
UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO

JÚLIO CESAR FARIAS DA LUZ

# DETECÇÃO DE TUNELAMENTO DNS BASEADA EM APRENDIZAGEM DE MÁQUINA: UM ESTUDO COMPARATIVO

Recife, Pernambuco

2025

JÚLIO CESAR FARIAS DA LUZ

# DETECÇÃO DE TUNELAMENTO DNS BASEADA EM APRENDIZAGEM DE MÁQUINA: UM ESTUDO COMPARATIVO

Trabalho apresentado ao Curso de Graduação em Engenharia da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Engenharia da Computação.

**Área de Concentração**: Cibersegurança, Machine Learning, Sistemas de Detecção de Intrusão.

**Orientador**: Paulo Freitas de Araújo Filho

Recife, Pernambuco

2025

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Luz, Júlio Cesar Farias da.
Detecção de tunelamento DNS baseada em aprendizagem de máquina: um estudo comparativo / Júlio Cesar Farias da Luz. - Recife, 2025.
39 p. : il., tab.

Orientador(a): Paulo Freitas de Araújo Filho
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Informática, Engenharia da Computação - Bacharelado, 2025.
Inclui referências.

1. Segurança da informação. 2. Redes de computadores. 3. Aprendizagem de máquina. 4. Detecção de intrusão. 5. Tunelamento DNS. 6. DNS. I. Araújo Filho, Paulo Freitas de. (Orientação). II. Título.

000 CDD (22.ed.)

JÚLIO CESAR FARIAS DA LUZ

**DETECÇÃO DE TUNELAMENTO DNS BASEADA EM APRENDIZAGEM DE MÁQUINA: Um estudo comparativo**

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia da Computação do Centro de Informática da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de bacharel em Engenharia da Computação.

Aprovado em: 02/04/2025

**BANCA EXAMINADORA**

_____

Prof. Dr. Paulo Freitas de Araújo Filho (Orientador)

Universidade Federal de Pernambuco

_____

Prof. Dr. Divanilson Rodrigo Campelo (Examinador Interno)

Universidade Federal de Pernambuco

## AGRADECIMENTOS

Agradeço à minha família, especialmente aos meus pais, Vanda e Antônio, pelo amor, apoio, suporte e incentivo em todos os momentos, permitindo que eu chegasse até aqui. Dedico essa conquista principalmente a vocês.

À minha namorada, Marília, por ser minha fonte de paz e alegria, e por compreender minha ausência nos momentos em que me dediquei aos estudos.

Ao meu orientador, aos professores e aos profissionais que me guiaram, pelos valiosos ensinamentos que foram fundamentais para minha evolução e realização deste trabalho.

Aos meus amigos e colegas de curso, pela parceria, companheirismo e apoio mútuo ao longo dessa caminhada. Vocês tornaram essa experiência ainda mais especial.

Por fim, agradeço a todos que, direta ou indiretamente, contribuíram para essa conquista. A vocês, meu sincero muito obrigado.

*"The important thing is not to stop questioning. Curiosity has its own reason for existing. One cannot help but be in awe when one contemplates the mysteries of eternity, of life, of the marvelous structure of reality. It is enough if one tries to comprehend only a little of this mystery"*

- Albert Einstein

## RESUMO

Domain Name System (DNS) é um protocolo fundamental para o funcionamento da Internet e tem como principal função mapear nomes de domínio para endereços IP. No entanto, sua ampla utilização em redes de computadores tem sido explorada por atacantes através da técnica de tunelamento DNS para evadir mecanismos de defesa, como firewalls, e então realizar atividades maliciosas como vazamento de dados, execução de comandos e upload de códigos maliciosos, causando sérios prejuízos financeiros às instituições. Para detectar o uso dessa técnica, sistemas de detecção de intrusão (IDSs, do inglês Intrusion Detection Systems) foram propostos com duas abordagens principais: detecção baseada em assinatura, que oferece alto desempenho de detecção para ataques conhecidos, e detecção baseada em anomalias, capaz de detectar até mesmo comportamentos maliciosos nunca vistos antes. Dentre os métodos de detecção baseada em anomalias, destacam-se os métodos não-supervisionados, que apresentam melhor desempenho na detecção de ataques desconhecidos e não necessitam de dados rotulados, reduzindo custos associados à aquisição de dados. Neste trabalho, realizamos uma análise comparativa de métodos não-supervisionados de aprendizagem de máquina para o desenvolvimento de IDSs baseados em anomalias, com foco em detectar atividades de vazamento de dados através de túneis DNS. Comparamos o desempenho de métodos de detecção baseados nos algoritmos KMeans, Isolation Forest, Autoencoder e SOM-KNN com outras abordagens propostas na literatura. Os resultados dos experimentos mostraram que a abordagem SOM-KNN obteve, em geral, os melhores resultados em termos de desempenho de detecção e tempo de inferência. Além disso, as abordagens de autoencoder e dos trabalhos da literatura também apresentaram resultados próximos e competitivos.



**Palavras-chaves**: sistemas de detecção de intrusão; tunelamento DNS; DNS; aprendizagem de máquina.

## ABSTRACT

Domain Name System (DNS) is a fundamental protocol for the Internet operation, primarily responsible for mapping domain names to IP addresses. However, its widespread use in computer networks has been exploited by attackers through the DNS tunneling technique to evade defense mechanisms, such as firewalls, and carry out malicious activities, including data exfiltration, remote command execution, and the upload of malicious code, causing significant financial losses to institutions. In order to detect the use of this technique, Intrusion Detection Systems (IDSs) have been proposed with two main approaches: signature-based detection, which offers high detection performance for known attacks, and anomaly-based detection, capable of identifying even previously unseen malicious behaviors. Among anomaly-based detection methods, unsupervised methods stand out, as they exhibit better performance in detecting unknown attacks and do not require labeled data, reducing costs associated with data acquisition. In this work, we conducted a comparative analysis of unsupervised machine learning methods for the development of anomaly-based IDSs, focusing on detecting data exfiltration activities through DNS tunnels. We compared the performance of detection methods based on the KMeans, Isolation Forest, Autoencoder, and SOM-KNN algorithms with other approaches proposed in the literature. The experimental results showed that the SOM-KNN approach generally achieved the best results in terms of detection performance and inference time. Additionally, the autoencoder-based and literature approaches also presented close and competitive results.



**Keywords**: intrusion detection systems; DNS tunneling; DNS; machine learning.

## LISTA DE FIGURAS

# LISTA DE TABELAS

## LISTA DE SÍMBOLOS

| | |
|---|---|
| **AUC-ROC** | Area Under Curve - Receiver Operating Characteristics |
| **ANN** | Artificial Neural Networks |
| **DNS** | Domain Name System |
| **FPR** | False Positive Rate |
| **FQDN** | Fully Qualified Domain Name |
| **IDS** | Intrusion Detection System |
| **iForest** | Isolation Forest |
| **KNN** | K Nearest Neighbors |
| **ML** | Machine Learning |
| **SOM** | Self-Organizing Maps |
| **TPR** | True Positive Rate |

## SUMÁRIO

# 1 INTRODUÇÃO

## 1.1 MOTIVAÇÃO

A nossa atual dependência de infraestrutura digital tem aumentado a superfície de ataque para ameaças cibernéticas, que estão se tornando mais frequentes e disruptivas em infraestruturas críticas, serviços públicos e privacidade individual (FORUM, 2025). A insegurança cibernética no Brasil vem piorando nos últimos anos. Dados de até agosto de 2024 mostraram que desde 2020 mais de 50.000 casos de incidentes de cibersegurança ocorreram, incluindo registros de violações de segurança das redes federais e alertas de vulnerabilidades emitidos pelo Centro de Prevenção, Tratamento e Resposta a Incidentes Cibernéticos (CTIR) (CANIATO, 2024).

Nesse contexto, uma das técnicas utilizadas em ataques cibernéticos é a de tunelamento DNS, que explora o Sistema de Nomes de Domíno (DNS, do inglês Domain Name System), um componente essencial da Internet que tem como propósito principal converter nomes de domínios, como "www.ufpe.br", para um endereço IP a fim de transferir pacotes de rede entre dispositivos. Apesar desse propósito principal, cibercriminosos utilizam esse protocolo para ajudar a realizar atividades maliciosas, como vazamento de dados, upload de arquivos maliciosos e execução remota de comandos arbitrários de sistema operacional (WANG et al., 2021).

Cibercriminosos utilizam a técnica de tunelamento DNS para estabelecer um canal de comunicação entre o dispositivo da vítima e um servidor próprio do atacante, a fim de enviar e receber dados, possibilitando ao atacante evadir mecanismos de defesa em suas ações. Dessa forma, atores maliciosos como OilRig (MITRE ATTCK, 2017b), FIN7 (MITRE ATTCK, 2017a), PinkStats (FISHER, 2013), DNSpionage (CISCO, 2019) e xHunt (UNIT42, 2019) já utilizaram tunelamento DNS como parte de seus ataques, vazando dados sensíveis e causando danos financeiros significativos às instituições, incluindo até mesmo organizações governamentais (PAIREDS, 2015) (UNIT42, 2023).

Um dos principais motivos que fazem os atacantes usarem tunelamento DNS é evadir mecanismos de defesa, como firewalls, que têm como configuração padrão permitir a passagem de pacotes com o protocolo DNS para não comprometer o funcionamento de muitas aplicações em uma rede, como a navegação web. Dessa forma, a maioria dos firewalls, exceto os que possuem tecnologia de inspeção de mensagens DNS, são incapazes de fornecer proteção contra

atividades maliciosas envolvendo túneis DNS (AHMED et al., 2019).

Além de firewalls, outros mecanismos de defesa que possuem dificuldades para detectar ataques envolvendo tunelamento DNS são sistemas de detecção de intrusão (IDSs) baseados em assinaturas. Esses sistemas, por terem sua detecção baseada em assinaturas de ataques previamente conhecidos, não são capazes de detectar implementações novas e até então desconhecidas de túneis DNS. Essa desvantagem é bastante relevante considerando as variações utilizadas na técnica de tunelamento DNS em diferentes ataques. Por exemplo, um único grupo de atores maliciosos, OilRig, usou 5 variações diferentes de ferramentas para implementação de tunelamento DNS em menos de 3 anos (UNIT42, 2023).

Para superar limitações de outros mecanismos de defesa, consideramos o uso de IDSs baseados em anomalias, que são capazes de detectar até mesmo ataques nunca vistos antes. Esses IDSs estimam o comportamento normal de sistemas e detectam atividades maliciosas medindo o desvio de novas atividades em relação ao comportamento normal estimado e podem ser construídos baseados em métodos estatísticos, ou com aprendizagem de máquina (machine learning ou ML). Métodos baseados em ML são divididos em técnicas supervisionadas, que requerem treinamento com dados rotulados, semi-supervisionadas, que requerem dados parcialmente rotulados para o treinamento, ou não-supervisionadas, que dispensam a necessidade de qualquer rótulo.

