de origem para destino
9	dbytes	Numérico	Bytes de destino para origem
10	rate	Numérico	Taxa de conexão
11	sttl	Numérico	Tempo de vida da origem ao destino
12	dttl	Numérico	Tempo de vida do destino à origem
13	sload	Numérico	Bits de origem por segundo
14	dload	Numérico	Bits de destino por segundo
15	sloss	Nominal	Pacotes da origem retransmitidos ou descartados
16	dloss	Numérico	Pacotes do destino retransmitidos ou descartados
17	sintpkt	Numérico	Tempo de chegada entre pacotes da origem (mSec)
18	dintpkt	Numérico	Tempo de chegada entre pacotes do destino (mSec)
19	sjit	Numérico	Tremulação da origem (mSec)
20	djit	Numérico	Tremulação do destino (mSec)
21	swin	Numérico	Propaganda da janela TCP da origem
22	stcpb	Numérico	Número de sequência TCP da origem
23	dtcpb	Numérico	Número de sequência TCP do destino
24	dwin	Numérico	Anúncio da janela TCP de destino
25	tcprtt	Numérico	A soma de 'synack' e 'ackdat' do TCP
26	synack	Numérico	O tempo entre os pacotes SYN e SYN_ACK do TCP.
27	ackdat	Numérico	O tempo entre os pacotes SYN_ACK e ACK do TCP

Continua na próxima página

Tabela 4 – continuação da página anterior

Coluna	Nome	Tipo	Descrição
28	smean	Numérico	Média do tamanho do pacote de fluxo transmitido pelo src
29	dmean	Numérico	Média do tamanho do pacote de fluxo transmitido pelo dst
30	trans_depth	Numérico	A profundidade na conexão da transação de solicitação/resposta http
31	response_body_len	Numérico	O tamanho do conteúdo dos dados transferidos do serviço http do servidor.
32	ct_srv_src	Numérico	Nº de conexões que contêm o mesmo serviço (14) e endereço de origem (1) em 100 conexões de acordo com a última vez (26)
33	ct_state_ttl	Numérico	Nº para cada estado (6) de acordo com a faixa específica de valores para tempo de vida de origem/destino (10) (11)
34	ct_dst_ltm	Numérico	Nº de conexões do mesmo endereço de destino (3) em 100 conexões de acordo com a última vez (26)
35	ct_src_dport_ltm	Numérico	Nº de conexões do mesmo endereço de origem (1) e porta de destino (4) em 100 conexões de acordo com o último tempo (26).
36	ct_dst_sport_ltm	Numérico	Nº de conexões do mesmo endereço de destino (3) e da porta de origem (2) em 100 conexões de acordo com o último tempo (26).
37	ct_dst_src_ltm	Binário	Número de conexões do mesmo endereço de origem (1) e destino (3) em 100 conexões de acordo com a última vez (26).
38	is_ftp_login	Numérico	Se a sessão ftp for acessada por usuário e senha, então 1 senão 0

Continua na próxima página

Tabela 4 – continuação da página anterior

Coluna	Nome	Tipo	Descrição
39	ct_ftp_cmd	Numérico	Nº de fluxos que possui comando na sessão ftp
40	ct_flw_http_mthd	Binário	Nº de fluxos que possui métodos como Get e Post no serviço http.
41	ct_src_ltm	Numérico	Nº de conexões do mesmo endereço de origem (1) em 100 conexões de acordo com a última vez (26)
42	ct_srv_dst	Numérico	Nº de conexões que contém o mesmo serviço (14) e endereço de destino (3) em 100 conexões de acordo com o último horário (26)
43	is_sm_ips_ports	Numérico	Se a origem (1) for igual ao destino (3) endereços IP e números de porta (2) (4) forem iguais, esta variável assume o valor 1 senão 0
44	Tipo de ataque	Nominal	Especifica o tipo de ataque
45	Classe	Nominal	Clase da amostra "ataque"ou "normal"

3.5.3 Tipos de ataques encontrados nas bases

Diversos tipos de ataques são encontrados nas bases de dados citadas e cada um deles é descrito com mais detalhes abaixo.

a) **DoS - *Denial of Service*** é um ataque que tenta interromper o fluxo de tráfego de e para (*from to*) o sistema de destino. Um exemplo de implementação do DoS é dado por um sistema que é inundado com uma quantidade anormal de tráfego que ele não consegue controlar e desliga para se proteger, o que impede que o tráfego normal visite uma rede.

b) ***Probe*** ou vigilância é um ataque que tenta obter informações de uma rede. O objetivo é agir como um ladrão e roubar informações importantes, sejam informações pessoais de clientes ou informações bancárias.

c) **U2R** (*User to Root*) é um ataque que começa com uma conta de usuário normal e tenta obter acesso ao sistema ou rede como um superusuário (*root*). O invasor tenta explorar as vulnerabilidades em um sistema para obter privilégios/acesso *root*.

d) **R2L** (*Remote to Local*) é um ataque que tenta obter acesso local a uma máquina remota. O invasor não tem acesso local ao sistema/rede e tenta “hackear” seu caminho para a rede.

e) ***Analysis*** é um ataque de análise de tráfego, no qual um hacker tenta acessar a mesma rede que o usuário para ouvir e capturar todo o tráfego de sua rede.

f) ***Backdoor*** é um ataque que utiliza um *malware* para obter acesso não autorizado à rede enquanto contorna todas as medidas de segurança implementadas.

g) ***Exploits*** do verbo inglês *"to exploit"*, que significa "usar algo em benefício próprio", é um pedaço de software que se aproveita de um *bug* ou vulnerabilidade para causar ataques imprevistos. Quando usados, os exploits permitem que um intruso acesse remotamente uma rede e obtenha privilégios elevados.

h) ***Fuzzers*** é um tipo de ataque que busca detectar *bugs* de forma automática, possibilitando que aplicativos sejam estressados e causam comportamentos inesperados, vazamento de recursos ou travamentos.

i) ***Generic*** trata-se de um "ataque genérico"contra uma primitiva criptográfica, que pode ser executado independentemente dos detalhes de como essa primitiva criptográfica é implementada.

j) ***Reconnaissance*** ou ataque de reconhecimento é um tipo de ataque de segurança em que um invasor usa para reunir todas as informações possíveis sobre o alvo antes de lançar um ataque real. O invasor usa um ataque de reconhecimento como uma ferramenta de preparação para um ataque real.

k) ***Shellcode*** é um ataque em que o hacker executa uma sequência de código de máquina ou instruções executáveis, que é injetada na memória de um computador com a intenção de assumir o controle de um programa em execução.

l) ***Worms*** é um tipo de ataque onde um programa autônomo de *malware* se replica para se espalhar para outros computadores, utilizando geralmente uma rede de computadores, contando com falhas de segurança no computador de destino para acessá-lo.

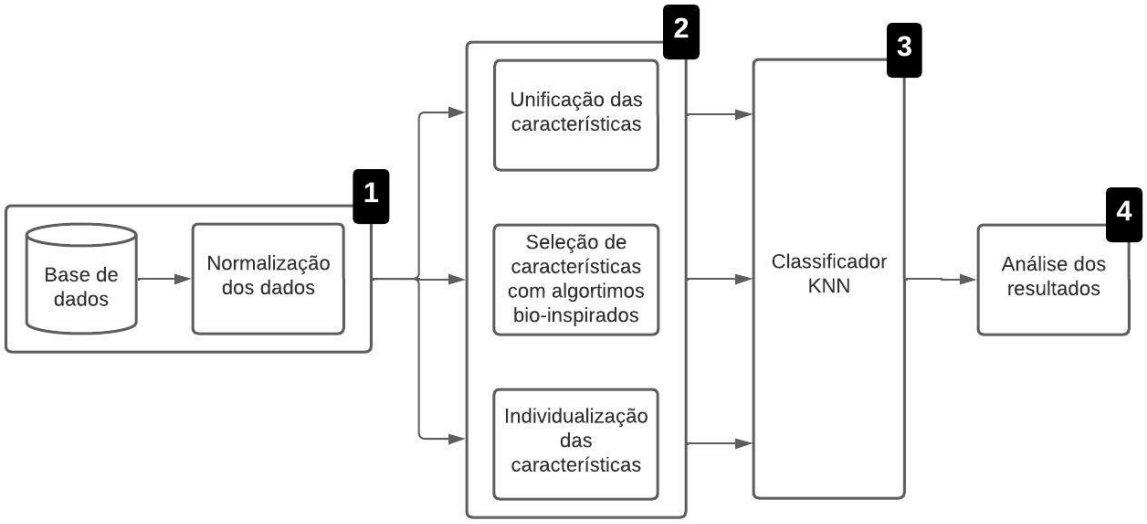
4 METODOLOGIA

Neste capítulo será apresentada a metodologia proposta para analisar e correlacionar as características de fluxo de navegação em redes para detecção de intrusão. Inicialmente, será apresentado o fluxograma geral da estrutura do trabalho e, em seguida, cada bloco será descrito detalhadamente.