Neste trabalho, exploramos métodos baseados em ML, usando técnicas não-supervisionadas para construir IDSs baseados em anomalias. Métodos de ML foram escolhidos por apresentarem bons resultados na detecção de ataques complexos (ARAUJO-FILHO et al., 2021) e por não exigirem dados rotulados, reduzindo custos de aquisição e melhorando a detecção de ataques desconhecidos (NGUYEN et al., 2020).

## 1.2 OBJETIVO

Este trabalho tem como objetivo fornecer um estudo comparativo entre diferentes técnicas de ML não-supervisionadas (SOM-KNN, KMeans, isolation forest e autoencoders) para o desenvolvimento de um sistemas de detecção de intrusão baseados em anomalias, com foco em detectar atividades de vazamento de dados através de túneis DNS.

## 1.3 CONTRIBUIÇÕES

Este trabalho apresenta uma análise comparativa de técnicas de aprendizado de máquina aplicadas à detecção de tunelamento DNS, contribuindo para o avanço das estratégias de defesa contra esse tipo de ataque. As principais contribuições incluem:

- Comparação de técnicas de detecção, evidenciando as vantagens e desvantagens de cada uma, destacando as mais adequadas para desenvolvimento de IDSs focados na detecção de tunelamento DNS.
- Publicação do artigo "Unsupervised SOM-Based Intrusion Detection System for DNS Tunneling Attacks" (LUZ et al., 2023), que propõe um IDS não-supervisionado para detecção de tunelamento DNS que mostrou ter bom desempenho de detecção e taxas de falsos positivos menores que 0.0001% avaliado em uma rede empresarial real.
- Realização da Oficina "Empowering Intrusion Detection Systems with Machine Learning" no Simpósio Brasileiro de Segurança da Informação 2024 (SBSeg 2024) (ARCOVERDE et al., 2024). Compartilhamos o conhecimento no desenvolvimento de IDSs baseados em anomalia usando ML adquirido durante a realização desse trabalho com a comunidade acadêmica e profissional.

# 2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, discutiremos brevemente conceitos necessários para o desenvolvimento e compreensão do nosso trabalho.

## 2.1 SISTEMA DE NOMES DE DOMÍNIO (DNS)

O DNS é um componente essencial da Internet usado por diversas redes e aplicações diferentes (RIJSWIJK-DEIJ et al., 2016). Segundo (KUROSE; ROSS, 2016), há duas definições para o DNS: (1) uma base de dados distribuída implementada em uma hierarquia de servidores DNS, e (2) o protocolo de camada de aplicação que permite que usuários consultem essa base de dados. O propósito principal do DNS é converter nomes de domínios, que são fáceis de serem lembrados e manipulados por humanos, como "www.ufpe.br", para endereços IP, que são indispensáveis para o envio de pacotes em redes de computadores.

Nomes de domínios são compostos por cadeias de caracteres separadas por pontos ".". Cada uma dessas cadeias é chamada de *label* e pode ter até 63 caracteres. Por exemplo, "www.ufpe.br" possui 3 *labels*. Um nome de domínio completo não possui distinção entre letras maiúsculas e minúsculas e pode ter no máximo 255 caracteres. O DNS armazena dados relacionados a nomes de domínios através de *resource records* (RRs), que estabelecem mapeamentos entre nomes de domínios e tipos específicos de recursos, como endereços IPv4, IPv6, domínio canônico e servidores de email, com os respectivos RRs: A, AAAA, CNAME e MX (MOCKAPETRIS, 1987).

### 2.1.1 Tunelamento DNS

Tunelamento DNS se trata de uma técnica usada para transmitir dados entre dois dispositivos, através do encapsulamento de dados em consultas e respostas DNS. Essa técnica é frequentemente usada por atacantes para criar um canal de comunicação com a máquina da vítima a fim de evadir bloqueios de firewalls. Essa evasão é possível porque, por padrão, firewalls permitem o tráfego do protocolo DNS para não comprometer o funcionamento de inúmeras aplicações legítimas, como navegadores web. Dessa forma, atacantes podem utilizar o canal de comunicação discreto criado pela técnica de tunelamento DNS para realizar ativi-

dades maliciosas, como vazamento de dados, upload de arquivos maliciosos e execução remota de comandos arbitrários de sistema operacional (WANG et al., 2021).

Os atacantes utilizam essa técnica após já terem obtido acesso inicial à máquina da vítima, considerando que possam realizar consultas e obter respostas DNS usando o dispositivo comprometido. A partir disso, um atacante pode enviar dados no sentido upstream (i.e. da máquina comprometida para o servidor DNS próprio do atacante), codificando dados e encapsulando-os usando o subdomínio de um nome de domínio, por exemplo, "dados-codificados.dominio-atacante.com", de forma que "dados-codificados" representam os dados que o atacante deseja transferir e "dominio-atacante" representa um nome de domínio de segundo nível previamente registrado e associado a um servidor sob controle do invasor. Por outro lado, os dados podem ser enviados no sentido downstream (i.e. do servidor próprio do atacante para a máquina comprometida) ao serem codificados e transmitidos por meio de campos de RRs DNS, como CNAME, MX, TXT, entre outros (WANG et al., 2021).

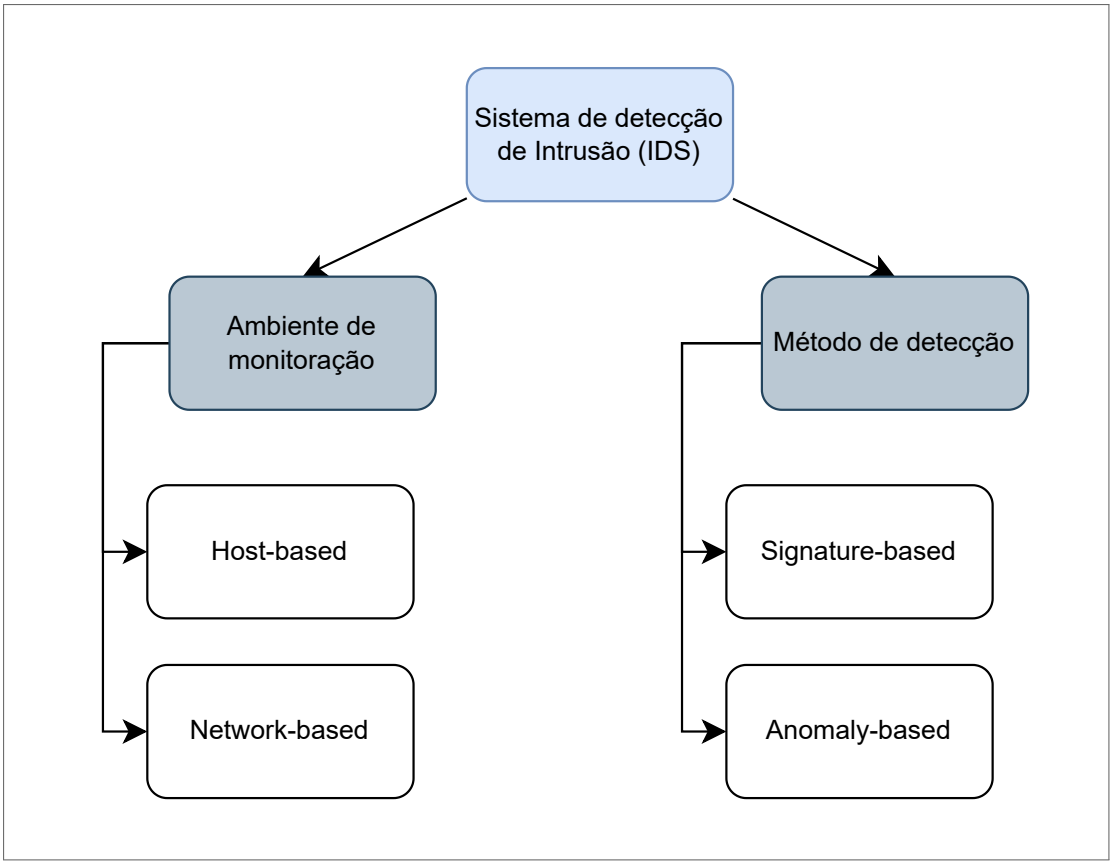
## 2.2 SISTEMA DE DETECÇÃO DE INTRUSÃO (IDS)

Detecção de intrusão é o processo de monitoramento dos eventos que ocorrem em um sistema computacional ou rede e sua análise em busca de sinais de possíveis incidentes, que são violações ou ameaças iminentes de violação de políticas de segurança computacional, políticas de uso aceitável ou práticas padrão de segurança (SCARFONE; MELL, 2007). Sistemas de detecção de intrusão são mecanismos de segurança que automatizam o processo de detecção de intrusão. A Figura 1 ilustra uma forma simples de classificar IDSs com base em diferentes critérios, como ambiente de monitoração e método de detecção.

O ambiente de monitoração faz os IDSs se classificarem em dois tipos: host-based IDS (HIDS) e network-based IDS (NIDS). HIDS inspecionam dados que se originam do sistema de um host, como um servidor ou estação de trabalho, como logs de sistema operacional e logs de aplicação. Por outro lado, NIDS inspecionam dados relacionados ao tráfego de rede, como pacotes de rede ou fluxos de rede que possam indicar atividades suspeitas ou maliciosas.

Outro critério de classificação de IDSs é o método de detecção, que pode ser baseado em assinaturas ou anomalias. IDSs baseados em assinaturas possuem a detecção baseada em técnicas de pattern matching, que consistem em comparar padrões previamente conhecidos com novos dados que são submetidos ao IDS (KHRAISAT et al., 2019). Exemplos de IDSs que utilizam esse método de detecção são SNORT (ROESCH, 1999), BRO (PAXSON, 1999) e

Figura 1 – Classificação de IDSs




**Fonte:** Elaborado pelo autor

Suricata (SURICATA, ). Apesar de essa abordagem geralmente apresentar baixas taxas de falsos positivos e altas taxas de detecção para ataques conhecidos, esses IDSs não costumam ser eficazes para detectar ataques novos e desconhecidos, pois, nesse caso, ainda não existiriam assinaturas dos mesmos (KHRAISAT et al., 2019).

Por outro lado, IDSs baseados em anomalia são capazes de detectar até mesmo ataques nunca vistos antes. Esses IDSs estimam o comportamento normal de um sistema usando métodos estatísticos ou métodos de aprendizagem de máquina. Dessa forma, anomalias são detectadas medindo o desvio de novas atividades em relação ao comportamento normal estimado. Essas anomalias detectadas podem ou não se tratar de intrusões. Por exemplo, uma falha de configuração de uma ferramenta ou um comportamento anormal legítimo podem ser acusados como anomalia, apesar de serem benignos. Por esse motivo de que nem toda anomalia é uma intrusão, IDSs baseados em anomalia tendem a ter maiores taxas de falsos positivos (KHRAISAT et al., 2019).

## 2.3 APRENDIZAGEM DE MÁQUINA (ML)

Aprendizagem de máquina ou machine learning é um subcampo da inteligência artificial focado no estudo e desenvolvimento de algoritmos que aprendem a resolver tarefas a partir de dados históricos, sem serem explicitamente programados (ZHANG et al., 2021) (SAMUEL, 1967). Esses algoritmos costumam ter ganhos de desempenho à medida que mais dados são fornecidos. Algoritmos de ML podem ser classificados em três principais categorias: supervisionados, semi-supervisionados e não-supervisionados. Essas categorias se diferenciam se dados fornecidos para o treinamento necessitam ou não de rótulos, que são valores esperados que o modelo dê como saída para cada exemplo de entrada. Os algoritmos supervisionados requerem dados rotulados, os semi-supervisionados requerem dados parcialmente rotulados e os algoritmos não-supervisionados dispensam a necessidade de qualquer rótulo (ZHANG et al., 2021).

## 2.4 REDES NEURAIS ARTIFICIAIS (ANN)

Redes neurais artificiais consistem em um modelo computacional inspirado no funcionamento do cérebro humano, compostos por camadas de unidades conectadas, conhecidas como neurônios artificiais. Neurônios artificiais possuem pesos multiplicativos e aditivos e funcionam recebendo dados de entrada, aplicando operações lineares, funções de ativação e, por fim, transmitindo o resultado de saída. Essas redes são bastante utilizadas no campo de aprendizagem de máquina, principalmente para o desenvolvimento de soluções baseadas em redes neurais profundas, que são redes neurais com múltiplas camadas de neurônios e são empregadas para tarefas complexas como reconhecimento de fala, geração de imagens, geração de texto, entre outros (ZHANG et al., 2021).

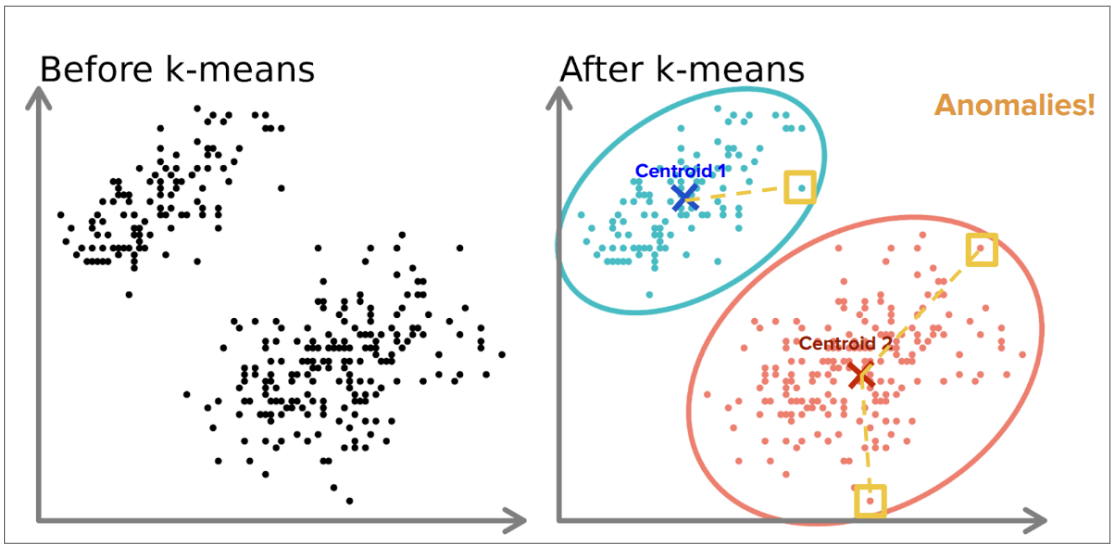
## 2.5 K-MEANS

K-Means é um algoritmo não-supervisionado de aprendizagem de máquina amplamente utilizado para tarefas de clustering. Clustering consiste no problema de agrupar instâncias de dados em grupos, de forma que instâncias de um mesmo grupo sejam semelhantes entre si e instâncias de grupos diferentes sejam distintas (CHANDOLA; BANERJEE; KUMAR, 2009). O algoritmo K-Means realiza o agrupamento a partir de um número fornecido $K$ de grupos. O algoritmo funciona iterativamente, alternando entre duas etapas principais: (1) atribuição

de instâncias ao cluster mais próximo, com base na distância euclidiana, e (2) atualização dos centróides (pontos médios) de cada cluster. O processo continua até que os centróides convergam ou um número máximo de iterações seja atingido (AHMED; SERAJ; ISLAM, 2020).

A detecção de anomalias com o K-Means pode ser realizada partindo da ideia de que instâncias distantes dos centróides de seus respectivos grupos podem ser consideradas anomalias. Portanto, uma abordagem comum para computar o escore de anomalia de uma nova instância é determinar a distância da mesma para o centróide mais próximo. Caso o escore de anomalia seja maior do que um determinado limiar de decisão pré-estabelecido, a instância é acusada como uma anomalia (AHMED; SERAJ; ISLAM, 2020). A figura 2 ilustra o processo de detecção de anomalias do algoritmo K-Means.

Figura 2 – Detecção de anomalias com K-Means




**Fonte:** Elaborado pelo autor. O algoritmo K-Means agrupa as instâncias do conjunto de dados fornecido em $K$ grupos. A distância de instâncias para o centróide do grupo pode ser usada para identificar instâncias anômalas.

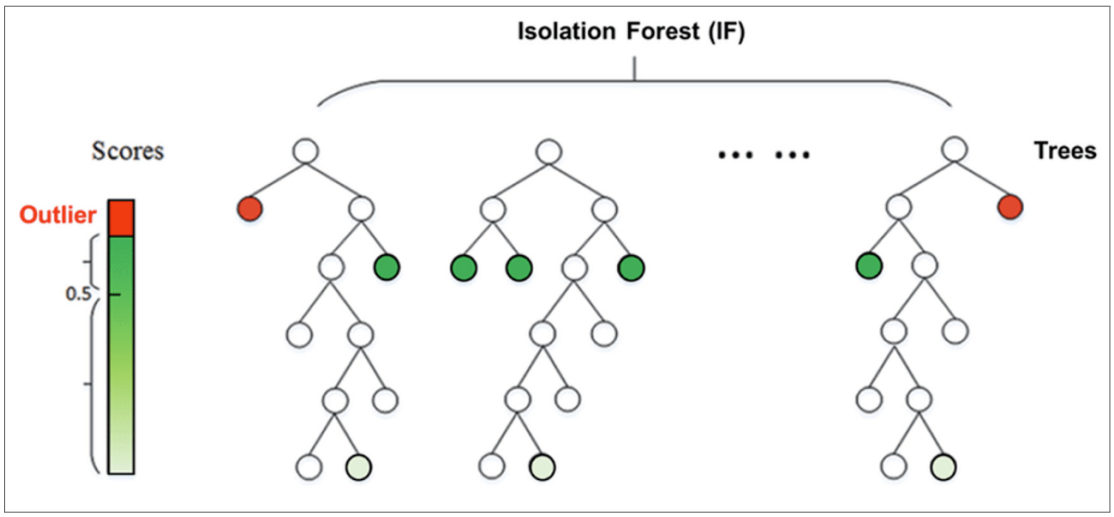
## 2.6 ISOLATION FOREST (IFOREST)

Isolation Forest é um algoritmo não-supervisionado de aprendizagem de máquina projetado para detecção de anomalias. Diferente de métodos tradicionais que focam em criar um modelo do comportamento normal dos dados, o iForest identifica anomalias partindo das premissas de que anomalias são minorias e de que possuem atributos muito diferentes de instâncias normais. O iForest constrói um conjunto de árvores que dividem instâncias de uma amostra de dados de entrada com base em seus atributos, selecionando sucessivamente um atributo, um valor de separação $v$ e separando amostras que possuem valores iguais ou acima de $v$ das que possuem

valores abaixo de $v$. Essas árvores são chamadas de isolation trees ou iTrees e são construídas selecionando aleatoriamente atributos e valores de separação até que todas as instâncias de entrada estejam isoladas umas das outras (LIU; TING; ZHOU, 2008).

A detecção de anomalias com o Isolation Forest é realizada por meio de um escore de anomalia, que mede o quão facilmente uma instância foi isolada. Quanto menor o número de divisões necessárias para isolar uma observação, maior a probabilidade de ela ser uma anomalia. Esse escore é calculado com base no comprimento médio do caminho percorrido nas árvores de isolamento. Um limiar de decisão pode ser definido para classificar as instâncias como normais ou anômalas (LIU; TING; ZHOU, 2008). A figura 3 ilustra o processo de detecção de anomalias do algoritmo Isolation Forest.

Figura 3 – Detecção de anomalias com Isolation Forest




**Fonte:** Elaborado pelo autor. O algoritmo Isolation Forest é composto por um conjunto de isolation trees. Instâncias fáceis de serem isoladas pelas isolation trees tendem a ser anomalias.