O fluxograma da Figura 7 apresenta a metodologia proposta. Na primeira etapa é realizada a extração de todas as características das bases de dados escolhidas para executar a normalização dos dados. Em seguida, as características passam por uma etapa de processamento separada em três grupos: unificação das características, seleção de características com algoritmos bio-inspirados e separação das características.

Logo após, os dados obtidos de cada um destes grupos serão utilizados como parâmetros de entrada da terceira etapa que é responsável por classificar os dados. Por fim, os resultados obtidos serão analisados e comparados baseados em métricas de classificação.

Figura 7 – Fluxograma da metodologia proposta neste trabalho.



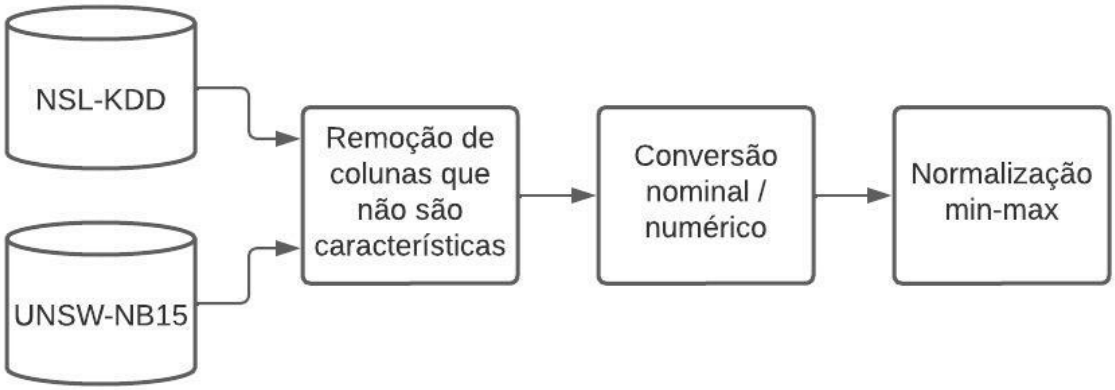
Fonte: Autor.

As Seções 4.1, 4.2, 4.3 e 4.4 explicam de forma detalhada as etapas 1, 2, 3 e 4 da Figura 7, respectivamente.

4.1 CONFIGURAÇÃO DOS DADOS

A primeira etapa da metodologia proposta é responsável por preparar os dados de entrada que serão utilizados nas etapas subsequentes. Esta etapa é composta pela extração das características das bases de dados e pela normalização delas. A Figura 8 mostra o fluxograma da configuração dos dados que será detalhado no decorrer desta seção.

Figura 8 – Fluxograma da configuração dos dados



Fonte: Autor.

4.1.1 Bases de dados

Para este trabalho, as bases de dados NSL-KDD (Tavallaee et al. (2009)) e UNSW-NB15 (Moustafa e Slay (2015)) foram escolhidas para realizar a análise proposta. Ambas as bases estão detalhadas na Seção 3.5.

A Tabela 5 apresenta um sumário das distribuições das amostras de cada base, como são encontradas originalmente.

Tabela 5 – Distribuição de amostras das bases de dados

Base	**Amostras de treino**		**Amostras de teste**		**Total de amostras**
	Normais	**Ataques**	**Normais**	**Ataques**	
NSL-KDD	67343	58630	9711	12833	148517
UNSW-NB15	56000	119341	37000	45332	257673

Antes dos dados serem enviados para a etapa de normalização, parte deles são retirados do conjunto, tendo em vista que não são características e não fazem sentido para a classificação. Para a base NSL-KDD, as colunas 42 e 43 que representam a classe e o nível de dificuldade foram removidas. Para a base UNSW-NB15, as colunas 1, 44 e 45 que representam o id da amostra, o tipo de ataque e a classificação da amostra foram removidas.

Desta forma, o novo conjunto de dados formado contempla apenas as características de navegação de cada base de dados, totalizando 41 para a base NSL-KDD e 42 para a base UNSW-NB15. Posteriormente, as amostras seguem para a etapa de normalização.

4.1.2 Normalização dos dados

Os dados provindos das bases selecionadas possuem a mesma formatação, contendo dados do tipo binário, numérico e nominal. Para aplicar a metodologia proposta de forma adequada, as bases passam por um processo de normalização, separado em duas partes.

A primeira parte consiste em atribuir um valor numérico para as variáveis nominais. Sendo assim, é feita uma contagem da quantidade de palavras diferentes em cada característica e cada uma delas é transformada em um número de forma sequencial.

Portanto, as características *proto-type* (2), *service* (3) e *flag* (4) da base de dados NSL-KDD sofrem modificação, assim como as características *proto* (2), *service* (3) e *state* (4) da base de dados UNSW-NB15.

A segunda parte consiste em normalizar todos os valores para a mesma base. Desta forma, é realizada a aplicação da normalização min-max, explicada na Seção 3.3.3, para limitar os intervalos das variáveis de [0, 1].

Ao fim desta etapa, as características estão prontas para inciar o processamento.

4.2 PROCESSAMENTO DAS CARACTERÍSTICAS

A segunda etapa da metodologia proposta é responsável por processar as caracteríistiscas que serão utilizadas como parâmetro de entrada do classificador, representado pela etapa 3. O processamento é composto pela unificação das características, pela seleção de características com algoritmos bio-inspirados e pela individualização das características. Para cada um destes casos são utilizados os mesmos dados recebidos da etapa anterior, considerando as duas bases normalizadas.

4.2.1 Unificação das características

Esta parte do processamento consiste em agrupar todas as características de cada base para serem classificadas na etapa seguinte. Assim, todo o conjunto é utilizado como dado de entrada do classificador e é possível observar os resultados obtidos após realizar somente a normalização dos dados.

Deste forma, as 41 características da base de dados NSL-KDD e as 42 características da base de dados UNSW-NB15 ficam agrupadas para a etapa de classificação.

4.2.2 Seleção de características com algoritmos bio-inspirados

A seleção de características pode ser considerado um problema de otimização, tendo em vista que o seu objetivo é diminuir a quantidade de características e aumentar ou diminuir algum parâmetro preestabelecido. Desta forma, com a finalidade de potencializar a performance

do classificador, as principais características das bases de dados serão selecionadas com o uso de algoritmos bio-inspirados.

Como demonstrado por Rostami, Berahmand e Forouzandeh (2020), os algoritmos bio-inspirados do tipo inteligência de enxame se destacam para a função de seleção de características. Então, cinco algoritmos bio-inspirados do tipo inteligencia de enxame (Molina et al. (2020)) são propostos para esta avaliação. Estes algoritmos estão detalhados na Seção 3.2 e são listado abaixo.

a) *Ant Lion Optimizer (ALO)*;
b) *Cuckoo Search (CS)*;
c) *Grey Wolf Optimizer (GWO)*;
d) *Krill Herd* (KH); e
e) *Whale Optimization Algorithm* (WOA).