## 2.7 SELF-ORGANIZING MAPS (SOM)

Self-Organizing Maps ou Mapas de Kohonen são um tipo de rede neural artificial com treinamento não-supervisionado introduzidas na década de 1980 por Teuvo Kohonen. Essas redes são capazes de reduzir a dimensionalidade de dados de entrada, criando uma representação discretizada enquanto ainda são capazes de preservar relações de similaridade e diferença entre os dados fornecidos. O treinamento dessas redes envolve processos competitivos e cooperativos. O processo competitivo do treinamento ocorre quando os neurônios da rede competem para representar padrões de entrada, sendo eleito como best matching unit (BMU) o neurônio com o vetor de pesos mais próximo ao vetor de dados de entrada fornecido do treinamento. O

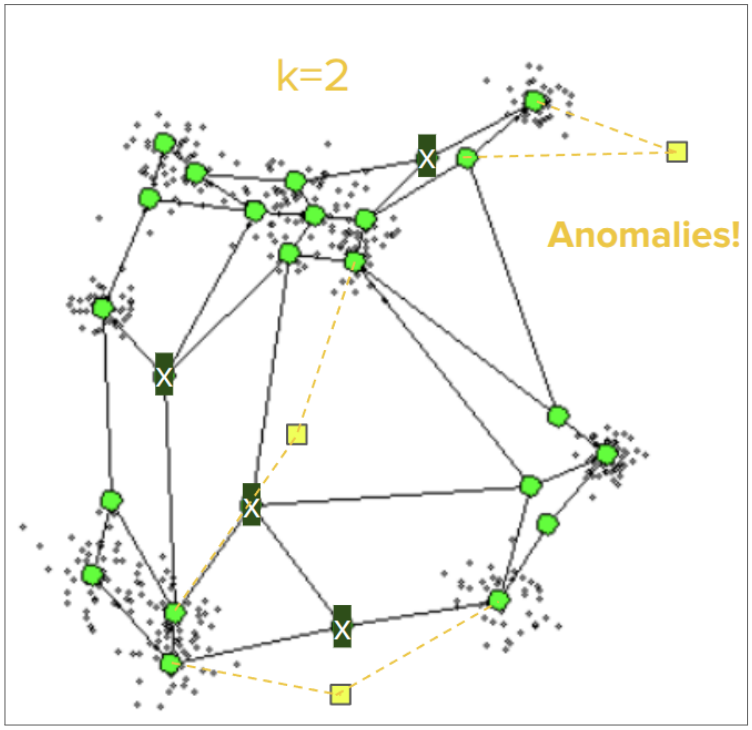
processo cooperativo ocorre quando o BMU compartilha com outros neurônios da vizinhança como os pesos dos mesmos devem ser atualizados para representar melhor o padrão dos dados de entrada, além de atualizar seus próprios pesos. SOM possui diversas aplicações, como redução de dimensionalidade, clustering e até mesmo detecção de anomalias (KOHONEN, 2004) (MILJKOVIć, 2017).

### 2.7.1 SOM-KNN

(TIAN; AZARIAN; PECHT, 2014) propuseram uma variação do algoritmo de SOM combinando-o com o algoritmo K Nearest Neighbors (KNN), que aprimorou o desempenho de SOM para tarefas de detecção de anomalias. Vamos nos referir a essa combinação de algoritmos usada para detecção de anomalias como SOM-KNN durante o restante do trabalho.

SOM-KNN consiste em treinar uma rede neural artificial SOM da mesma forma que no algoritmo original, porém com mudanças depois desse processo. Após o treinamento, SOM-KNN primeiro remove os neurônios da SOM que não foram eleitos BMUs para um número mínimo de instâncias do treinamento e que poderiam representar padrões discrepantes. Em seguida, para computar um escore de anomalia, o algoritmo KNN identifica os $K$ neurônios mais próximos da instância fornecida e o escore de anomalia final é dado como a média aritmética das $K$ distâncias para esses neurônios mais próximos. Por fim, caso o escore de anomalia seja maior do que um determinado limiar de decisão pré-estabelecido, a instância é acusada como uma anomalia. A figura 4 ilustra o processo de detecção de anomalias do algoritmo SOM-KNN.

Figura 4 – Detecção de anomalias com SOM-KNN




**Fonte:** Elaborado pelo autor. O algoritmo SOM-KNN é capaz de identificar anomalias medindo a distância de novas instâncias para os $K$ neurônios mais próximos de uma rede SOM treinada.

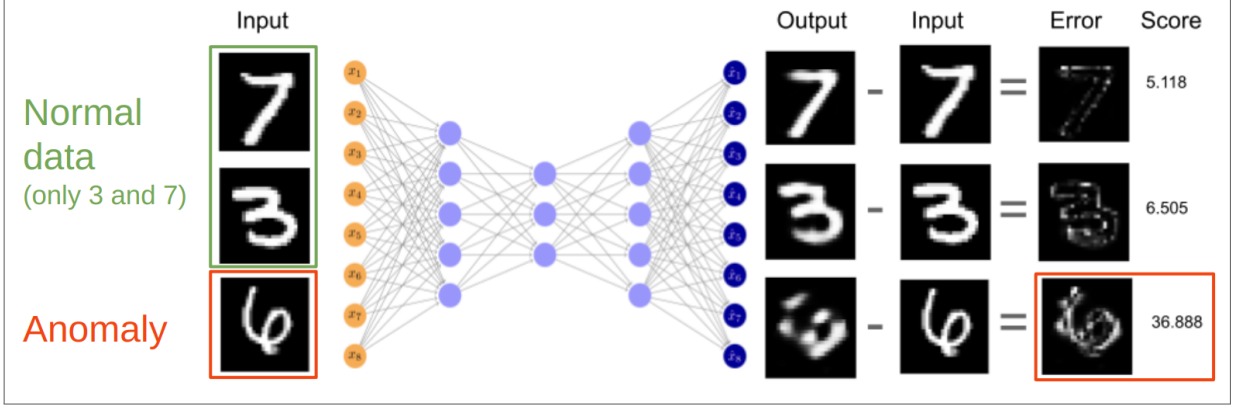
## 2.8 AUTOENCODER

Autoencoders são um tipo de redes neurais artificiais originalmente criados para compressão de dados, devido à sua capacidade de aprender representações de dados a partir de um processo iterativo de codificação e decodificação (PARK et al., 2022). Autoencoders consistem em uma rede neural que pode ser dividida em duas partes: encoder e decoder. O encoder é responsável por comprimir os dados de entrada fornecidos, criando uma representação compacta. Por outro lado, o decoder tem a função de reconstruir os dados de entrada originais a partir da representação comprimida. O processo de compressão e reconstrução possui um erro associado chamado erro de reconstrução.

Além do uso original de compressão de dados, autoencoders também são amplamente utilizados para detecção de anomalias. Com essa finalidade, é realizado o treinamento da rede para que a mesma esteja otimizada para codificar e decodificar dados de uma determinada natureza, que tende a minimizar o erro de reconstrução para esses dados. No entanto, quando dados com características discrepantes são processados, o erro de reconstrução tende a aumentar, visto que o autoencoder não foi otimizado para esse tipo de entrada. Dessa forma, autoencoders são capazes de usar o erro de reconstrução como escore de anomalia e apontar

uma nova instância como suspeita caso esse erro seja maior do que algum limiar de decisão pré-estabelecido (PARK et al., 2022). A figura 5 ilustra o processo de detecção de anomalias de autoencoders.

Figura 5 – Detecção de anomalias com autoencoder




**Fonte:** Elaborado pelo autor. Autoencoders são capazes de identificar anomalias a partir do erro de reconstrução de novas instâncias que distoem do padrão das instâncias usadas no treinamento. Na figura acima, consideramos uma rede que treinou apenas para reconstruir imagens dos números sete e três. Quando fornecida uma imagem do número seis, há um erro maior de reconstrução, identificando uma anomalia.

# 3 TRABALHOS RELACIONADOS

Muitos autores propuseram IDSs supervisionados para detectar atividades de tunelamento DNS. O trabalho (LAMBION et al., 2020) realizou experimentos usando redes neurais convolucionais (CNN) e o algoritmo de random forest, utilizando características linguísticas extraídas de consultas DNS, como quantidade de caracteres e quantidade de dígitos. Esse trabalho atingiu uma acurácia de 96,65% e uma Area Under Curve - Receiver Operating Characteristics (AUC-ROC) de 99,84% ao combinar os dois algoritmos. Em (LAL et al., 2022) foi proposta uma abordagem que que combinou CNN e support vector machines (SVM) atingindo um F1-Score de 99,99%. Apesar dos resultados obtidos por esses trabalhos, eles empregam métodos supervisionados, e consequentemente possuem sua capacidade de detecção dependente dos dados maliciosos usados no treinamento, dessa forma, tendo dificuldades para detectar ataques desconhecidos. Ademais, IDSs supervisionados necessitam da aquisição de dados maliciosos rotulados para o treinamento, o que é um processo custoso.

Por outro lado, outros autores consideraram abordagens não-supervisionadas para detecção de tunelamento DNS. Os autores de (NGUYEN et al., 2020) consideraram o uso do algoritmo density-based spatial clustering of applications with noise (DBSCAN) para detectar anomalias em tráfego de rede, atingindo uma AUC-ROC de até 0,992 no melhor cenário de experimento. Os autores de (AHMED et al., 2019) propuseram um mecanismo de detecção em tempo real não-supervisionado para detectar ataques de tunelamento DNS. Esse mecanismo é baseado no algoritmo de isolation forest, que identifica anomalias em consultas DNS. Para validar a proposta, o sistema foi avaliado em duas redes coorporativas. O método proposto atingiu uma taxa de falsos positivos de 1,56% e uma taxa de verdadeiros positivos de 98,44% para identificar atividades de vazamentos de dados através de túneis DNS. O trabalho (CAMPBELL; ZINCIR-HEYWOOD, 2020) usou o algoritmo de SOM para detectar ataques de tunelamento DNS, extraindo características linguísticas de consultas DNS e inferindo se uma nova consulta é maliciosa ou não com base na classe previamente atribuída aos neurônios da SOM, atingindo resultados de F1-Score de até 99%. Incluímos experimentos com os sistemas propostos em (AHMED et al., 2019) e (CAMPBELL; ZINCIR-HEYWOOD, 2020) para o estudo comparativo.

# 4 ESTUDO COMPARATIVO PROPOSTO

Esse trabalho realiza um estudo comparativo entre técnicas não-supervisionadas de aprendizagem de máquina baseadas em detecção de anomalias para o desenvolvimento de sistemas de detecção de intrusão focados em detectar atividades de exfiltração de dados através de túneis DNS. A proposta é avaliar os resultados e analisar vantagens e desvantagens de cada método, evidenciando os melhores.

Para isso, serão comparados quatro algoritmos não supervisionados amplamente utilizados na detecção de anomalias: Isolation Forest, Autoencoders, K-Means e SOM-KNN. Além desses algoritmos, duas abordagens dos trabalhos relacionados ((AHMED et al., 2019), (CAMPBELL; ZINCIR-HEYWOOD, 2020)) foram incluídas como referências para comparação dos resultados obtidos por nosso estudo com outras propostas da literatura de detecção de tunelamento DNS.

A avaliação do desempenho será realizada com base em métricas tradicionais para problemas de classificação, envolvendo principalmente métricas como acurácia, AUC-ROC, precision, recall, F1-Score, taxas de verdadeiros positivos e taxas de falsos positivos. Essas métricas são relevantes pois possibilitam a compreensão dos erros do tipo falso positivo, que correspondem a alertas falsos e desnecessários que podem sobrecarregar analistas de segurança, e falsos negativos, que correspondem a ataques que não foram detectados. Além disso, também consideramos a comparação entre o tempo de detecção das abordagens, possibilitando a compreensão de quais métodos são mais leves para implantações no mundo real. A análise de resultados a partir dessas métricas permite identificar vantagens e desvantagens de cada método e entender as melhores relações de trade-offs entre potencial de detecção, taxas de falsos positivos e custo computacional.

Este estudo se diferencia de trabalhos anteriores ao focar exclusivamente em métodos não supervisionados de detecção de anomalias, que são particularmente relevantes em cenários onde os dados rotulados são escassos e novos padrões de ataques surgem constantemente. Espera-se que o trabalho contribua para a área de segurança de redes ao fornecer insights sobre a aplicabilidade de métodos não supervisionados em cenários reais de detecção de anomalias em tráfego DNS.

# 5 METODOLOGIA E EXPERIMENTOS

Essa seção descreve a metodologia e experimentos realizados para conduzir o estudo comparativo. Todas as abordagens foram implementadas usando a linguagem Python devido ao seu amplo uso para o desenvolvimento de aplicações baseadas em aprendizagem de máquina e aprendizagem profunda. As abordagens foram implementadas usando principalmente as seguintes bibliotecas: PyTorch (PASZKE et al., 2019), Scikit-learn (PEDREGOSA et al., 2011) e MiniSom (VETTIGLI, 2018). Os experimentos foram realizados utilizando o processador Intel(R) Core i5-12400.

## 5.1 DATASETS

Para avaliar diferentes métodos de detecção com um dataset abrangente e representativo, criamos um dataset combinando consultas DNS maliciosas e benignas obtidas de dois outros datasets base disponibilizados publicamente. O primeiro dataset base, DNS Tunneling Queries for Binary Classification (BUBNOV, 2019), possui FQDNs utilizadas em atividades de vazamento de dados através de túneis DNS usando diferentes ferramentas, como dns2tcp, dnscapy, iodine e tuns. Esses dados foram obtidos transferindo um arquivo de 2 MB através de uma conexão SSH estabelecida pelos túneis DNS. Por outro lado, o segundo dataset base, CAIDA UCSD IPv4 Routed /24 DNS Names (CAIDA, 2021), possui originalmente dezenas de milhões de FQDNs benignos obtidos através de estudos da topologia da Internet.

Portanto, esses dois datasets foram combinados para construir o dataset e os conjuntos de treino, validação e teste. A Tabela 1 apresenta um sumário do número de instâncias benignas (ou normais) e maliciosas divididas nos conjuntos elaborados. O conjunto de treino consiste em apenas instâncias benignas, visto que estamos comparando abordagens não-supervisionadas baseadas em detecção de anomalias, e foi utilizado para treinar os métodos avaliados. O conjunto de validação foi usado para otimizar os hiperparâmetros dos modelos treinados. Por fim, o conjunto de teste foi usado para avaliar os resultados das diferentes soluções e compará-los.

Tabela 1 – Dataset utilizado e divisão em conjuntos de treino, validação e teste

| Conjunto | #Domínios benignos | #Domínios maliciosos |
|---|---|---|
| Treino | 200.000 | 0 |
| Validação | 8.000 | 8.000 |
| Teste | 8.000 | 8.000 |

**Fonte:** Elaborado pelo autor

## 5.2 EXTRAÇÃO DE ATRIBUTOS

Após obter as FQDNs separadas em conjuntos de treino, validação e teste, investigamos quais atributos poderiam ser extraídos das cadeias de caracteres que representam os nomes de domínios. Após uma revisão dos atributos utilizados por trabalhos da literatura de detecção de tunelamento DNS e problemas parecidos envolvendo a detecção de domínios maliciosos, como em trabalhos de detecção de FQDNs geradas por Domain Generation Algorithms (DGAs) (PARK et al., 2022) (ZHAO et al., 2019), apresentamos os atributos extraídos e utilizados em nossas abordagens comparadas conforme a Tabela 2. Os atributos utilizados foram, portanto, os mesmos do trabalho publicado (LUZ et al., 2023) de nossa autoria.

Os atributos extraídos focam em características de consultas DNS que podem ser alteradas em ataques envolvendo tunelamento DNS. Por exemplo, é esperado que consultas que envolvam o comportamento de tunelamento DNS tenham um tamanho maior do que o normal, a fim de melhorar a transmissão de dados. Outro exemplo é que consultas maliciosas podem ter uma taxa de dígitos maior que o normal, como ocorre quando o atacante usa o método de codificação hexadecimal, que usa um conjunto de 16 caracteres, sendo 10 deles dígitos.

Nós adaptamos o atributo "Reputation Value" do trabalho (ZHAO et al., 2019) proposto para detectar domínios maliciosos gerados por DGAs para o contexto de tunelamento DNS. Esse atributo consiste em um valor extraído da popularidade dos $n$-gramas encontrados em FQDNs, somando um peso $W_{N-Gram}(i) = log_2(N \times C_{N-Gram(i)})$ para cada $n$-grama encontrado na cadeia de caracteres do nome de domínio que deseja-se calcular o "Reputation Value". O termo $W_{N-Gram}(i)$ é o peso atribuído ao $i$-ésimo $n$-grama da cadeia de caracteres, $N$ é o número de caracteres do $n$-grama e $C_{N-Gram(i)}$ representa o número de vezes que o $n$-grama apareceu em um conjunto de domínios de referência. Para esse conjunto de referência, foi escolhido o conjunto dos 100.000 FQDNs mais acessados do mundo, obtidos através da lista Majestic Million (MAJESTIC, 2023).

Tabela 2 – Atributos utilizados e descrição

| Feature | Descrição |
|---|---|
| Entropia | Entropia de Shannon para uma cadeia de caracteres |
| Tamanho | Número total de caractes de um domínio |
| Número de subdomínios | Número de subdomínios de um domínio a partir do domínio de terceiro nível (i.e., "www.google.com"tem 1) |
| Tamanho da maior label | Tamanho do maior nível de domínio (i.e., "www.google.com"tem 6) |
| Tamanho da maior sequência de dígitos | Tamanho da maior sequência de dígitos (i.e., "123abcd45ef.net"tem 3) |
| Tamanho da maior sequência de letras | Tamanho da maior sequência de letras (i.e., "123abcd45ef.net"tem 4) |
| Contagem de caracteres especiais | Número total de aparições de caracteres especiais (excluindo pontos ".") |
| Taxa de caracteres especiais | Número total de aparições de caracteres especiais (excluindo pontos ".") divido pelo tamanho do domínio |
| Contagem de dígitos | Número total de aparições de dígitos |
| Taxa de dígitos | Número total de aparições de dígitos dividido pelo tamanho do domínio |
| Contagem de vogais | Número total de aparições de vogais |
| Taxa de vogais | Número total de aparições de vogais dividido pelo tamanho do domínio |
| Reputation value | Valor extraído da popularidade dos $n$-gramas encontrados no domínio |
| Reputation value por $n$-grama | Reputation value divido pelo número de $n$-gramas obtido para cálculo do Reputation value |

**Fonte:** Elaborado pelo autor, publicada em (LUZ et al., 2023)

## 5.3 ABORDAGENS DE DETECÇÃO

Consideramos como abordagens de detecção testadas quatro algoritmos não-supervisionados amplamente utilizados para detecção de anomalias: Isolation Forest, Autoencoders, KMeans e SOM-KNN. Além disso, incluímos os métodos dos trabalhos de referência (AHMED et al., 2019) e (CAMPBELL; ZINCIR-HEYWOOD, 2020), baseados, respectivamente, nos algoritmos de Isolation Forest e SOM. Explicações sobre os algoritmos considerados nos experimentos estão na seção 2.

O ajuste de hiperparâmetros foi realizado com a framework Optuna (AKIBA et al., 2019), que é uma ferramenta de otimização automática de hiperparâmetros baseada em busca bayesiana. A ferramenta constrói um modelo de probabilidade da função objetivo e utiliza-o para selecionar hiperparâmetros com maior probabilidade de otimizar a função objetivo.

A Tabela 3 mostra os hiperparâmetros otimizados de diferentes abordagens, suas descrições e os intervalos de valores ou listas de valores usados para ajustar os hiperparâmetros. A framework Optuna considerou os intervalos fornecidos para escolher um número finito de

valores para serem usados no processo de ajuste de hiperparâmetros. Executamos até 100 experimentos de otimização de hiperparâmetros com o Optuna para cada uma das abordagens. A métrica que definimos para ser otimizada é a de AUC-ROC.

Os modelos de referência foram reproduzidos dos artigos (AHMED et al., 2019) e (CAMPBELL; ZINCIR-HEYWOOD, 2020), treinados, validados e testados com os mesmos conjuntos que utilizamos para nossas outras abordagens. Para a referência (AHMED et al., 2019) usamos o mesmo método de ajuste de hiperparâmetros que o nosso experimento com o isolation forest, porém com o método de extração de atributos proposto pelo trabalho de referência.

Para a referência (CAMPBELL; ZINCIR-HEYWOOD, 2020) reproduzimos a implementação conforme as informações disponíveis no artigo, porém, ao treinar o modelo apenas com dados benignos, mesmo com 500 iterações de otimização, não foi possível atingir resultados satisfatórios. Isso ocorreu devido à existência de neurônios da rede classificados como benignos que, na verdade, representavam comportamentos outliers de pouquíssimas instâncias de dados benignos. Portanto, da mesma forma que na nossa abordagem de SOM, adicionamos o argumento "min_times_as_bmu" para excluir neurônios da SOM que não foram eleitos BMUs para um número mínimo de instâncias.