Como estes algoritmos de otimização possuem a finalidade de aumentar a performance do classificador, a função objetivo deve estar diretamente ligada ao resultado obtido em uma classificação. Sendo assim, o próprio classificador que será especificado na seção seguinte foi utilizado para extrair os parâmetros da função objetivo, descrita pela Equação 45. A função objetivo empregada neste trabalho visa a minimização do erro e da quantidade de características utilizadas na classificação.

$$\text{Função objetivo} = w_1 * (1 - acc) + w_2 * \frac{CS}{CT} \tag{45}$$

Onde acc é a acurácia obtida no classificador, CS é o número de características selecionadas e CT é o número total de características da base. Os parâmetros w_1 e w_2 são os pesos atribuídos para as variáveis da função objetivo. A acurácia é considerado o fator de maior importância para esta otimização e, por isso, o valor adotado de w_1 é 0,95. Consequentemente, o valor de w_2 é 0,05.

Por fim, a Tabela 6 mostra os parâmetros iniciais em comum de todos os algortimos bio-inspirados citados, enquanto as Tabelas 9, 7 e 8 mostram os parâmetros específicos para os algoritmos WOA, CS e KH, respectivamente.

Tabela 6 – Lista de parâmetros iniciais comuns dos algoritmos bio-inspirados

Parâmetro	**Valor**
Limite superiror	1
Limite inferior	0
Limiar de decisão	0,5
Agentes de busca	10
Dimensões	Número de Características Utilizadas da Base
Número de Iterações	100
Função objetivo	Equação 45

Tabela 7 – Lista de parâmetros iniciais do algoritmo CS

Parâmetro	**Valor**
Pa	0,25
alpha	1
beta	1,5

Tabela 8 – Lista de parâmetros iniciais do algoritmo KH

Parâmetro	**Valor**
Vf	0,02
Dmax	0,005
Nmax	0,01
Sr	0

Tabela 9 – Lista de parâmetros iniciais do algoritmo WOA

Parâmetro	**Valor**
b	1

4.2.3 Individualização das características

Por fim, nesta etapa do processamento é feita a separação das características que serão utilizadas como parâmetro de entrada do classificador individualmente. Sendo assim, é possível verificar a infuência de cada uma das características na detecção de ataques com o classificador escolhido.

Portanto, a base de dados NSL-KDD é separada em 41 subgrupos e a base de dados UNSW-NB15 é separada em 42 subgrupos, contendo apenas uma característica em cada um deles.

4.3 CLASSIFICAÇÃO

A terceira etapa da metodologia proposta consiste na classificação dos dados recebidos da etapa de processamento. Para esta classificação foi escolhido o algoritmo k-vizinhos mais próximos, que está detalhado na Seção 3.3.1.

Para este trabalho, o número de K considerado é dado pela Equação 46.

$$K = \sqrt{N} \tag{46}$$

onde N é o número de amostras a serem classificadas. Para evitar empates na decisão da classe da amostra a ser classificada, o valor de K deve ser arredondado para o número ímpar mais próximo do resultado obtido na equação.

Além disso, para evitar que ocoram sobreajuste (*overfitting*) com o modelo proposto, é utilizada a validação cruzada (*k-fold*), com $K = 10$. Este conceito está descrito na Seção 3.3.2

4.4 ANÁLISE DOS RESULTADOS

A quarta e última etapa da metodologia proposta consiste na análise e comparação dos resultados obtidos na etapa de classificação. Para cada conjunto de dados classificados será gerado um grupo de métricas. Para utilizar estas métricas, foi feita uma binarização do tipo de amostra, onde uma amostra de ataque é interpretada como "1"e uma amostra normal é interpretada como "0". As métricas escolhidas estão detalhadas na Seção 3.4 e listadas abaixo.

a) Acurácia;
b) *F1-Score*;
c) Precisão;
d) Revocação;
e) Taxa de falsos positivos (FPR);
f) Taxa de falsos negativos (FNR); e
g) Taxa de verdadeiros negativos (TNR).

Os resultados que mais se destacam são comparados com os principais trabalhos do estado da arte na detecção de intrusão em redes.

5 RESULTADOS E DISCUSSÃO

Neste capítulo serão apresentados os resultados obtidos a partir da aplicação da metodologia proposta no Capítulo 4, além da discussão dos pontos importantes que foram observados na detecção de ataques.

Inicialmente, será demonstrado o resultado obtido com a aplicação do KNN como classificador utilizando todas as características que compõem as bases de dados NSL-KDD (Tabela 2) e UNSW-NB15 (Tabela 4). Em seguida, será apresentado o desempenho do classificador utilizando apenas as características selecionadas por meio dos algoritmos bio-inspirados propostos.

Em seguida, os resultados obtidos serão comparados com os trabalhos relacionados do estado da arte para verificar a eficiência do método empregado. Por fim, cada característica extraída será analisada individualmente para determinar a sua influência no processo de detecção de ataques.

5.1 CLASSIFICAÇÃO DE ATAQUES COM KNN

A técnica empregada para atuar como classificador do modelo do IDS proposto foi o KNN (Seção 3.3.1). Para analisar o desempenho do classificador selecionado, antes da etapa de seleção das características mais relevantes para o modelo, foi realizado o treinamento das bases de dados na íntegra, utilizando todas características disponíveis em cada uma delas.

5.1.1 KNN na Base de Dados NSL-KDD

A base de dados NSL-KDD (Tabela 2) possui 41 características de tráfego na rede. Para a análise incial, todas as características foram utilizadas no classificador selecionado.

O número total de amostras desta base é 148.517, portanto, o K utilizado no KNN para classificação dos dados é $K = 385$. A Tabela 10 mostra as métricas obtidas na análise proposta aqui.

Tabela 10 – Métricas do classificador KNN na base de dados NSL-KDD

Acurácia	F1 Score	sensibilidade	Precisão	FPR	FNR	TNR
95,72%	95,87%	96,58%	95,12%	5,18%	3,42%	94,82%

5.1.2 KNN na Base de Dados UNSW-NB15

A base de dados UNSW-NB15 (Tabela 4) possui 42 características de fluxo de navegação do usuário na rede. Assim como na base de dados NSL-KDD, todas as características foram utilizadas na primeira análise do classificador.

Esse subconjunto da base de dados possui 257.673 amostras. Desta forma, o K utilizado no KNN para classificação dos dados é $K = 507$. A Tabela 11 mostra as métricas obtidas na análise apresentada. Nota-se que, comparado à base anterior, o classificador obteve desempenho inferior em todas as métricas de avaliação, indicando a priori um grau de dificuldade maior na classificação de ataques.

Tabela 11 – Métricas do classificador KNN na base de dados UNSW-NB15

Acurácia	F1 Score	sensibilidade	Precisão	FPR	FNR	TNR
84,14%	88,02%	85,20%	90,99%	18,13%	14,80%	81,87%

5.2 ALGORITMOS BIO-INSPIRADOS PARA SELEÇÃO DE CARACTERÍSTICAS

Os algoritmos bio-inspirados escolhidos foram utilizados para otimizar o desempenho do classificador com base na seleção de características, além de auxiliar na identificação das características que mais influenciam na detecção de ataques em redes, haja vista que parte dos dados são redundantes ou irrelevantes para a classificação.

Os parâmetros iniciais básicos dos algoritmos utilizados estão descritos na Tabela 6.

5.2.1 Seleção de Características na Base de Dados NSL-KDD

Após a execução dos algoritmos propostos para seleção de características da base de dados NSL-KDD, foi possível observar que todos eles reduziram ao menos 50% a quantidade de características que foram utilizadas no classificador KNN.

A Tabela 12 apresenta as características selecionadas por cada um dos algoritmos bio-inpirados que otimizam o desempenho do classificador. O algoritmo que utilizou o maior número de características foi o CS, selecionando 19 entre elas, enquanto o WOA selecionou o menor número de características com um total de 5.