Tabela 3 – Hiperparâmetros otimizados

| **Abordagem** | **Hiperparâmetro** | **Descrição** | **Intervalo ou valores testados** |
|---|---|---|---|
| KMeans | K | Número de clusters | $\{ x \in Z \mid 2 \leq x \leq 30\}$ |
| iForest | n_estimators | Número de estimadores base (isolation trees) | $\{ x \in Z \mid 2 \leq x \leq 100\}$ |
| iForest | max_samples | Porcentagem de amostras extraídas do conjunto de treino para treinar cada estimator base (isolation tree) | $\{ x \in R \mid 0,1 \leq x \leq 1\}$ |
| iForest | contamination | Proporção de anomalias no conjunto de treino | $\{ x \in R \mid 0 \leq x \leq 0.3\}$ |
| SOM, SOM-KNN | size | Quantidade de neurônios dispostos em matriz bidimensional para criar a rede SOM | {(20,20), (30,30), (35,35)} |
| SOM, SOM-KNN | iterations | Número de iterações de treinamento | $\{ x \in Z \mid 1.000 \leq x \leq 70.000\}$ |
| SOM, SOM-KNN | learning_rate | Taxa de aprendizado dos neurônios da rede SOM | $\{ x \in R \mid 0,0001 \leq x \leq 0.8\}$ |
| SOM, SOM-KNN | sigma | Raio inicial da função de vizinhança | $\{ x \in R \mid 0,1 \leq x \leq 15\}$ |
| SOM, SOM-KNN | min_times_as_bmu | Percentual mínimo de vezes com base nas instâncias do conjunto de treinamento que um neurônio precisa ter sido eleito como BMU para que o mesmo seja usado para computar escores de anomalia | $\{ x \in R \mid 0,0001 \leq x \leq 0,005\}$ |
| SOM-KNN | learning_decay | Taxa de decaimento da learning_rate da SOM por iteração de treinamento | $\{ x \in R \mid 0,0001 \leq x \leq 0,001\}$ |
| SOM-KNN | radius_decay | Taxa de decaimento do sigma da SOM por iteração de treinamento | $\{ x \in R \mid 0,0001 \leq x \leq 2\}$ |
| SOM-KNN | number_of_neighbors | Número de neurônios da SOM usados para calcular a distância de uma nova instância a ter seu escore de anomalia avaliado | $\{ x \in Z \mid 1 \leq x \leq 15\}$ |
| Autoencoder | hidden_layer_sizes | Número de neurônios por camada escondida | {(7,4,7), (10,5,10), (12,6,12), (9,5,3,5,9), (10,6,4,6,10), (12,7,5,7,12)} |
| Autoencoder | learning_rate | Taxa de aprendizado dos neurônios | $\{ 1 \times 10^{-4}, 5 \times 10^{-4}, 1 \times 10^{-3} \}$ |
| Autoencoder | batch_sizes | Quantidade de instâncias utilizadas em conjunto para atualizar os pesos dos neurônios da rede | {128, 256, 512} |
| Autoencoder | dropout_rate | Taxa de neurônios com as outputs removidas durante o treinamento para evitar overfitting | {0, 0,1, 0,2} |
| Autoencoder | epochs | Número de vezes que todas as instâncias do conjunto de treinamento são usadas | {20} |

**Fonte:** Elaborado pelo autor. "iForest" se refere tanto à baseline (AHMED et al., 2019) quanto ao modelo testado com a extração de atributos proposta por nosso trabalho. "SOM" se refere à baseline (CAMPBELL; ZINCIR-HEYWOOD, 2020).

## 5.4 MÉTRICAS DE AVALIAÇÃO

As abordagens foram avaliadas e comparadas utilizando métricas amplamente usadas em problemas de classificação binária. As métricas utilizadas foram: acurácia, taxa de falso positivo (FPR), taxa de verdadeiro positivo ou recall (TPR), precision, f1-score e área sob a curva ROC (AUC-ROC).

Essas métricas nos possibilitam avaliar adequadamente o desempenho de cada abordagem e a relação de troca entre o aumento da taxa de detecção dos modelos e o aumento de falsos positivos. Destacamos a métrica AUC-ROC por ser independente da escolha de um limiar de decisão, possibilitando avaliar a relação de TPR e FPR de maneira holística. As métricas estão descritas conforme as equações abaixo:

$$\text{Acurácia} = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.1}$$

$$\text{FPR} = \frac{FP}{FP + TN} \tag{5.2}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{5.3}$$

$$\text{TPR} = \text{Recall} = \frac{TP}{TP + FN} \tag{5.4}$$

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.5}$$

$$\text{AUC-ROC} = \int_0^1 \text{TPR}(FPR)\, d(FPR), \tag{5.6}$$

# 6 RESULTADOS

Nessa seção, vamos apresentar os resultados de detecção de cada uma das abordagens descritas previamente. A tabela 4 mostra os resultados das diferentes abordagens testadas. Para escolher o limiar de decisão ótimo para cada abordagem, usamos o index de Youden (RUOPP et al., 2008) considerando o máximo de 10% de FPR. Fizemos esse processo para todas as abordagens exceto com a de (CAMPBELL; ZINCIR-HEYWOOD, 2020), que não é originalmente baseada no cálculo de um escore de anomalia e não usa um limiar de decisão, fornecendo um resultado de detecção inflexível.

Além disso, avaliamos os resultados das abordagens para cenários com restrição nas taxas de falsos positivos, que são mais próximos da realidade de implementação de IDSs. Altas taxas de falsos positivos são problemáticas pois cada evento falso positivo gera uma carga de trabalho desnecessária para um analista, que em casos mais extremos, pode ser tão grande a ponto de tornar a implantação de um IDS inviável. Portanto, avaliamos um cenário restringindo a taxa de falsos positivos para até 1%, também escolhendo o limiar de decisão ótimo com o index de Youden. Esses resultados podem ser vistos conforme a tabela 5. Não consideramos o método (CAMPBELL; ZINCIR-HEYWOOD, 2020) nesses resultados, pois o mesmo não conta com o uso de limiar de decisão, portanto, impossibilitando a redução da taxa de falsos postivos.

Tabela 4 – Resultados

| Abordagem | FPR | TPR | Precision | Recall | F1-score | Acurácia | Tempo de inferência |
|---|---|---|---|---|---|---|---|
| iForest | 10,00% | 83,69% | 0,8932 | 0,8369 | 0,8641 | 86,84% | 223 ± 9 ms |
| KMeans | 10,00% | 90,51% | 0,9005 | 0,9051 | 0,9028 | 90,26% | 30 ± 4 ms |
| Autoencoder | 9,50% | 93,50% | 0,9077 | 0,9350 | 0,9211 | 92,00% | **16 ± 6 ms** |
| SOM-KNN | 9,86% | **98,58%** | 0,9090 | **0,9858** | **0,9458** | **94,36%** | 108 ± 7 ms |
| iForest (baseline) (AHMED et al., 2019) | 9,76% | 98,20% | **0,9095** | 0,9820 | 0,9444 | 94,22% | 149 ± 4 ms |
| SOM (baseline) (CAMPBELL; ZINCIR-HEYWOOD, 2020) | 13,34% | 97,83% | 0,8800 | 0,9783 | 0,9265 | 92,24% | 94 ± 7 ms |

**Fonte:** Elaborado pelo autor. O tempo de inferência fornecido é o tempo necessário para processar uma amostra de 200.000 instâncias.

As Figuras 6 e 7 apresentam as curvas ROC, que mostram a métrica AUCROC e a relação de troca entre taxas de falsos positivos e verdadeiros positivos que ocorre a partir da escolha de diferentes limiares de decisão das abordagens testadas. A Figura 7 é uma ampliação da Figura 6 para o limite de até 5% de taxa de falso positivo. Dessa forma, avaliamos os desempenhos

Tabela 5 – Resultados em até 1% de FPR

| Abordagem | FPR | TPR | Precision | Recall | F1-score | Acurácia | Tempo de inferência |
|---|---|---|---|---|---|---|---|
| iForest | 1,00% | 74,81% | 0,9868 | 0,7481 | 0,8510 | 86,91% | 223 ± 9 ms |
| KMeans | 0,39% | 74,54% | **0,9948** | 0,7454 | 0,8522 | 87,08% | 30 ± 4 ms |
| Autoencoder | 1,00% | 76,65% | 0,9871 | 0,7665 | 0,8629 | 87,83% | **16 ± 6 ms** |
| SOM-KNN | 1,00% | **77,98%** | 0,9873 | **0,7798** | **0,8714** | **88,49%** | 108 ± 7 ms |
| iForest (baseline) (AH-MED et al., 2019) | 0,85% | 74,54% | 0,9887 | 0,7454 | 0,8500 | 86,84% | 149 ± 4 ms |

**Fonte:** Elaborado pelo autor. O tempo de inferência fornecido é o tempo necessário para processar uma amostra de 200.000 instâncias.

das abordagens de forma independente da escolha de um limiar de decisão até mesmo para taxas reduzidas de falso positivo.