Das características selecionadas, a única que aparece em todos os cenários é a 3, que se refere ao serviço utilizado na conexão conforme é demonstrado na Tabela 2.

Tabela 12 – Características Selecionadas na base NSL-KDD

Método	Características Selecionadas	Total
ALO-KNN	3, 8, 10, 21, 26, 27, 29, 34, 36, 39, 40, 41	12
GWO-KNN	3, 10, 12, 25, 27, 28, 29, 34, 36, 39, 40	11
KH-KNN	3, 5, 6, 7, 8, 12, 13, 15, 17, 19, 22, 27, 31, 34, 37, 38, 39	17
CS-KNN	2, 3, 7, 9, 11, 12, 13, 14, 15, 17, 21, 25, 26, 28, 29, 35, 36, 40, 41	19
WOA-KNN	3, 5, 6, 35, 36	5

Com base nas características selecionadas, utilizou-se novamente o KNN como classificador. A Tabela 13 demonstra as métricas obtidas para cada um dos cenários, sendo possível

observar que todas as métricas superam a performance da primeira análise que utilizou as 41 características para classificação.

Dentre as características selecionadas, o GWO obteve a maior acurácia com 97,82% e maior F1 Score com 97,94%, selecionando 11 das 41 características do banco de dados. O CS obteve a maior precisão com 97,63%, menor taxa de falsos alarmes (FPR) com 2,55% e maior TNR com 97,45%, selecionando 19 características. Já o WOA obteve a maior sensibilidade com 98,30% e a menor quantidade de características utilizadas com 5 de 41.

Apesar da quantidade de características utilizadas ter variado consideravelmente em cada método, o resultado obtido foi convergente. Portanto, para o objetivo de detectar ataques, o GWO se mostrou mais eficiente, pois obteve a maior acurácia e utilizou apenas 11 das 41 características da base NSL-KDD. No entanto, todos os demais algoritmos obtiveram resultados similares, mas o WOA-KNN foi o que obteve o resultado similar com o menor número de características, destacando-se de um modo geral.

Tabela 13 – Métricas Obtidas das Características Selecionadas na base NSL-KDD

Método	Acc	F1	Rev.	Prec.	FPR	FNR	TNR
ALO-KNN	97,64%	97,69%	98,20%	97,24%	2,95%	1,80%	97,05%
GWO-KNN	**97,82%**	**97,94%**	98,24%	97,54%	2,63%	1,76%	97,37%
KH-KNN	97,38%	97,48%	98,27%	96,65%	3,54%	1,73%	96,46%
CS-KNN	97,60%	97,66%	97,74%	**97,63%**	**2,55%**	2,26%	**97,45%**
WOA-KNN	97,32%	97,43%	**98,30%**	96,50%	3,70%	**1,70%**	96,30%
KNN	95,72%	95,87%	96,58%	95,12%	5,18%	3,42%	94,82%

5.2.2 Seleção de Características na Base de Dados UNSW-NB15

Assim como para a base de dados NSL-KDD, os algortimos bio-inpirados utilizados reduziram ao menos em 50% a quantidade de características selecionadas para uso no classificador KNN. A Tabela 14 mostra as características selecionadas pelos algoritmos propostos para a otimização do desempenho do classificador.

Para a base UNSW-NB15, o menor número de características selecionadas foi obtido com o algortimo ALO com um total de 11 características, enquanto o KH selecionou o maior número de características com 18. Analisando as características em comum em cada uma das técnicas aplicadas, a única que se repete em todos os casos é a 11, que se refere ao parâmetro *time to live* (TTL) entre destino e fonte, de acordo com a Tabela 4.

Seguindo a mesma metodologia, as características selecionadas com os algoritmos bio-inspirados foram utilizadas no classificador para obtenção das novas métricas. A Tabela 15 mostra os valores obtidos para cada um dos casos e, assim como na base de dados NSL-KDD, todos os cenários apresentam melhorias com relação ao uso do classificador sem a seleção de características.

Tabela 14 – Características Selecionadas na base UNSW-NB15

Método	Características Selecionadas	Total
ALO-KNN	2,3,11,15,23,26,29,32,33,36,38	11
GWO-KNN	1,2,10,11,15,20,31,32,34,35,36,38	12
KH-KNN	4,6,7,10,11,14,16,17,20,23,27,28,30,35,37,38,41,42	18
CS-KNN	2,3,10,11,15,20,24,26,27,28,29,35,36,41	14
WOA-KNN	2,3,5,6,10,11,14,23,27,28,29,30,35,36,41,42	16

Utilizado o algoritmo GWO, obteve-se a maior acurácia com 93,39% e o maior F1 Score com 94,93%, selecionando 12 características. O maior valor de sensibilidade e a menor taxa de falsos negativos foram alcançados com o algoritmo CS com 94,29% e 5,71%, respectivamente, utilizando 14 das 42 características. Por fim, por meio do algoritmo ALO, foi possivel encontrar a maior precisão com 97,09% e a menor quantidade de características selecionadas com 11.

Sendo assim, analisando os resultados obtidos com o mesmo critério de detecção eficiente de ataques, o melhor método foi o GWO novamente com o maior valor de acurácia utilizando apenas 12 das 42 características da base de dados UNSW-NB15.

Tabela 15 – Métricas Obtidas das Características Selecionadas na base UNSW-NB15

Método	Acc	F1	Rev.	Prec.	FPR	FNR	TNR
ALO-KNN	92,57%	94,37%	91,76%	**97,09%**	**5,75%**	8,24%	94,25%
GWO-KNN	**93,39%**	**94,93%**	93,40%	96,47%	6,63%	6,60%	93,37%
KH-KNN	90,71%	92,79%	92,08%	93,50%	11,83%	7,92%	88,17%
CS-KNN	92,91%	94,45%	**94,29%**	94,64%	9,55%	**5,71%**	90,45%
WOA-KNN	92,65%	94,29%	94,28%	94,21%	10,23%	5,72%	89,77%
KNN	84,14%	88,02%	85,20%	90,99%	18,13%	14,80%	81,87%

5.3 COMPARAÇÃO COM OS TRABALHOS RELACIONADOS

Para comprovar a eficiência do método proposto, foi feita uma comparação dos algoritmos implementados com os principais trabalhos relacionados do estado da arte. As tabelas de comparações foram preenchidas com as mesmas métricas utilizadas neste trabalho.

5.3.1 Comparação na Base de Dados NSL-KDD

Por se tratar de uma base de dados popular no meio científico, diversos trabalhos a utilizaram para validar modelos de IDS. Desta forma, existem trabalhos relevantes que podem ser comparados com a metodologia proposta neste projeto. A Tabela 17 mostra as métricas obtidas comparadas com os melhores trabalhos do estado da arte. Os valores das métricas que compõem esta tabela foram extraídos de cada um dos trabalhos mencionados, conforme mostra a Tabela 16.

Ao analisar estes trabalhos, é possível verificar que a metodologia proposta supera a maior parte deles em todas as métricas de avaliação. Embora os trabalhos de Alsaleh e Binsaeedan (2021) com o método SSA-XGBoost e de Sarvari et al. (2020) com o método MCF-ANN permaneçam com os melhores resultados para esta base de dados, o trabalho proposto utilizou o menor número de características para detecção de ataque, o que diminui a quantidade de dados que entra no classificador e consequentemente o tempo de execução.

Tabela 16 – Referências dos trabalhos comparados na base de dados NSL-KDD

Método	**Referência**
GA-NB	Mahmood e Hameed (2016)
IG-NB	Mahmood e Hameed (2016)
BPSO-NB	Najeeb e Dhannoon (2018)
BFA-NB	Najeeb e Dhannoon (2018)
GWO-SVM	Safaldin, Otair e Abualigah (2021)
SIG-PIO-DT	Alazzam, Sharieh e Sabri (2020)
COS-PIO-DT	Alazzam, Sharieh e Sabri (2020)
SSA-XGBoost	Alsaleh e Binsaeedan (2021)
SSA-NB	Alsaleh e Binsaeedan (2021)
MCF-ANN	Sarvari et al. (2020)
DT-DDE	Popoola e Adewumi (2017)

Tabela 17 – Comparação de resultados na base de dados NSL-KDD

Método	**Acc**	**F1**	**Rev.**	**Prec.**	**FPR**	**FNR**	**TNR**	**#CS**
ALO-KNN	97,64%	97,69%	98,20%	97,24%	2,95%	1,80%	97,05%	12
GWO-KNN	97,82%	97,94%	98,24%	97,54%	2,63%	1,76%	97,37%	11
KH-KNN	97,38%	97,48%	98,27%	96,65%	3,54%	1,73%	96,46%	17
CS-KNN	97,60%	97,66%	97,74%	97,63%	2,55%	2,26%	97,45%	19
WOA-KNN	97,32%	97,43%	98,30%	96,50%	3,70%	1,70%	96,30%	5
GA-NB	97,51%	N/A	97,63%	N/A	2,60%	N/A	N/A	10
IG-NB	97,12%	N/A	96,19%	N/A	2,07%	N/A	N/A	39
BPSO-NB	90,63%	N/A	N/A	N/A	9,37%	N/A	N/A	22
BFA-NB	92,20%	N/A	N/A	N/A	7,98%	N/A	N/A	14
GWO-SVM	96,00%	N/A	96,00%	N/A	3,00%	N/A	N/A	12
SIG-PIO-DT	86,90%	86,40%	81,70%	N/A	6,40%	N/A	N/A	18
COS-PIO-DT	88,30%	88,20%	86,60%	N/A	8,80%	N/A	N/A	5
SSA-XGBoost	99,00%	99,00%	99,10%	98,00%	2,10%	0,60%	98,00%	20
SSA-NB	94,00%	94,00%	97,20%	90,50%	8,40%	4,20%	91,50%	13
MCF-ANN	98,16%	N/A	96,83%	N/A	2,90%	N/A	N/A	22
DT-DDE	88,73%	89,00%	89,00%	90,00%	9,30%	N/A	N/A	16

#CS = Número de características selecionadas

5.3.2 Comparação na Base de Dados UNSW-NB15

Assim como a base de dados NSL-KDD, a base UNSW-NB15 é popular no meio científico para validação de IDS. Contudo, quando comparada com a base NSL-KDD, os trabalhos possuem métricas inferiores, demonstrando assim a maior complexidade da base UNSW-NB15 na detecção de ataques.

A Tabela 19 mostra uma comparação das métricas dos melhores trabalhos que utilizam esta base de dados. Os valores das métricas que compõem esta tabela foram extraídos de cada um dos trabalhos mencionados, conforme mostra a Tabela 18. Nota-se que o desempenho do método proposto neste projeto novamente supera os outros trabalhos em praticamente todos os aspectos.

Por meio da técnica GWO-KNN, que apresentou os melhores resultados como mostrado na Seção 5.2.2, obteve-se a melhor acurácia e F1 score, com apenas 12 características, o que supera todos os outros trabalhos do estado da arte.

Tabela 18 – Referências dos trabalhos comparados na base de dados UNSW-NB15

Método	**Referência**
SIG-PIO-DT	Alazzam, Sharieh e Sabri (2020)
COS-PIO-DT	Alazzam, Sharieh e Sabri (2020)
SSA-XGBoost	Alsaleh e Binsaeedan (2021)
SSA-NB	Alsaleh e Binsaeedan (2021)
J48-GA	Almomani (2020)
J48-GWO	Almomani (2020)
J48-FFA	Almomani (2020)
SVM-PSO	Almomani (2020)
SVM-GA	Almomani (2020)

5.4 ANÁLISE DAS CARACTERÍSTICAS

Com a finalidade de verificar a influência de cada característica na identificação de um ataque em uma rede, cada uma delas foi utilizada como entrada do classificador KNN individualmente para obtenção das métricas de avaliação. Este processo foi realizado para todas as características de cada base de dados, independente se a característica foi selecionada ou não, quando utilizados os algoritmos bio-inspirados. A métrica escolhida para esta análise individual é a acurácia.

5.4.1 Análise das Características da Base de Dados NSL-KDD

Após utilizar o classificador KNN para cada uma das 41 características da base NSL-KDD de forma individual, obteve-se o valor de acurácia de cada uma delas demonstrado na

Tabela 19 – Comparação de resultados na base de dados UNSW-NB15

Método	Acc	F1	Rev.	Prec.	FPR	FNR	TNR	#CS
ALO-KNN	92,57%	94,37%	91,76%	97,09%	5,75%	8,24%	94,25%	11
GWO-KNN	93,39%	94,93%	93,40%	96,47%	6,63%	6,60%	93,37%	12
KH-KNN	90,71%	92,79%	92,08%	93,50%	11,83%	7,92%	88,17%	18
CS-KNN	92,91%	94,45%	94,29%	94,64%	9,55%	5,71%	90,45%	14
WOA-KNN	92,65%	94,29%	94,28%	94,21%	10,23%	5,72%	89,77%	16
SIG-PIO-DT	91,30%	90,40%	89,70%	N/A	5,20%	N/A	N/A	14
COS-PIO-DT	91,70%	90,90%	89,40%	N/A	3,40%	N/A	N/A	5
SSA-XGBoost	93,00%	94,40%	96,00%	93,00%	6,90%	7,70%	93,00%	23
SSA-NB	85,00%	89,40%	99,00%	82,00%	18,00%	3,80%	82,00%	14
J48-GA	86,80%	86,80%	96,70%	78,80%	21,10%	33,00%	78,80%	23
J48-GWO	85,60%	85,40%	93,70%	78,50%	20,90%	62,00%	79,00%	20
J48-FFA	86,00%	86,10%	96,50%	77,70%	22,50%	34,10%	77,40%	21
SVM-PSO	89,10%	87,10%	79,50%	96,30%	25,90%	20,40%	97,40%	25
SVM-GA	86,30%	86,50%	96,90%	78,00%	22,20%	30,30%	77,70%	23

#CS = Número de características selecionadas

forma de gráfico de barras. A Figura 9 apresenta as características de 1 a 21, enquanto a Figura 10 apresenta as características de 22 a 41.

Nota-se que algumas características se destacam quando analisadas individualmente. As características que possuem acurácia acima de 80% foram separadas para análise e compõem a Tabela 20.

A característica que mais se destaca utilizando o classificador KNN é a número 5 com 94% de acurácia, a qual está relacionada à quantidade de bytes de dados transferidos da origem para o destino. Em segundo lugar, encontra-se a característica número 6 com 90% de acurácia que também está relacionada à quantidade de bytes transferidos, porém, neste caso, do destino para a origem. Logo em seguida está a característica número 3 com 89%, que indica o tipo de serviço utilizado na conexão. Depois, com 86% de acurácia, aparece a característica número 4 que se refere à indicação do estado da conexão.

Posteriormente, aparecem as características 30 com 85% de acurácia, indicando o percentual das conexões em diferentes serviços e a característica 29 com 84%, indicando o percentual de conexões nos mesmos serviços. Ainda com 84% de acurácia, obteve-se a característica número 33, que representa o número de conexões com o mesmo host de destino e mesmo serviço e, em seguida, a característica 34, que representa o percentual de conexões com o mesmo host de destino e mesmo serviço com 83%.