Figura 6 – Curvas ROC

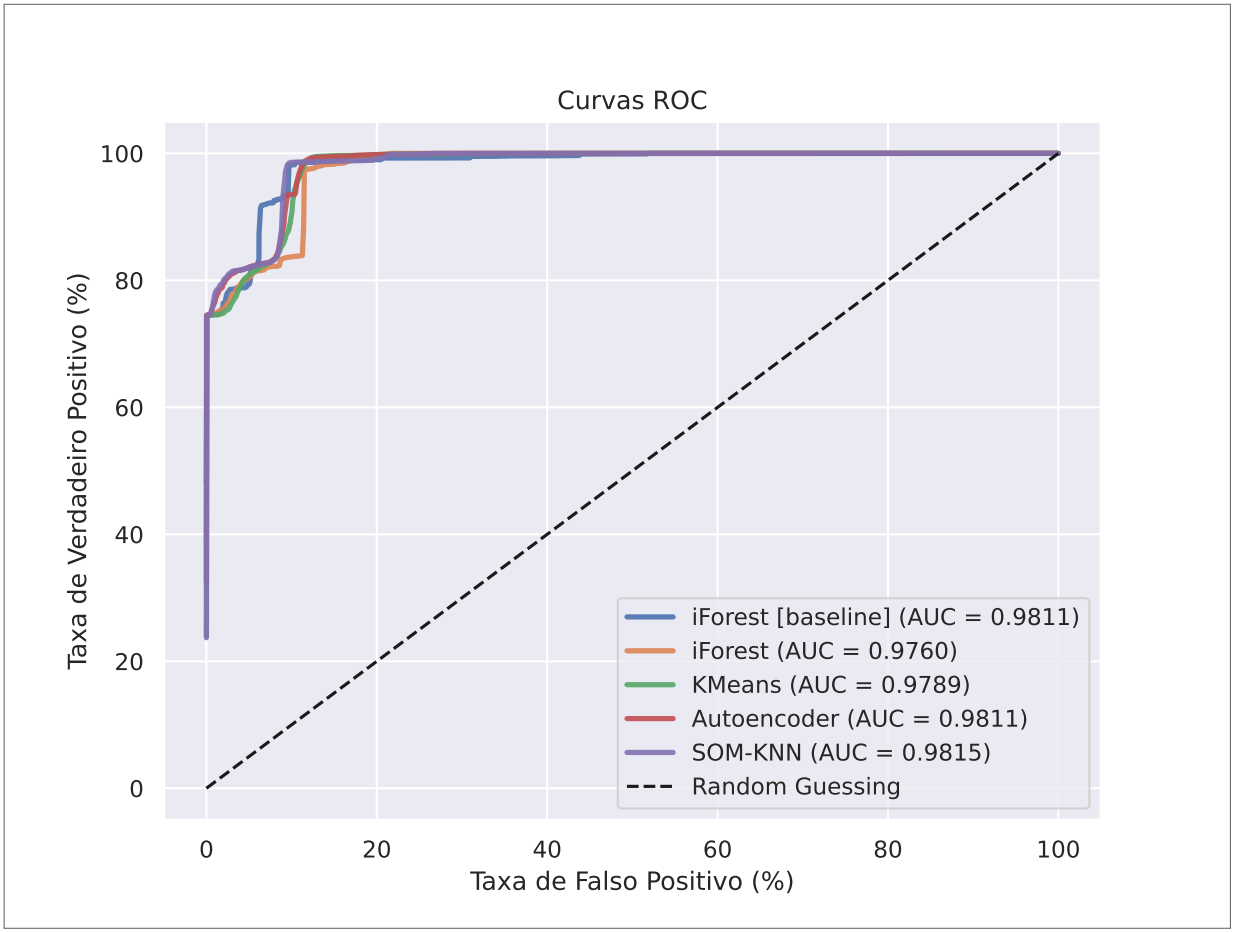
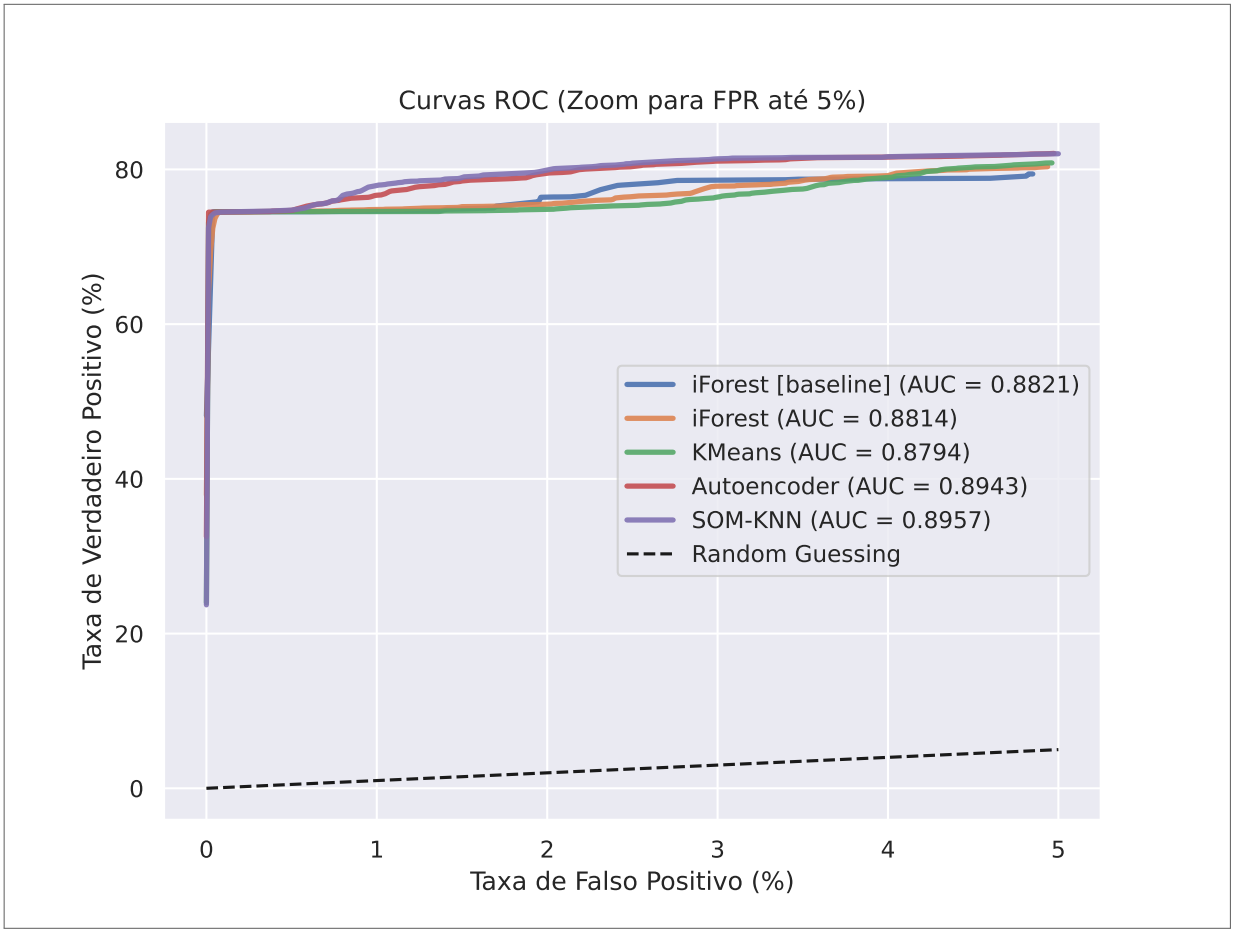


Figura 7 – Curvas ROC em até 5% de FPR



# 7 DISCUSSÃO

Nesse capítulo, vamos analisar e discutir os resultados apresentados no capítulo 6, evidenciando as melhores abordagens.

## 7.1 RESULTADOS DE DETECÇÃO

Em geral, não houve variações muito grandes entre o desempenho de detecção das abordagens. Todas as abordagens tiveram uma AUC-ROC no intervalo de 0,9760 e 0,9815, conforme a Figura 6 e quando limitamos essa visão para até 5% de FPR, tiveram uma AUC-ROC no intervalo de 0,8794 e 0,8957, conforme a Figura 7. Além disso, essa última figura mostra que a relação entre as taxas FPR e TPR é bastante parecida para taxas de FPR de até 0,5%.

Mesmo sem variações grandes, houveram diferenças significativas entre o desempenho de detecção das abordagens. A Tabela 4 mostra que, ao considerar uma taxa de falsos positivos de cerca de 10%, a melhor abordagem no geral foi a de SOM-KNN, que obteve os melhores resultados nas métricas de TPR ou Recall, F1-Score e Acurácia. Porém, esses resultados ficaram próximos da abordagem de (AHMED et al., 2019). Por outro lado, no pior dos casos, a abordagem de isolation forest com os atributos usados no estudo comparativo teve uma taxa de verdadeiro positivo de cerca de 15 pontos percentuais a menos que o modelo SOM-KNN, o que mostra uma diferença significativa de desempenho.

Ao avaliar os resultados para taxas de falso positivo de até 1% a partir da Tabela 5, o modelo SOM-KNN também forneceu os melhores resultados de detecção, com as melhores métricas de TPR ou Recall, F1-Score e acurácia. Dessa vez, a abordagem que mais se aproximou dos resultados de SOM-KNN foi a baseada em autoencoder. Os resultados das curvas ROC também mostram o melhor desempenho do modelo SOM-KNN, que atingiu a melhor AUC-ROC geral e para taxas de FPR de até 5%.

## 7.2 TEMPO DE INFERÊNCIA

Houve diferenças significativas entre os desempenhos das abordagens ao analisar o tempo de inferência das mesmas. Por exemplo, o tempo de inferência da abordagem de Isolation Forest com os atributos do nosso estudo, chega a ser aproximadamente 14 vezes o tempo de

inferência do Autoencoder. Essa diferença pode ser significativa para decidir qual método será usado em um IDS que lide com altas volumetrias de eventos.

Destacamos que, além do algoritmo, a diferença de tempo de inferência pode ser causada pela escolha de hiperparâmetros que impactem na complexidade do modelo. Por exemplo, após o ajuste de hiperparâmetros, a nossa abordagem de Isolation Forest teve o número ideal de estimadores base (isolation trees) como 19, porém a abordagem de Isolation Forest de (AHMED et al., 2019) teve um número ideal de 12 estimadores base. Portanto, essa escolha impactou na complexidade dos modelos, tornando o modelo da abordagem de (AHMED et al., 2019) mais rápido ao realizar inferências. O Autoencoder foi outro modelo beneficiado pela escolha de hiperparâmetros, principalmente devido à simples arquitetura de 3 camadas escondidas de 12, 6 e 12 neurônios, respectivamente, que tornou o mesmo com a menor complexidade computacional de todas as abordagens testadas, conforme podemos ver pelo seu tempo de inferência apresentado nas Tabelas 4 e 5.

## 7.3 CONSIDERAÇÕES FINAIS

O estudo comparativo conduzido por esse trabalho mostra que, em geral, o modelo SOM-KNN teve um melhor desempenho de detecção em comparação aos demais, porém as abordagens de autoencoder, (CAMPBELL; ZINCIR-HEYWOOD, 2020) e (AHMED et al., 2019) fornecem um desempenho próximo, com destaque para a abordagem de autoencoder que possui a menor complexidade computacional. Apesar do método (CAMPBELL; ZINCIR-HEYWOOD, 2020) ter apresentado resultados competitivos, uma desvantagem do mesmo é a falta de um mecanismo como um limiar de decisão para controlar a taxa de falsos positivos, o que pode dificultar a implantação desse método em ambientes reais. Por fim, o melhor modelo a ser utilizado para a criação de um IDS depende dos requisitos do mesmo, em relação à complexidade computacional e taxas estabelecidas como satisfatórias de FPR e TPR.

# 8 CONCLUSÃO E TRABALHOS FUTUROS

Neste trabalho, realizamos um estudo comparativo entre métodos de aprendizagem de máquina não-supervisionados para construir IDSs baseados em anomalias com foco em detectar atividades de vazamento de dados através de túneis DNS. Esse trabalho possibilitou a publicação do artigo (LUZ et al., 2023) no SBSeg 2023 e a realização da oficina (ARCOVERDE et al., 2024) no SBSeg 2024.

Em nossa metodologia e experimentos, utilizamos um conjunto de dados considerando consultas DNS maliciosas geradas pelas ferramentas dns2tcp, dnscapy, iodine e tuns. Propusemos um novo conjunto de atributos extraídos que capturam características relevantes para distinguir consultas DNS benignas de maliciosas. A partir do uso dos novos atributos, comparamos os algoritmos KMeans, Isolation Forest, Autoencoder e SOM-KNN com duas outras soluções também não-supervisionadas baseadas em aprendizagem de máquina da literatura (AHMED et al., 2019) (CAMPBELL; ZINCIR-HEYWOOD, 2020). A análise comparativa realizada considerando desempenho de detecção e tempo de inferência mostrou que a abordagem SOM-KNN obteve, em geral, os melhores resultados. Além disso, as abordagens de autoencoder, (AHMED et al., 2019) e (CAMPBELL; ZINCIR-HEYWOOD, 2020) também tiveram resultados próximos e competitivos.

Por fim, trabalhos futuros podem ser realizados investigando métodos de detecção de tunelamento DNS focados em volumetria, e não na inspeção de pacotes individuais. Além disso, outras oportunidades de trabalhos futuros estão na avaliação de métodos de detecção de tunelamento DNS para cenários mais desafiadores, como túneis DNS que limitem a quantidade de caracteres, aumentando a complexidade de detecção.

# REFERÊNCIAS

AHMED, J.; GHARAKHEILI, H. H.; RAZA, Q.; RUSSELL, C.; SIVARAMAN, V. Real-time detection of dns exfiltration and tunneling from enterprise networks. In: *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. [S.l.: s.n.], 2019. p. 649–653.