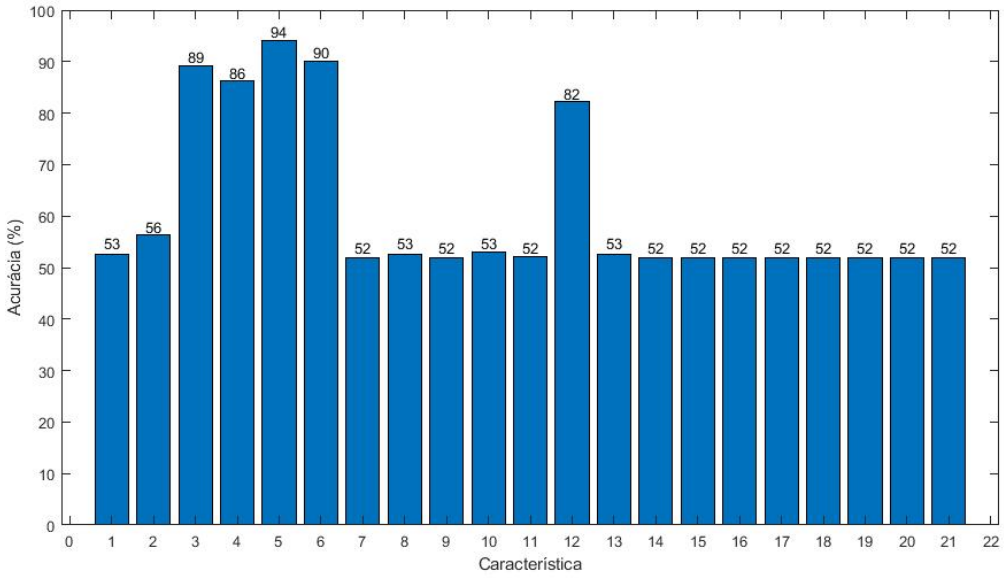
Logo após, encontra-se a característica 12 com 82% de acurácia, que mostra o status do login, se bem-sucedido ou não. Com os mesmos 82%, observa-se a característica 35, que representa o percentual de conexões com serviços diferentes no host atual. Por fim, com 80% de acurária, aparece a característica 23, que representa o número de conexões com o mesmo host.

Por meio dos dados obtidos, é possível analisar que as características que mais influenciam individualmente o classificador KNN na detecção de ataques são aquelas que envolvem (i)

o volume de dados transferidos na rede, representado pelas características 5 e 6; (ii) o tipo de serviço empregado na conexões, representado pelas características 3, 29, 30 e 35; (iii) o estado da conexão, representado pelas características 4, 23, 33 e 34; e (iv) o sucesso ou não do login, representado pela característica 12. Sendo assim, sugere-se o monitoramento contínuo destas características, a fim de mitigar potenciais intrusões.

Apesar destas características apresentarem um valor de acurácia significativo para identificar ataques em redes, o uso de determinadas características em conjunto potencializa a capacidade do classificador em detectar ataques de forma eficiente, como foi mostrado na Seção 5.2 anterior.

Figura 9 – Acurácia de cada característica individualmente parte 1

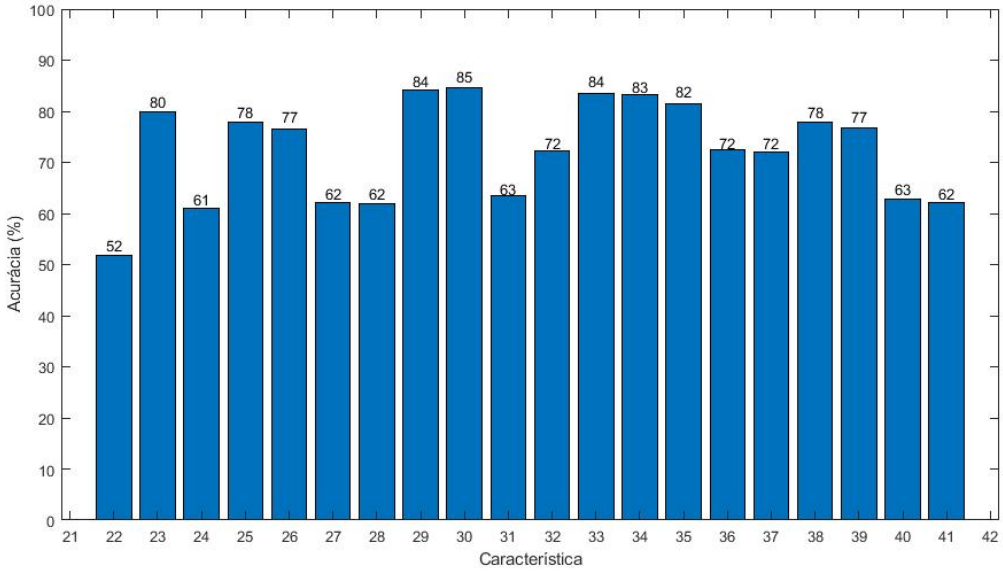


Fonte: Autor.

Tabela 20 – Características da base NSL-KDD com maiores acurácias utilizando KNN

Carac.	**Descrição**	**Acurácia (%)**
5	Bytes de dados transferidos da origem para o destino	94
6	Bytes de dados transferidos do destino para a origem	90
3	Destino do serviço de rede	89
4	Estado da conexão	86
30	Percentual de conexões em diferentes serviços	85
29	Percentual de conexões nos mesmos serviços	84
33	Número de conexões com o mesmo host de destino com o mesmo serviço	84
34	Percentual de conexões com o mesmo host de destino com o mesmo serviço	83
12	Status do login: 1 com sucesso, senão 0	82
35	Percentual de conexões com serviços diferentes no host atual	82
23	Número de conexões com o mesmo host	80

Figura 10 – Acurácia de cada característica individualmente parte 2



Fonte: Autor.

5.4.2 Análise das Características da Base de Dados UNSW-NB15

Assim como foi feito para a base de dados NSL-KDD, as 42 características da base UNSW-NB15 foram utilizadas individualmente como entrada do classificador KNN para obtenção da acurácia de cada uma delas na detecção de ataques em uma rede. O resultado obtido está representado nas Figuras 11 e 12, sendo que a primeira mostra as características de 1 a 21, enquanto a segunda mostra as características 22 a 42.

Nesta seção, foram aplicadas as mesmas estratégias de avaliação utilizadas para a base de dados KSL-KDD nas seções anteriores. Assim as características que possuem acurácia maior ou igual a 80% foram separadas para análise e estão destacadas na Tabela 21. Neste caso, apenas 4 características atingiram este valor, indicando que, para esta base de dados, é mais complexo determinar se uma amostra representa um ataque ou não, analisando apenas uma característica.

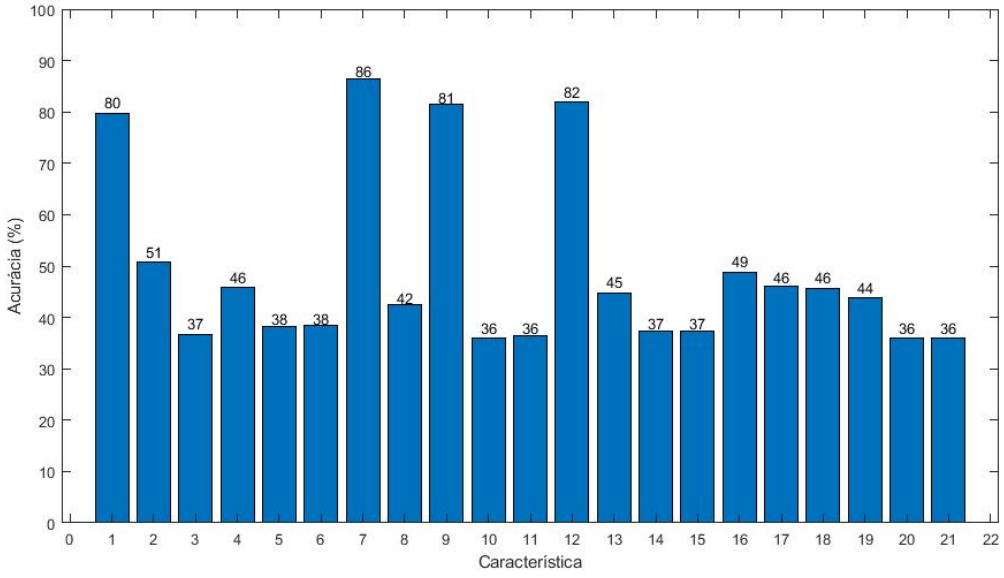
Esta complexidade reflete nas métricas obtidas nos tópicos anteriores, que se mostraram inferiores em todos os aspectos quando comparadas com a base de dados NSL-KDD.