AHMED, M.; SERAJ, R.; ISLAM, S. M. S. The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics*, v. 9, n. 8, 2020. ISSN 2079-9292. Disponível em: <https://www.mdpi.com/2079-9292/9/8/1295>.

AKIBA, T.; SANO, S.; YANASE, T.; OHTA, T.; KOYAMA, M. Optuna: A next-generation hyperparameter optimization framework. In: *The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. [S.l.: s.n.], 2019. p. 2623–2631.

ARAUJO-FILHO, P. Freitas de; KADDOUM, G.; CAMPELO, D. R.; SANTOS, A. G.; MACêDO, D.; ZANCHETTIN, C. Intrusion detection for cyber–physical systems using generative adversarial networks in fog environment. *IEEE Internet of Things Journal*, v. 8, n. 8, p. 6247–6256, 2021.

ARCOVERDE, H.; ARAUJO-FILHO, P.; LUZ, J.; ROCHA, P. Empowering intrusion detection systems with machine learning. In: SBC. *Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais 2024 (SBSeg 2024)*. São José dos Campos, 2024.

BUBNOV, Y. *DNS Tunneling Queries for Binary Classification*. 2019. Mendeley Data.

CAIDA. *The CAIDA UCSD IPv4 Routed /24 DNS Names Dataset*. 2021. Https://www.caida.org/catalog/datasets/ipv4_dnsnames_dataset/.

CAMPBELL, A. J.; ZINCIR-HEYWOOD, N. Exploring tunneling behaviours in malicious domains with self-organizing maps. In: *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. [S.l.: s.n.], 2020. p. 1419–1426.

CANIATO, B. *Segurança em risco: a ameaça crescente dos ataques de hackers a órgãos públicos*. 2024. Disponível em: <https://veja.abril.com.br/brasil/seguranca-em-risco-a-ameaca-crescente-dos-ataques-de-hackers-a-orgaos-publicos>. Acesso em: 19 fev. 2025.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection: A survey. *ACM Comput. Surv.*, Association for Computing Machinery, New York, NY, USA, v. 41, n. 3, jul. 2009. ISSN 0360-0300. Disponível em: <https://doi.org/10.1145/1541880.1541882>.

CISCO. *DNSpionage brings out the Karkoff*. 2019. Disponível em: <https://blogs.cisco.com/security/talos/dnspionage-brings-out-the-karkoff>. Acesso em: 19. mar. 2025.

FISHER, D. *Researchers Uncover PinkStats APT Toolkit*. 2013. Disponível em: <https://threatpost.com/researchers-uncover-pinkstats-apt-toolkit/101077/>. Acesso em: 19. mar. 2025.

FORUM, W. E. *Previous Global Future Council on the Future of Cybersecurity.* 2025. Disponível em: <https://www.weforum.org/communities/gfc-on-cybersecurity/>. Acesso em: 19 fev. 2025.

KHRAISAT, A.; GONDAL, I.; VAMPLEW, P.; KAMRUZZAMAN, J. Survey of intrusion detection systems: techniques, datasets and challenges. *Cybersecurity*, v. 2, 12 2019.

KOHONEN, T. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, v. 43, p. 59–69, 2004. Disponível em: <https://api.semanticscholar.org/CorpusID:206775459>.

KUROSE, J. F.; ROSS, K. W. *Computer Networking: A Top-Down Approach*. 7. ed. Boston, MA: Pearson, 2016. ISBN 978-0-13-359414-0.

LAL, A.; PRASAD, A.; KUMAR, A.; KUMAR, S. Dns-tunnet: A hybrid approach for dns tunneling detection. In: *2022 4th International Conference on Advances in Computer Technology, Information Science and Communications (CTISC)*. [S.l.: s.n.], 2022. p. 1–6.

LAMBION, D.; JOSTEN, M.; OLUMOFIN, F.; COCK, M. D. Malicious dns tunneling detection in real-traffic dns data. In: *2020 IEEE International Conference on Big Data (Big Data)*. [S.l.: s.n.], 2020. p. 5736–5738.

LIU, F. T.; TING, K. M.; ZHOU, Z.-H. Isolation forest. In: *2008 Eighth IEEE International Conference on Data Mining*. [S.l.: s.n.], 2008. p. 413–422.

LUZ, J.; ARAUJO-FILHO, P.; ARCOVERDE, H.; CAMPELO, D. Unsupervised som-based intrusion detection system for dns tunneling attacks. In: *Anais do XXIII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*. Porto Alegre, RS, Brasil: SBC, 2023. p. 516–521. ISSN 0000-0000. Disponível em: <https://sol.sbc.org.br/index.php/sbseg/article/view/27232>.

MAJESTIC. *Top 1 million websites in the world*. 2023. Https://majestic.com/reports/majestic-million.

MILJKOVIć, D. Brief review of self-organizing maps. In: *2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. [S.l.: s.n.], 2017. p. 1061–1066.

MITRE ATTCK. *FIN7 - Groups*. 2017. Disponível em: <https://attack.mitre.org/groups/G0046/>. Acesso em: 19. mar. 2025.

MITRE ATTCK. *OilRig - Groups*. 2017. Disponível em: <https://attack.mitre.org/groups/G0049/>. Acesso em: 19. mar. 2025.

MOCKAPETRIS, P. *DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION*. [S.l.], 1987. Disponível em: <https://www.rfc-editor.org/rfc/rfc1035.txt>.

NGUYEN, T. Q.; LABORDE, R.; BENZEKRI, A.; QU'HEN, B. Detecting abnormal dns traffic using unsupervised machine learning. In: *2020 4th Cyber Security in Networking Conference (CSNet)*. [S.l.: s.n.], 2020. p. 1–8.

PAIREDS. *DNS Tunneling: The Risks and Real Examples*. 2015. Disponível em: <https://paireds.com/dns-tunneling-the-risks-and-real-examples/>. Acesso em: 19 fev. 2025.

PARK, K. H.; SONG, H. M.; YOO, J. D.; HONG, S.-Y.; CHO, B.; KIM, K.; KIM, H. K. Unsupervised malicious domain detection with less labeling effort. *Comput. Secur.*, Elsevier Advanced Technology Publications, GBR, v. 116, n. C, may 2022. ISSN 0167-4048. Disponível em: <https://doi.org/10.1016/j.cose.2022.102662>.

PASZKE, A.; GROSS, S.; MASSA, F.; LERER, A.; BRADBURY, J.; CHANAN, G.; KILLEEN, T.; LIN, Z.; GIMELSHEIN, N.; ANTIGA, L.; DESMAISON, A.; KOPF, A.; YANG, E.; DEVITO, Z.; RAISON, M.; TEJANI, A.; CHILAMKURTHY, S.; STEINER, B.; FANG, L.; BAI, J.; CHINTALA, S. Pytorch: An imperative style, high-performance deep learning library. In: WALLACH, H.; LAROCHELLE, H.; BEYGELZIMER, A.; BUC, F. d'Alché; FOX, E.; GARNETT, R. (Ed.). *Advances in Neural Information Processing Systems 32*. [S.l.]: Curran Associates, Inc., 2019. p. 8024–8035.

PAXSON, V. Bro: a system for detecting network intruders in real-time. *Comput. Netw.*, Elsevier North-Holland, Inc., USA, v. 31, n. 23–24, p. 2435–2463, dez. 1999. ISSN 1389-1286. Disponível em: <https://doi.org/10.1016/S1389-1286(99)00112-7>.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.

RIJSWIJK-DEIJ, R. van; JONKER, M.; SPEROTTO, A.; PRAS, A. A high-performance, scalable infrastructure for large-scale active dns measurements. *IEEE Journal on Selected Areas in Communications*, v. 34, n. 6, p. 1877–1888, 2016.

ROESCH, M. Snort - lightweight intrusion detection for networks. In: *Proceedings of the 13th USENIX Conference on System Administration*. USA: USENIX Association, 1999. (LISA '99), p. 229–238.

RUOPP, M. D.; PERKINS, N. J.; WHITCOMB, B. W.; SCHISTERMAN, E. F. Youden index and optimal cut-point estimated from observations affected by a lower limit of detection. *Biometrical Journal*, v. 50, n. 3, p. 419–430, 2008. Disponível em: <https://doi.org/10.1002/bimj.200710415>.

SAMUEL, A. L. Some studies in machine learning using the game of checkers. *IBM J. Res. Dev.*, v. 44, p. 206–227, 1967. Disponível em: <https://api.semanticscholar.org/CorpusID:2126705>.

SCARFONE, K. A.; MELL, P. M. *SP 800-94. Guide to Intrusion Detection and Prevention Systems (IDPS)*. Gaithersburg, MD, USA, 2007.

SURICATA. *Suricata*. Disponível em: <https://suricata.io/>. Acesso em: 22 fev. 2025.

TIAN, J.; AZARIAN, M.; PECHT, M. Anomaly detection using self-organizing maps-based k-nearest neighbor algorithm. In: *PHM Society European Conference*. [s.n.], 2014. v. 2, n. 1. Disponível em: <https://doi.org/10.36001/phme.2014.v2i1.1554>.

UNIT42. *xHunt Campaign: Attacks on Kuwait Shipping and Transportation Organizations*. 2019. Disponível em: <https://unit42.paloaltonetworks.com/xhunt-campaign-attacks-on-kuwait-shipping-and-transportation-organizations/>. Acesso em: 19. mar. 2025.

UNIT42. *Understanding DNS Tunneling Traffic in the Wild*. 2023. Disponível em: <https://unit42.paloaltonetworks.com/dns-tunneling-in-the-wild/>. Acesso em: 19 fev. 2025.

VETTIGLI, G. *MiniSom: minimalistic and NumPy-based implementation of the Self Organizing Map*. 2018. Disponível em: <https://github.com/JustGlowing/minisom/>.

WANG, Y.; ZHOU, A.; LIAO, S.; ZHENG, R.; HU, R.; ZHANG, L. A comprehensive survey on dns tunnel detection. *Computer Networks*, Elsevier, v. 197, p. 108322, 2021.

ZHANG, A.; LIPTON, Z. C.; LI, M.; SMOLA, A. J. *Dive into Deep Learning*. Cambridge University Press, 2021. Disponível em: <https://d2l.ai/>.

ZHAO, H.; CHANG, Z.; BAO, G.; ZENG, X.; CHAEIKAR, S. S. Malicious domain names detection algorithm based on n-gram. *J. Comput. Netw. Commun.*, Hindawi Limited, London, GBR, v. 2019, jan 2019. ISSN 2090-7141. Disponível em: <https://doi.org/10.1155/2019/4612474>.