Tabela 21 – Características da base UNSW-NB15 com maiores acurácias utilizando KNN

Característica	**Descrição**	**Acurácia (%)**
7	Bytes de dados transferidos da origem para o destino	86
12	Bits de origem	82
9	Taxa de conexão	81
1	Duração total	80

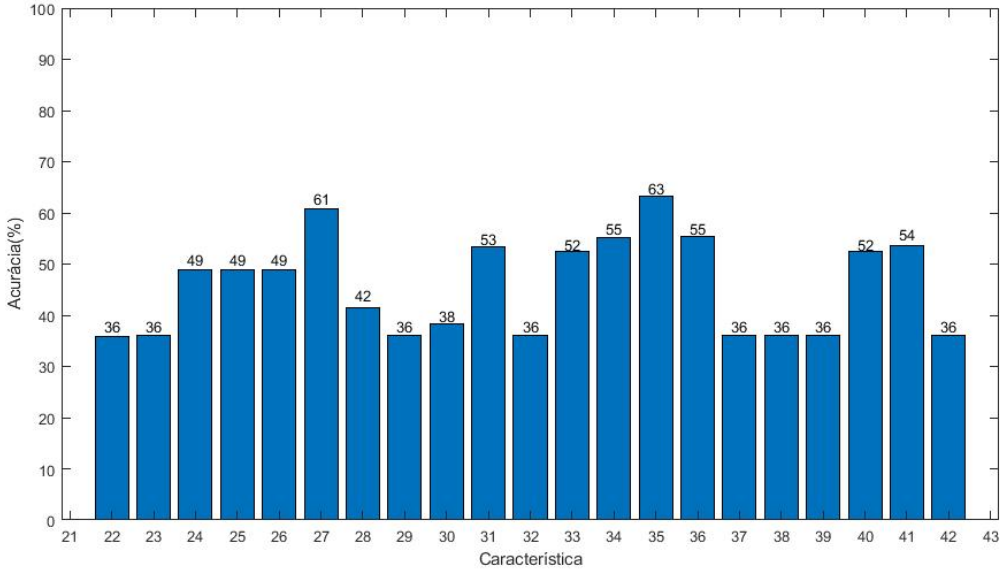
A característica que mais se destacou nesta etapa foi a número 7, com 86% de acurácia, que se refere aos bytes de dados transferidos da origem para o destino. Em seguida, nota-se a

Figura 11 – Acurácia de cada característica individualmente parte 1



Fonte: Autor.

Figura 12 – Acurácia de cada característica individualmente parte 2



Fonte: Autor.

característica 12 com 82%, que representa os bits da origem. Em seguida, com 81% aparece a característica 9, que representa a taxa de conexão e, por fim, a duração total da conexão com 80% de acurácia.

Para esta base de dados em análise, nota-se que mais uma vez o volume de dados na rede tem um papel importante na detecção de ataques. Além disso, as características relacionadas à conexão também se destacam quando comparadas as demais. Contudo, é possível observar que a classificação de dados nesta base é mais complexa do que na NSL-KDD, sendo o conjunto de

características específicas utilizadas como entrada do classificador KNN um importante potencializador na capacidade de detecção de ataques, vide os resultados obtidos no tópico anterior.

6 CONCLUSÃO

Os sistemas de detecção de intrusão (IDS) são reconhecidos como uma ferramenta robusta para proteger redes de diversos tipos de ataques. Neste trabalho foram propostos modelos de IDS para detecção de ataques em redes, além da análise individual das características que compõem as bases de dados para determinar quais delas estão mais relacionadas a um ataque, objetivando mitigar intrusões.

Para isso, foi utilizado o algoritmo KNN como classificador das características das bases de dados NSL-KDD e UNSW-NB15. Em seguida, foram propostos cinco algoritmos bio-inspirados para selecionar as características que potencializavam os resultados do classificador proposto, sendo eles GWO, WOA, KH, CS e ALO. Ao final, cada característica foi utilizada individualmente como parâmetro de entrada do classificador para verificar a sua influência em uma intrusão na rede.

Para a avaliação dos modelos propostos, foram utilizadas as métricas: acurácia, *F1-score*, sensibilidade, precisão, taxa de falsos positivos (FPR), taxa de falsos negativos (FNR) e taxa de verdadeiros negativos (TNR).

Os modelos propostos de IDS obtiveram resultados que se equiparam ou superam os principais trabalhos do estado da arte com aumento de acurácia, aumento de taxa de detecção de ataques (sensibilidade) e redução de taxa de falsos positivos. Apesar dos resultados serem semelhantes entre os bio-inspirados propostos para as bases para a NSL-KDD e UNSW-NB15, o GWO obteve os melhores valores nas métricas de avaliação em ambas as bases, enquanto o WOA se destaca por selecionar apenas 5 características na base NSL-KDD e, ainda assim, obter alta performance nas métricas de avaliação.

Além disso, quando analisadas individualmente, as características que representam o volume de dados na rede, o tipo de serviço empregado nas conexões, o estado da conexão e sucesso ou não do login são as que mais influenciam o classificador proposto na detecção de ataques na base de dados NSL-KDD. Para a base de dados UNSW-NB15, as características de volume de dados na rede e as características relacionadas à conexão são as que mais tiveram influência no classificador KNN, contudo, esta base de dados se mostrou mais complexa e depende de uma combinação de outras características para detectar eficientemente um ataque.

Assim, foi possível identificar as características que mais influenciam individualmente na detecção de ataques ao utilizar o classificador KNN para ambas as bases de dados propostas.

O modelo de IDS proposto que utiliza os algoritmos bio-inspirados para selecionar as principais características para detecção de ataques, juntamente com o uso do KNN como classificador, se mostrou competitivo com os principais trabalhos do estado da arte, de forma que os resultados obtidos foram equivalentes ou superiores.

Para trabalhos futuros, outras bases de dados podem ser utilizadas com a mesma metodologia para validação e comparação com os resultados apresentados, assim como outros algoritmos de classificação também podem ser utilizados, juntamente com os algoritmos bio-inpirados

propostos, a fim de comparar o desempenho entre eles. Por fim, é possível utilizar a mesma técnica proposta neste trabalho para avaliar as amostras das bases de dados com relação ao tipo de ataque apresentado e não somente separá-las entre ataque e não ataque.

REFERÊNCIAS

ABUALIGAH, Laith Mohammad. Feature Selection and Enhanced Krill Herd Algorithm for Text Document Clustering. In: STUDIES in Computational Intelligence. [S.l.: s.n.], 2018. Disponível em: <https://api.semanticscholar.org/CorpusID:56594355>.

ALAZZAM, Hadeel; SHARIEH, Ahmad Abdel-Aziz; SABRI, Khair Eddin. A feature selection algorithm for intrusion detection system based on Pigeon Inspired Optimizer. **Expert Syst. Appl.**, v. 148, p. 113249, 2020.

ALMOMANI, Omar. A Feature Selection Model for Network Intrusion Detection System Based on PSO, GWO, FFA and GA Algorithms. **Symmetry**, v. 12, p. 1046, 2020.

ALSALEH, Alanoud; BINSAEEDAN, Wojdan. The Influence of Salp Swarm Algorithm-Based Feature Selection on Network Anomaly Intrusion Detection. **IEEE Access**, v. 9, p. 112466–112477, 2021.

CHORAŚ, Michał; PAWLICKI, Marek. Intrusion detection approach based on optimised artificial neural network. **Neurocomputing**, v. 452, p. 705–715, 2021.

COVER, Thomas M.; HART, Peter E. Nearest neighbor pattern classification. **IEEE Trans. Inf. Theory**, v. 13, p. 21–27, 1967. Disponível em: <https://api.semanticscholar.org/CorpusID:5246200>.

DIAO, Ren; SHEN, Qiang. Nature inspired feature selection meta-heuristics. **Artificial Intelligence Review**, v. 44, p. 311–340, 2015.

DRENG, Shigeo Abe. Pattern Classification. In: SPRINGER London. [S.l.: s.n.], 2001.

FERRAG, Mohamed Amine et al. Deep Learning-Based Intrusion Detection for Distributed Denial of Service Attack in Agriculture 4.0. **Electronics**, v. 10, p. 1257, 2021.

GANDOMI, Amir Hossein; ALAVI, Amir Hossein. Krill herd: A new bio-inspired optimization algorithm. **Communications in Nonlinear Science and Numerical Simulation**, v. 17, p. 4831–4845, 2012.

GHARAEE, Hossein; HOSSEINVAND, Hamid. A new feature selection IDS based on genetic algorithm and SVM. **2016 8th International Symposium on Telecommunications (IST)**, p. 139–144, 2016. Disponível em: <https://api.semanticscholar.org/CorpusID:22722233>.

JIANG, Kaiyuan et al. Network Intrusion Detection Combined Hybrid Sampling With Deep Hierarchical Network. **IEEE Access**, v. 8, p. 32464–32476, 2020.

KOHAVI, Ron. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In: INTERNATIONAL Joint Conference on Artificial Intelligence. [S.l.: s.n.], 1995. Disponível em: <https://api.semanticscholar.org/CorpusID:2702042>.

LIAO, Hung-Jen et al. Intrusion detection system: A comprehensive review. **J. Netw. Comput. Appl.**, v. 36, p. 16–24, 2013.

MAHMOOD, Dhuha I.; HAMEED, Sarab M. A Feature Selection Model based on Genetic Algorithm for Intrusion Detection. In.

MIRJALILI, Seyed Mohammad. The Ant Lion Optimizer. **Adv. Eng. Softw.**, v. 83, p. 80–98, 2015.

MIRJALILI, Seyed Mohammad; JANGIR, Pradeep; SAREMI, Shahrzad. Multi-objective ant lion optimizer: a multi-objective optimization algorithm for solving engineering problems. **Applied Intelligence**, v. 46, p. 79–95, 2016. Disponível em: <https://api.semanticscholar.org/CorpusID:24194528>.

MIRJALILI, Seyed Mohammad; LEWIS, Andrew. The Whale Optimization Algorithm. **Adv. Eng. Softw.**, v. 95, p. 51–67, 2016.

MIRJALILI, Seyed Mohammad; MIRJALILI, Seyed Mohammad; LEWIS, Andrew. Grey Wolf Optimizer. **Adv. Eng. Softw.**, v. 69, p. 46–61, 2014. Disponível em: <https://api.semanticscholar.org/CorpusID:15532140>.

MOLINA, Daniel et al. Comprehensive Taxonomies of Nature- and Bio-inspired Optimization: Inspiration Versus Algorithmic Behavior, Critical Analysis Recommendations. **Cognitive Computation**, p. 1–43, 2020.

MOUSTAFA, Nour; SLAY, Jill. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). **2015 Military Communications and Information Systems Conference (MilCIS)**, p. 1–6, 2015.

NADIMI-SHAHRAKI, Mohammad-Hossein; TAGHIAN, Shokooh; MIRJALILI, Seyed Mohammad. An improved grey wolf optimizer for solving engineering problems. **Expert Syst. Appl.**, v. 166, p. 113917, 2021. Disponível em: <https://api.semanticscholar.org/CorpusID:224889026>.

NAJEEB, Rana F.; DHANNOON, Ban N. A FEATURE SELECTION APPROACH USING BINARY FIREFLY ALGORITHM FOR NETWORK INTRUSION DETECTION SYSTEM. In.

NANCY, P et al. Intrusion detection using dynamic feature selection and fuzzy temporal decision tree classification for wireless sensor networks. **IET Commun.**, v. 14, p. 888–895, 2020.

NASERI, Touraj Sattari; GHAREHCHOPOGH, Farhad Soleimanian. A Feature Selection Based on the Farmland Fertility Algorithm for Improved Intrusion Detection Systems. **Journal of Network and Systems Management**, v. 30, 2022. Disponível em: <https://api.semanticscholar.org/CorpusID:247574702>.

NGUYEN, Minh Tuan; KIM, Kiseon. Genetic convolutional neural network for intrusion detection systems. **Future Gener. Comput. Syst.**, v. 113, p. 418–427, 2020.

PATRO, S. Gopal Krishna; SAHU, Kishore Kumar. Normalization: A Preprocessing Stage. **ArXiv**, abs/1503.06462, 2015.

POPOOLA, E.; ADEWUMI, Aderemi Oluyinka. Efficient Feature Selection Technique for Network Intrusion Detection System Using Discrete Differential Evolution and Decision. **Int. J. Netw. Secur.**, v. 19, p. 660–669, 2017.

PRIYA, R. M. Swarna et al. An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture. **Comput. Commun.**, v. 160, p. 139–149, 2020.

RIYAZ, B.; GANAPATHY, Sannasi. A deep learning approach for effective intrusion detection in wireless networks using CNN. **Soft Computing**, p. 1–14, 2020.

ROSTAMI, Mehrdad; BERAHMAND, Kamal; FOROUZANDEH, Saman. Review of Swarm Intelligence-based Feature Selection Methods. **Eng. Appl. Artif. Intell.**, v. 100, p. 104210, 2020.

SÁ SILVA, Lílian de. **Uma Metodologia Para Detecção De Ataques No Tráfego De Redes Baseada Em Redes Neurais**. 2008. Tese (Doutorado) – Instituto Nacional de Pesquisas Espaciais (INPE).

SAFALDIN, Mukaram; OTAIR, Mohammed; ABUALIGAH, Laith Mohammad. Improved binary gray wolf optimizer and SVM for intrusion detection system in wireless sensor networks. **Journal of Ambient Intelligence and Humanized Computing**, p. 1–18, 2021.

SALO, Fadi; NASSIF, Ali Bou; ESSEX, Aleksander. Dimensionality reduction with IG-PCA and ensemble classifier for network intrusion detection. **Comput. Networks**, v. 148, p. 164–175, 2019.

SARVARI, Samira et al. An Efficient Anomaly Intrusion Detection Method With Feature Selection and Evolutionary Neural Network. **IEEE Access**, v. 8, p. 70651–70663, 2020.

SELVAKUMAR, B; MUNEESWARAN, K. Firefly algorithm based feature selection for network intrusion detection. **Comput. Secur.**, v. 81, p. 148–155, 2019.

TAVALLAEE, Mahbod et al. A detailed analysis of the KDD CUP 99 data set. **2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications**, p. 1–6, 2009.

WANG, Mingjing; CHEN, Huiling. Chaotic multi-swarm whale optimizer boosted support vector machine for medical diagnosis. **Appl. Soft Comput.**, v. 88, p. 105946, 2020. Disponível em: <https://api.semanticscholar.org/CorpusID:212637907>.

YANG, Hongyu; WANG, Fengyan. Wireless Network Intrusion Detection Based on Improved Convolutional Neural Network. **IEEE Access**, v. 7, p. 64366–64374, 2019.

YANG, Xin-She; DEB, Suash. Cuckoo Search via Lévy flights. **2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)**, p. 210–214, 2009. Disponível em: <https://api.semanticscholar.org/CorpusID:206491725>.

_____. Engineering optimisation by cuckoo search. **Int. J. Math. Model. Numer. Optimisation**, v. 1, p. 330–343, 2010. Disponível em: <https://api.semanticscholar.org/CorpusID:34889796>.

ZAVRAK, Sultan; ISKEFIYELI, Murat. Anomaly-Based Intrusion Detection From Network Flow Features Using Variational Autoencoder. **IEEE Access**, v. 8, p. 108346–108358, 2020.